Edicad

Piece Programming

User Manual

Version 2.6

www.tpaspa.it

Table of Contents

1 Preliminary

1.1 Preliminary

Edicad is a graphic editor module designed to provide writing, updating and maintenance of the part programs for the Numerical Control
The main *Edicad* features allow:

- creating a new working program
- assigning all the geometrical characteristics of the piece (shape and dimensions)
- inserting on the working faces elementary or complex (Macro) instructions
- linking the most recurrent instructions, in order to create a Subroutines library
- drawing geometrical profiles, that can be associated to technological working processes
- saving and printing the programs
- managing the program, in the "text" format, for modifying, cutting or moving the working instructions

To avoid overcharging this guide, for further information concerning the use of the *mouse*, *menus* and *toolbar* and all the current operative functions of Windows, we refer the reader to Windows® Operative System manuals.

1.2 How to use this manual

This manual has been designed to allow an easy learning of the system and its use.

The main arguments of the following sections are:
- Edicad windows
- definition of the piece and setting the main data.
- tools for inserting and changing the workings.
- using parametric programming and logical conditions.
- creation and use of Subroutines.
- Errors description
- customization of the work environment

1.3 Edicad windows

According with the usual Windows standards, the Edicad main graphic page includes different specific areas as, starting from the top:

Title bar includes the name of the program in process

Menu bar includes different literal menus, each one allowing a number of different operative features, in some case directly selectable by the corresponding Toolbar Button

Selections bar includes:
a set of buttons which allow you to activate or deactivate selection, insertion and positioning modes;
two boxes displaying:
the X and Y coordinates of the position of the mouse on the piece
correct dimensions of the mill-radius profile (only if the *mill-radius* is

active).

Toolbar contains the buttons which give quick access to commonly used commands and tools. The composition of the toolbar changes according to the window which is active at the time, as it is dynamically modified to contain only the commands which are necessary and which can be activated in the particular context.

Programming Area used for viewing the windows relevant to the program section being examined; these windows include a Main window, a General View window, which gives a three-dimensional view of the piece, and the Face View windows which display the selected face on a plane.

Palette these buttons are used for quick activation of tools or machining operations.

Status bar contains different information depending on whether the General View or Face View window is active.

Specifically, in the General View window it displays information regarding the comment on the piece and the dimensions.
In the Face View window it displays information relevant to the current operation. The contents of various fields can be customized from the ***Setup->Editor*** menu***.***

Each box displays the following, in order :

Operating Code and Work Name
consists of the operating code for the particular machining operation, enclosed within square brackets, followed by a descriptive name for the operation.

First custom area
may include from one to fifteen of the following information:
X axis position
Y axis position
Z axis position
X axis Center position
Y axis Center position
Z axis Center position
radius R of the arc
A rotation axis position (otherwise assigned C)
B branding axis position
W axis position
V axis position
machine (M)
group (G)
tool (U)
diameter (D)

Selection
If an asterisk (*****) character is displayed when the work is selected

Second custom area
may include customisable data, as:

Level	L (0 to 8)
Origin	O (0 to 3)

OK / NOK
OK if work instruction is correct, NOK if incorrect.

IF status
if the "?" character occurs, no logical conditions have been inserted, otherwise the "ON" or "OFF" item are displayed, according to the result of the logical test (the work will be executed or not).

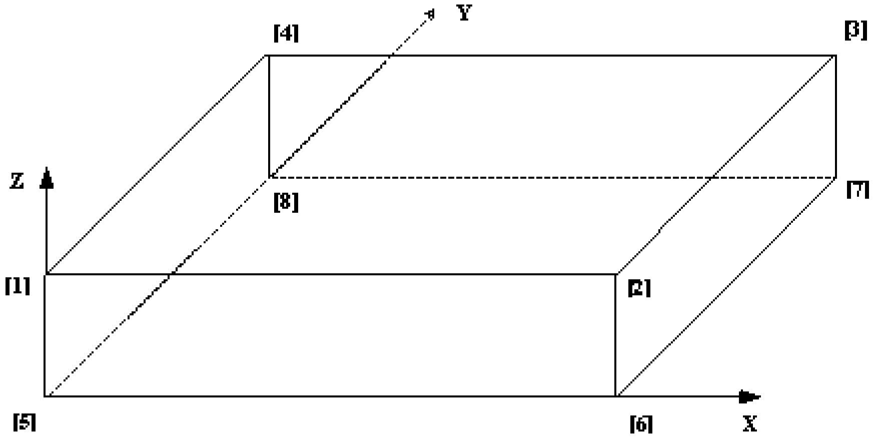
Sequence Number
it's the sequence number of the current instruction

Total Number
it's the total number of instructions programmed on the current face.

2 Piece

2.1 Piece Geometry

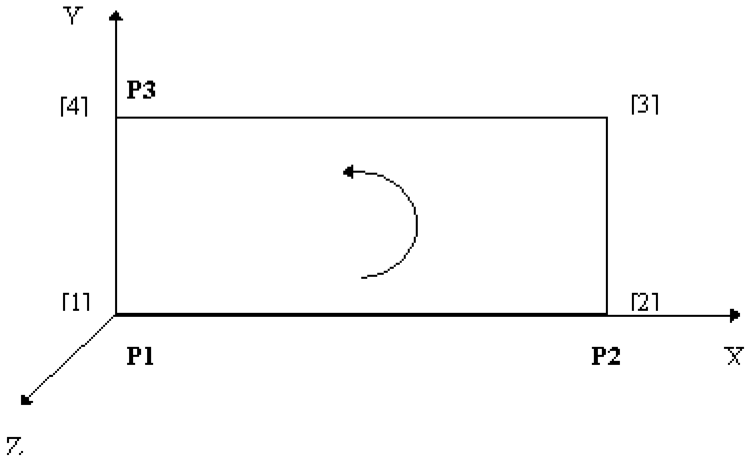
Edicad works on a 3D object, will be named "piece". Every piece may be defined by a list of a maximum of 99 faces: faces 1 to 6 are referred to as real, whereas the remaining faces are referred to as fictive.
The Origin of the Absolute Cartesian Reference System will be named, in the following, the *Zero piece*
The geometrically simplest piece is the parallelepiped, automatically created by Edicad when run in execution: the piece automatically assumes corners and sides according to the programmed dimensions: length, height and thickness. The faces comprising the parallelepiped are real faces. They are numbered according to convention, consequently, the top face is numbered as face 1, the opposite face is face 2, the remaining faces are numbered in a counterclockwise direction and in sequence, starting from the lowest face on the left, with respect to face 1. User may assign, to every face, a mnemonic name



The figure shows the faces of the basic piece, the edges and the reference system.

If the definition of the parallelepiped piece does not satisfy programming requirements fictive faces may be assigned to it. An example is shown in the creation of a cavity for the fitting of a glass panel in a door.

During the programming phase, the workings are referred to the reference system of the current face, which is displayed as shown below:



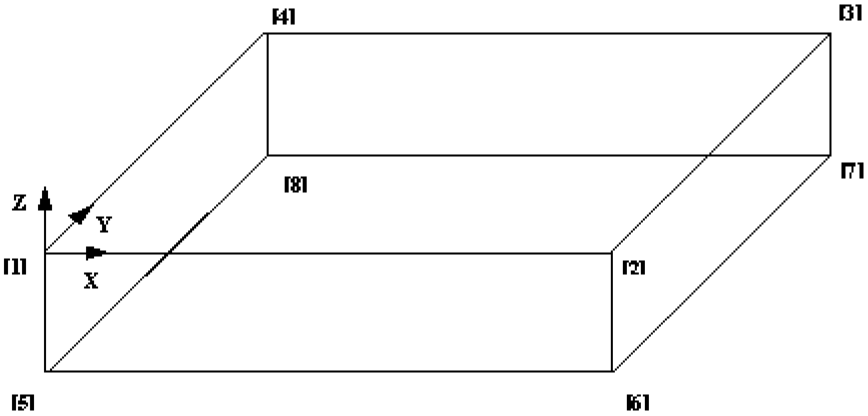
The figure shows a face as it is displayed during the programming phase.

2.2 General View

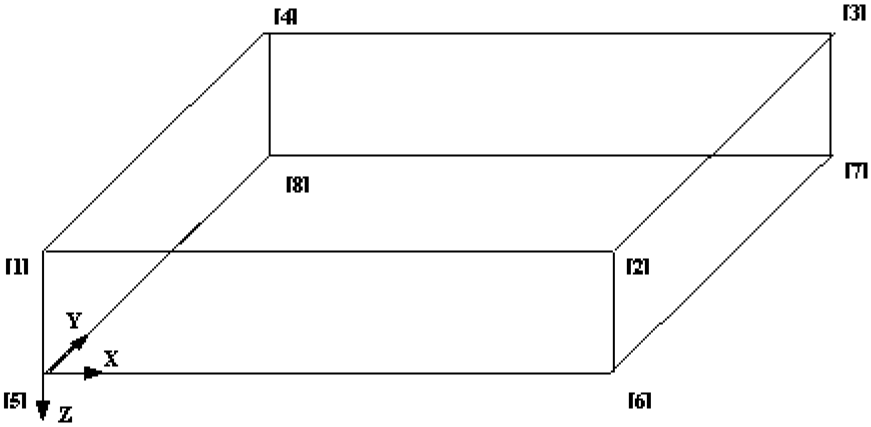
The General View window displays a three-dimensional view of the piece and the list of defined faces on the piece and for each of these the face number, the number of the workings present on the face and its mnemonic name (if it has one). The selected face appears in a different colour to all the others. In this phase all the general commands that could be carried out on the piece, such as for example assigning fictive faces and dimensions, can be activated.

2.3 Face View

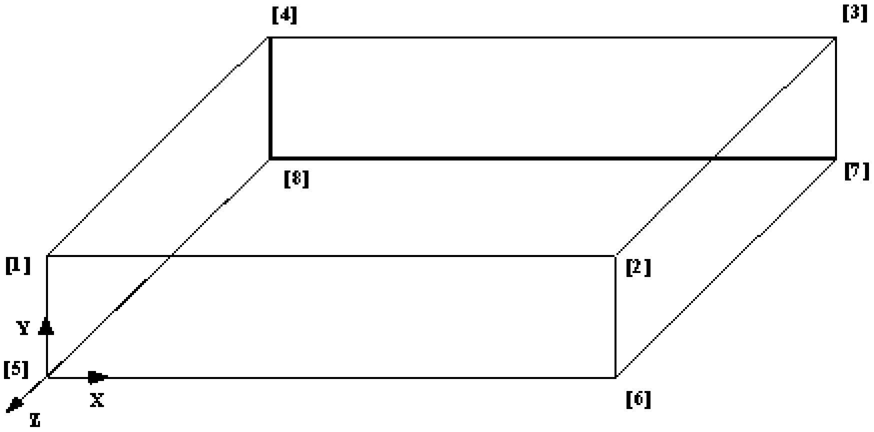
The Face View window displays a view of the face, as a plane, selected in the General View window. One Face View window opens for each selected face. The workings to be carried out are programmed according to the reference system of the face to which they are applied.
The local reference systems of each of the faces that comprise the parallelepiped piece are described below:



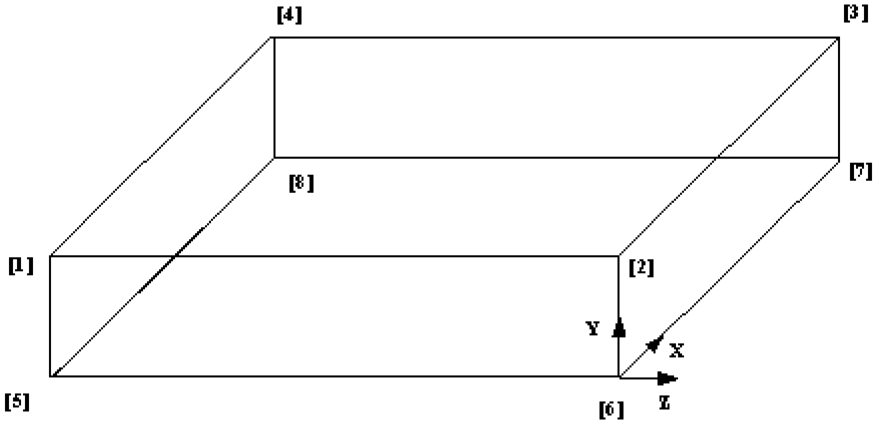
Face **1** local reference system



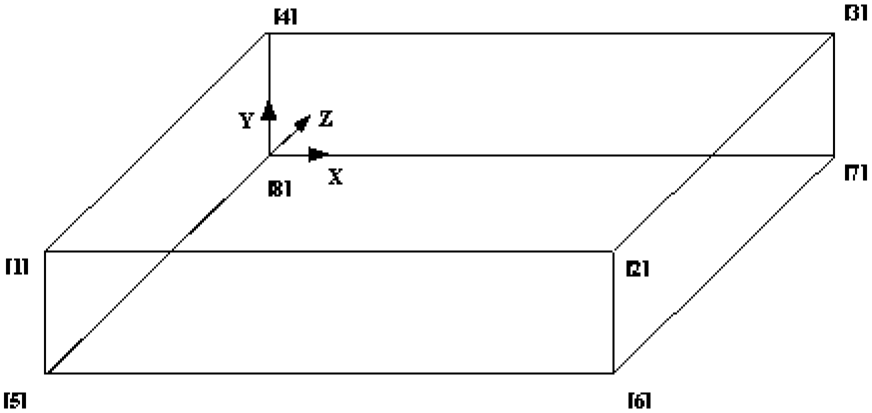
Face **2** local reference system



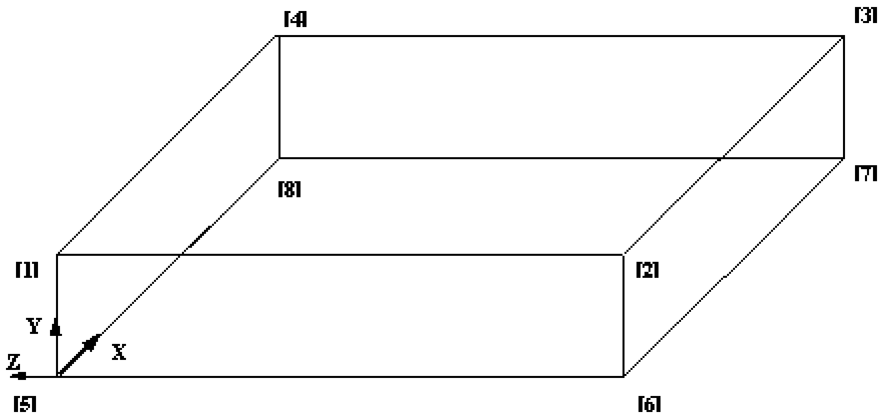
Face **3** local reference system



Face **4** local reference system



Face **5** local reference system



Face **6** local reference system

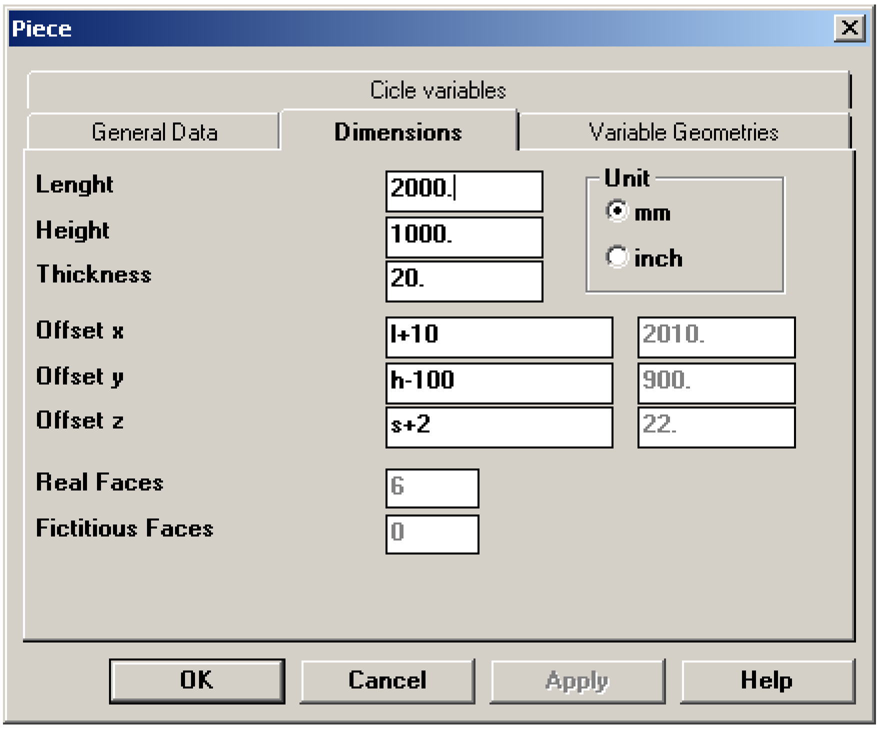
From the figures we can deduce that:

- a right-handed coordinate system is assumed for faces 1, 3, 4 .
- faces 6, 5, 2 are assigned in transparency, that is the reference systems of their opposing faces have the X,Y axes direction coinciding and the Z axis in the opposite direction (2->1; 5->3; 6->4). A left-handed coordinate system is assumed for faces 2,5,6.
- the Z axis is always positive at piece outlet.

2.4 Dimensions

Upon opening a new program the dimensions of the last piece that was saved and the default characteristics (offset and cycle variables) set by the manufacturer are assigned. If the piece to be edited has different characteristics they must be set through the ***Edit->Dimensions*** menu.
The dialog box displays several pages containing data. A detailed description follows:

Piece Dimensions



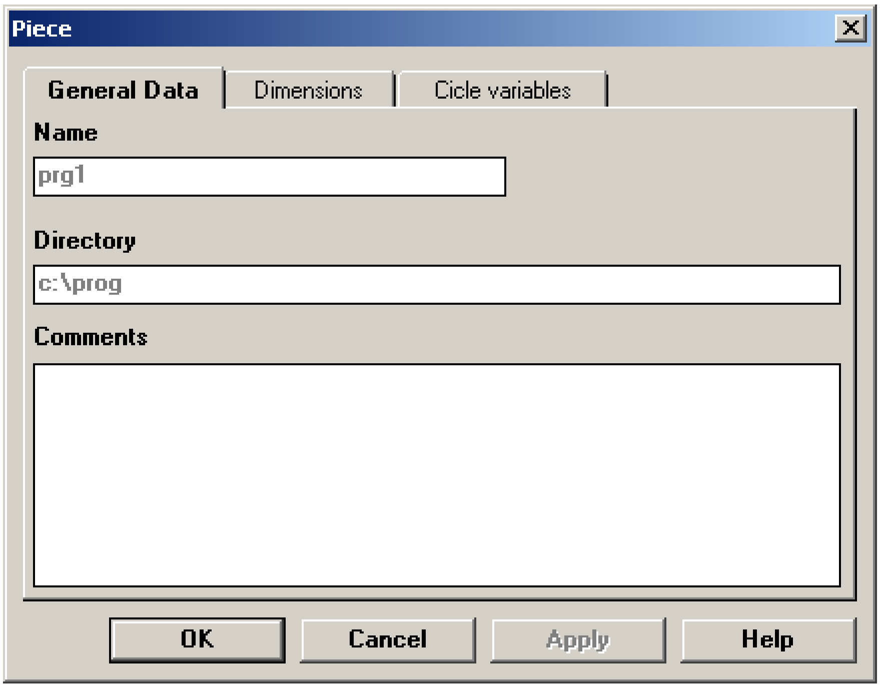
Lenght	2000.	
Height	1000.	
Thickness	20.	
Offset x	l+10	2010.
Offset y	h-100	900.
Offset z	s+2	22.
Real Faces	6	
Fictitious Faces	0	

Unit
● mm
○ inch

The following information can be defined for a piece:

- **Unit** of measurement in which all the following values are programmed
- **Length**, **Height** and **Thickness** are the dimensions of the piece.
- **Offset X, Y, Z** are the piece offsets which are added to a possible Execution offset in the execution phase. The offsets can also be set as numeric parameters (e.g.: 100+1) or according to the piece dimensions (e.g.: l-100). The operators "+ - * /" may be used. The dimension cannot be expressed as a denominator. The calculated value of a parametric expression may be displayed next to the data entry box.
- **Real faces:** number of the real faces of the piece (only displayed).
- **Fictitious Faces:** number of the fictive faces of the piece (only displayed).

General Data



Name: prg1

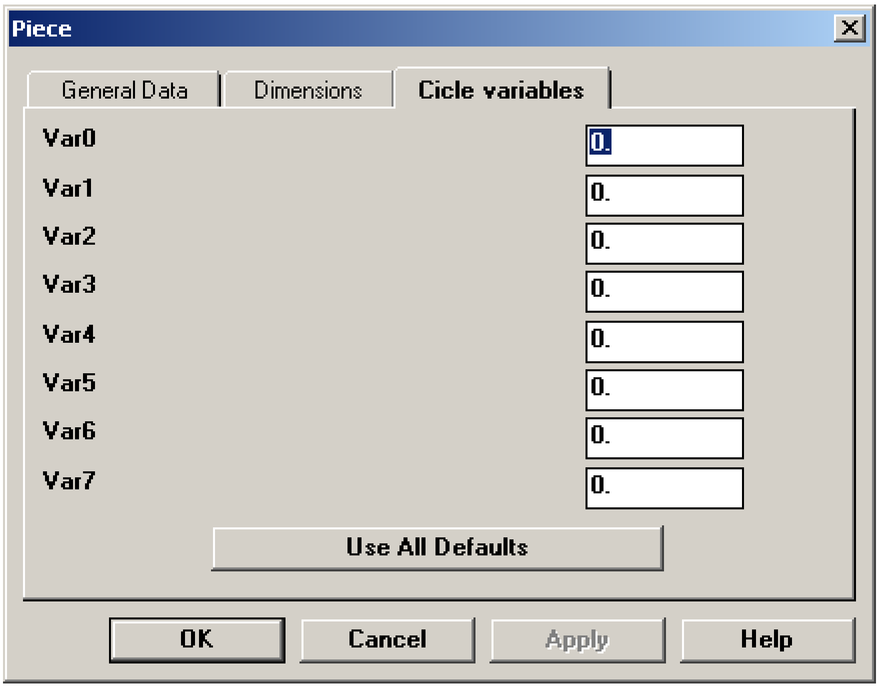
Directory: c:\prog

Comments:

Consisting of:

- ***Name* of the piece**, which is only displayed if the program has already been stored (only displayed).
- **Directory:** is the complete path for program saving; if it is a new program, the field is empty (only displayed).
- **Comments** on the program. A maximum number of 400 characters may be used for the comment.

Cycle Variables



Var0 0.
Var1 0.
Var2 0.
Var3 0.
Var4 0.
Var5 0.
Var6 0.
Var7 0.

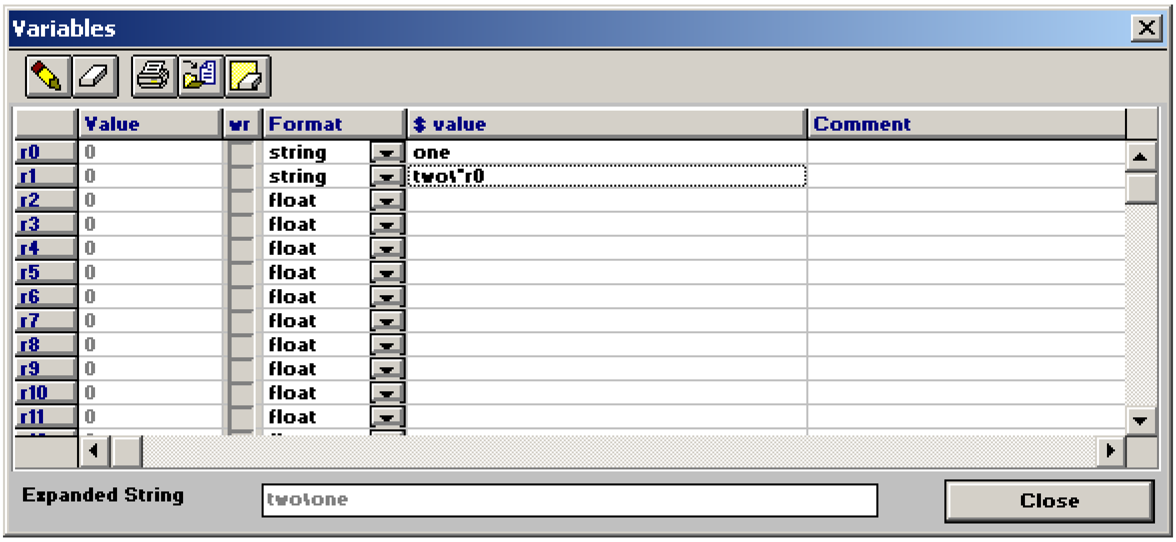
Use of the cycle variables is specific for each machine. These are values which become available during execution and which are not used during the piece editing phase. In the Edicad configuration phase the manufacturer can assign customized items, as shown in the example in the figure by the items "Execution" and "Load Flag ".
The values predefined by the manufacturer can be automatically set by means of the **[Use All defaults]** button.

Variable Geometries

By selecting an item other than **None** in the **Shaped piece** field a specific function of Edicad in some machines is enabled. Contact TPA technicians for further details.

2.5 Variables

A variable is a non-fixed value which may be assigned to a parameter of a machining operation. They must be set at the beginning of the program and they can only be redefined by an external program. They can be recalled to the program, to the parameterized formulas of the subsequent machining operations or used when running the program. They are used in the event of a Parametric Programming of the machinings. See the chapter on Parametric Programming for a more detailed explanation.



	Value	wr	Format	$ value	Comment
r0	0		string	one	
r1	0		string	two\"r0	
r2	0		float		
r3	0		float		
r4	0		float		
r5	0		float		
r6	0		float		
r7	0		float		
r8	0		float		
r9	0		float		
r10	0		float		
r11	0		float		

Expanded String: two\one

The dialogue window may be divided into two areas:

Variables

This is the area for editing the variables. The columns displayed are in the following order:

- variable name column (r0 ÷ r299).
- **Value** is the value calculated for a numeric variable if a parametric or numeric mathematical formula is assigned. (for example: 5+10 ...15 appears)
- **wr** if enabled this allows the user to re-assign new values to the variable while the program is running .
- **Format** is the variable format type. It can be either float, whole or string. For example a "name" value cannot be assigned to a Float format variable.
- **$ value** is the value of the variable. It may be assigned by means of parametric or numeric formulas. For example (10+5 or l+10).
- **Comment** is a comment on the variable. The maximum length is 40 characters.
- **Expanded string** is a field which shows the value of a string type variable when it is obtained from other variables.

Toolbar

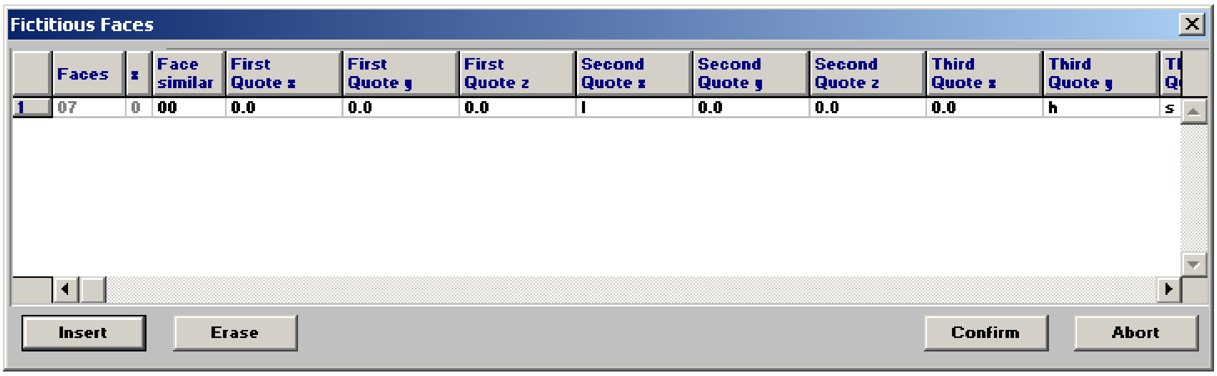
The Toolbar commands are only active if variables have been assigned. The icons for the commands are in the following order:

- **Set the variable:** confirms the settings entered for the current variable.
- **Reset the variable:** deletes the settings for the current variable.
- **Print the variable:** prints all the variables with the relevant settings.
- **Import the variable from another program:** imports all the variable settings from another program; the settings entered up to this point are lost as they are replaced by the imported ones.
- **Reset all the variables:** deletes the settings of all the variables.

2.6 Fictive Faces

A fictive face is a face that does not belong to the parallelepiped piece. For pieces with complex shapes, such as for example the cavity for the glass in a door, the definition of new faces must be set. The fictive faces are numbered progressively from 7 to 99. A description may be assigned by selecting the item in the ***Edit->Fictitious Faces*** menu.

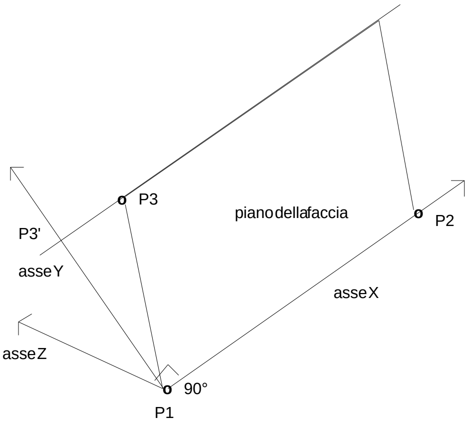
The window allows generating or modifying of these new faces.



Fictitious Faces

	Faces	x	Face similar	First Quote x	First Quote y	First Quote z	Second Quote x	Second Quote y	Second Quote z	Third Quote x	Third Quote y	Th Qu
1	07	0	00	0.0	0.0	0.0	l	0.0	0.0	0.0	h	s

The following figure shows how to set the triad of points required to define a fictive face.



Point P1 is the origin of the Cartesian coordinate system of the face. The line P1-P2 identifies the X axis of the face.
The line which is perpendicular to the X axis (on the plane of the face) identifies the Y axis: with P1 as zero and the orientation determined by the position of P3 (X axis rotates towards P3 with a smaller angle of rotation). If P1-P2 and P1-P3 are perpendicular to each other: the Y axis passes through P3. The line through P1, which is perpendicular to the plane of the face, identifies the Z axis of the face; the default orientation of the Z axis assigns a Cartesian coordinate system (right-handed).

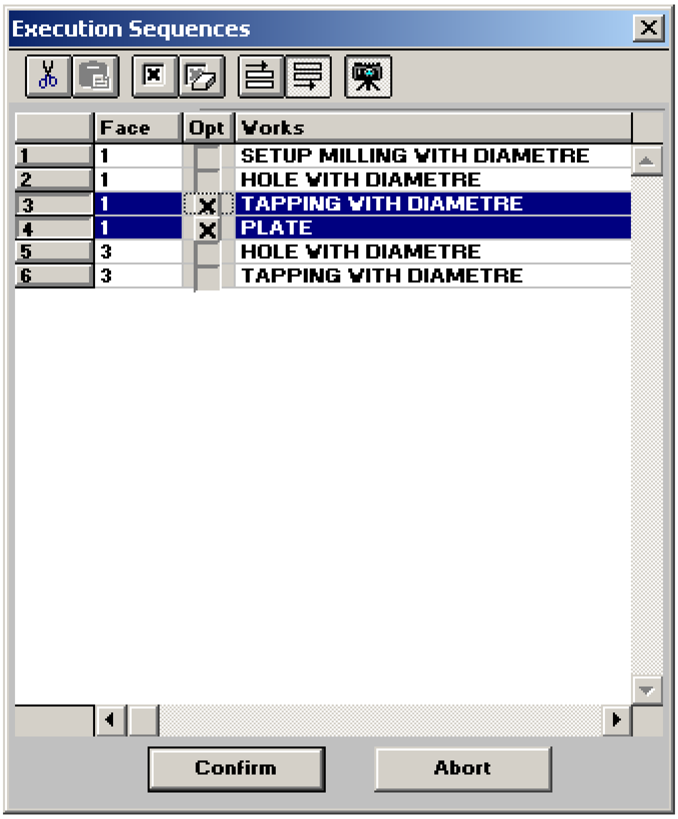
If the coordinate system of the set axes has an identical orientation to that of one of the parallelepiped faces then the latter is referred to as being a similar face to the fictive face. Consequently, the tools that are declared on the base face can also be used on the fictive face.

A fictive face with programmed machinings (except for subroutine -induced calls) cannot be eliminated. When a fictive face is deleted from the list of faces, the numbering of the remaining faces is modified to avoid gaps.

2.7 Sequences

The window for assigning machining sequences is enabled from the ***Apply->Execution sequences*** menu in General View.

This function is linked to the optimization settings and can be enabled or disabled for each individual piece. Normally it is activated by the manufacturer of the machine.
The user can determine execution sequences for the machining operations.



	Face	Opt	Works
1	1	□	SETUP MILLING VITH DIAMETRE
2	1	□	HOLE VITH DIAMETRE
3	1	☒	TAPPING VITH DIAMETRE
4	1	☒	PLATE
5	3	□	HOLE VITH DIAMETRE
6	3	□	TAPPING VITH DIAMETRE

The window displays a list of the machining operations which are present on all the faces of the Piece. The columns have the following meanings:

1. progressive number of the machining operation

2. **[Face]** number of the face to which the machining operation belongs

3. **[Opt]** if enabled, indicates that the optimization criteria must be applied to the machining operation (e.g. tool match and order)

4. **[Works]** description of the machining operation

To change the run order of a machining operation simply click on its progressive number.
To make changes to the machining list use the icons described below which are set out in the following order on the Toolbar:

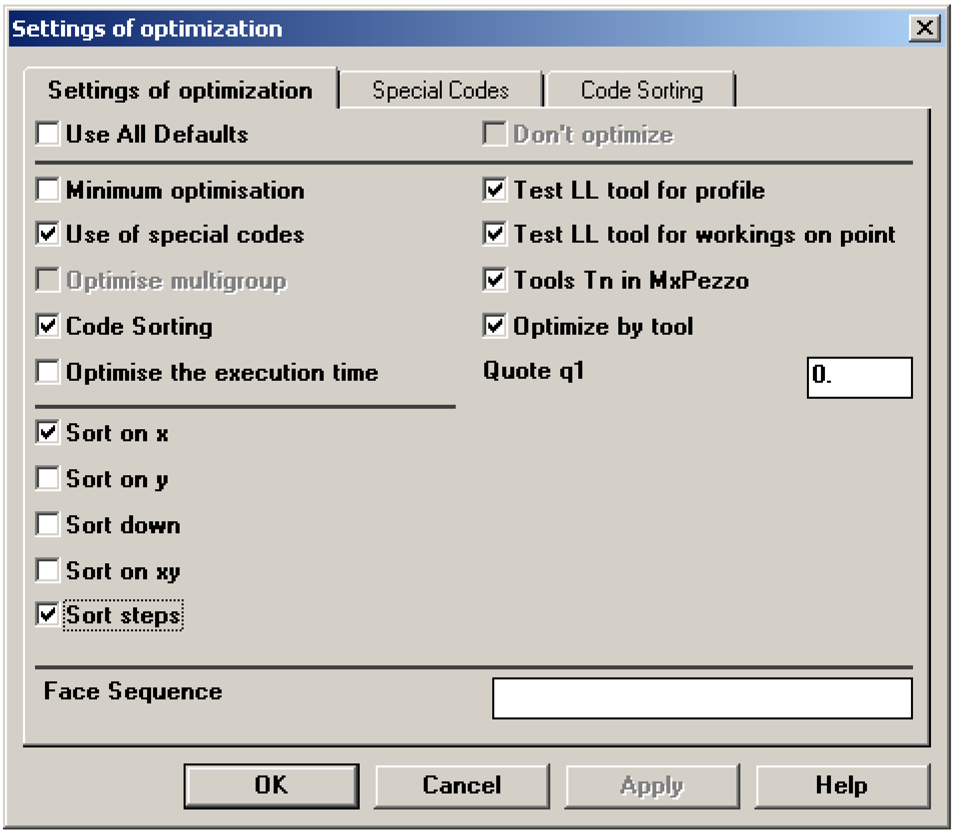
- **Remove selected blocks:** temporarily deletes the selected machining operations from the list so that they may be entered in another position. This command <u>does not eliminate</u> the machining operations.
- **Insert removed blocks:** inserts the machining operations, which were previously deleted, in the chosen position.
- **Activate optimization for selected blocks:** enables the use of optimization criteria for the selected machining operations.
- **Deactivate optimization for selected blocks:** disables the use of optimization criteria for the selected machining operations.
- **Insert block before:** the deleted machining operations are inserted before the selected operation.
- **Insert block after:** the deleted machining operations are inserted after the selected operation.
- **Activate deactivate view of face:** if enabled for the selection of a machining operation the corresponding operation on the face is also selected.

2.8 Optimization settings

The window for assigning the specific optimization settings for the individual piece is enabled from the ***Edit->Optimization Settings*** menu in General View.

The **Use All Defaults** command appears on each page in the dialog box. If selected, the settings defined by the Manufacturer in the Edicad configuration phase are used, otherwise the run settings can be modified according to the requirements of each individual piece.
The details of each page are given below:

Optimization Settings



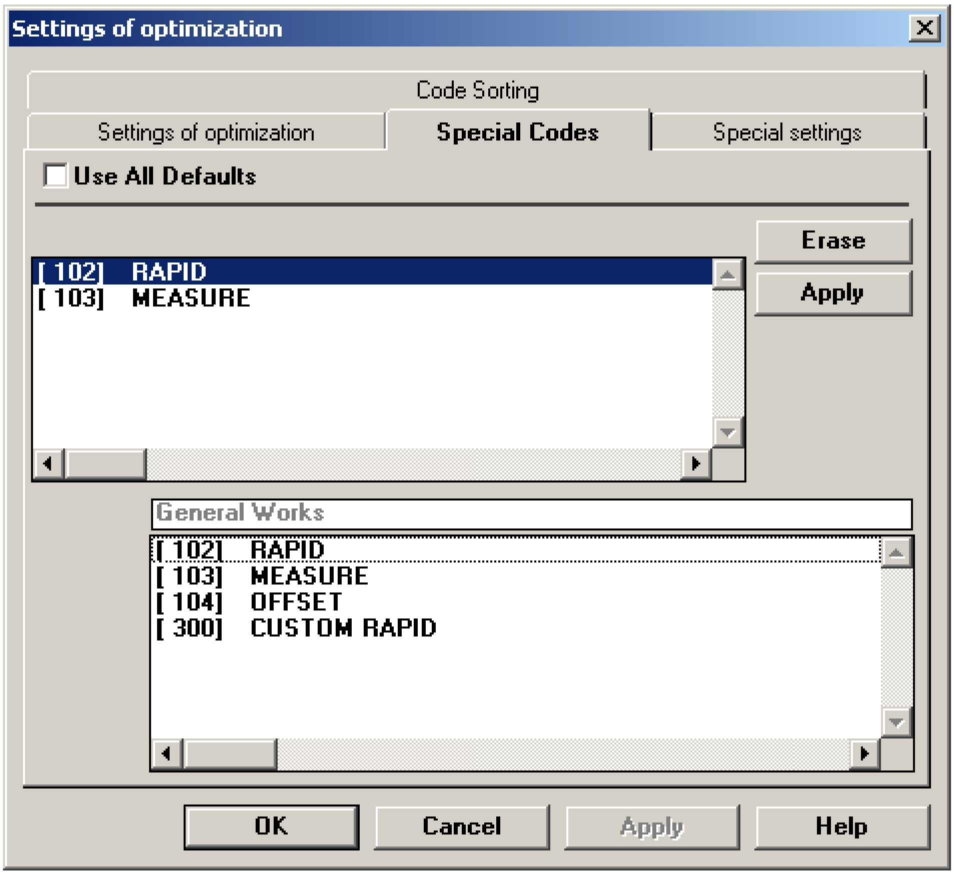
☐ Use All Defaults
☐ Don't optimize
☐ Minimum optimisation
☑ Use of special codes
☐ Optimise multigroup
☑ Code Sorting
☐ Optimise the execution time
☑ Test LL tool for profile
☑ Test LL tool for workings on point
☑ Tools Tn in MxPezzo
☑ Optimize by tool
Quote q1: 0.
☑ Sort on x
☐ Sort on y
☐ Sort down
☐ Sort on xy
☑ Sort steps
Face Sequence:

- **Use All defaults:** if selected, excludes all the other possibilities as it assumes the general criteria and data.
- **Minimum optimization:** if selected, carries out a check of all the machining operations programmed by tool and then the selection of tools for the operations programmed by diameter, without activating pattern match procedures. Furthermore, it does not sort the items.
- **Use of special codes:** if selected, checks the existence in the program of any codes defined in the item "Special Codes". If some exist, a check is made on the parameters of the tool in the machining operations programmed by tool, then on the selection of the tools for the operations programmed by diameter, without activating pattern match procedures. Furthermore, it does not sort the items.
- **Code Sorting:** if selected, enables the sorting procedure by code. This criterion is only used on single-unit machines.
- **Optimize the execution time:** if selected, optimizes the program run time, otherwise the optimization time is given priority.
- **Test LL tool for profile:** if selected, checks that the depth programmed on the setup machining operation is less than or equal to the length of the tool.
- **Test LL tool for workings on points:** if selected, checks that the depth programmed on the point machining operation is less than or equal to the length of the tool.
- **Tools Tn in MxPezzo:** enables the insertion of the multiple miller setup in the piece matrix. The inserted tools (max 4) are indicated in piece matrix even in the execution run sequence.
- **Optimize by tool:** enables the optimization by tool. It groups together the setup machining tools (cutters and blades) with the same number. This procedure takes into account the enabling of the sort by code and the presence of special codes. The management of these takes precedence over

an optimization by tool. If a custom optimization run is enabled, which activates all the controls on the validity of the tools and if necessary sorts them, the option is automatically disabled during the optimization phase.

- **Quote q1:** on single-unit machines the assigned value defines a dividing height for field Y. In the first zone the machining operations are sorted by ascending X values, in the second zone they are sorted by descending X values. On multi-unit machines it is the limit value below which the machining operations are only carried out by the low units.
- **Sort on x:** if selected, sorts the machining operations by descending or ascending X values according to whether the item Descending Sort is enabled or disabled. At a parity of X values, it sorts by ascending Y values. This criterion is dominant over the "Sorting by y" criterion It is only used on single-unit machines.
- **Sort on y:** if selected, sorts the machining operations by descending or ascending Y values according to whether the item Descending Sort is enabled or not. This parameter is only used on single-unit machines.
- **Sort down:** if selected, sorts the machining operations by descending Y or X values. This parameter is only used on single-unit machines.
- **Sort on xy:** if selected, sorts the point machinings on the panel according to the following criterion: sorting by ascending x and from machining operations with the same abscissa select the one with the ordinate which is closest to the ordinate of the previous point. If following a point machining, a series of Y point machinings are programmed, only the two end machinings are considered so that they compared to the ordinate of the previous point. All the others are then sorted by ascending or descending ordinate according to the chosen machining operation. This criterion is only used on single-unit machines.
- **Sort steps:** if selected, sorts the machine operations by ascending X before carrying out any optimization operation. This flag enables a sorting within the optimizer which does not re-occur when executing the machinings in the machine.
- **Face Sequence** sets the run sequence of the faces on the piece.

Special Codes



□ Use All Defaults

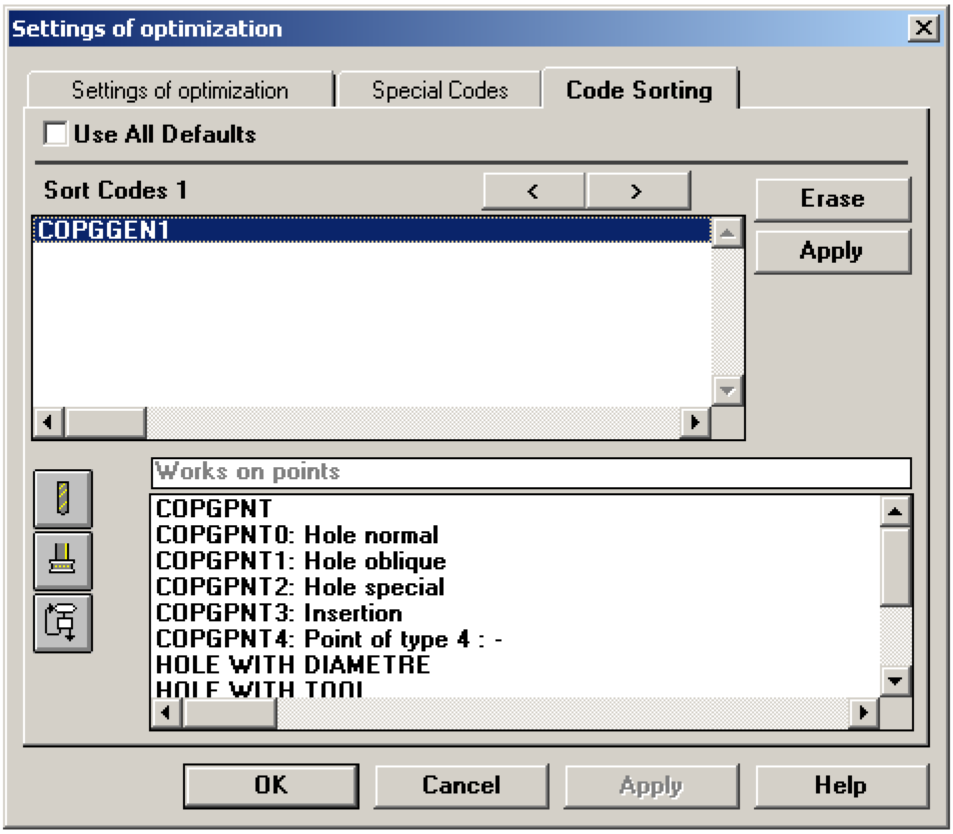
[102] RAPID
[103] MEASURE

General Works

[102] RAPID
[103] MEASURE
[104] OFFSET
[300] CUSTOM RAPID

- **Use default settings:** if selected, the general optimization settings are assumed.
- **Special codes:** list of 'engen1' type codes which, if defined in the program, enable the use of the **Special Code Management** criterion. The maximum number of codes which may be defined is 10. The list of codes to be sorted, which is pre-set by the Manufacturer, appears. Any codes not involved in the optimization procedure may be eliminated from the list by pressing the **[Cancel]** button. New codes can be added to the list by means of the list which appears below and by confirming with the **[Apply]** button.

Code sorting



☐ Use All Defaults

COPGGEN1

COPGPNT
COPGPNT0: Hole normal
COPGPNT1: Hole oblique
COPGPNT2: Hole special
COPGPNT3: Insertion
COPGPNT4: Point of type 4 : -
HOLE WITH DIAMETRE
HOLE WITH TOOL

- **Use All defaults:** if selected, the general optimization settings are assumed.
- **Sort Codes:** a list of codes, which is pre-set by the Manufacturer, appears to be sorted. Any codes not involved in the optimization procedure may be eliminated from the list by pressing the **[Erase]** button. New codes can be added to the list by means of the list which appears below, by using the three icons which correspond to the point, setup and special machining operations and by confirming with the **[Apply]** button. The generic code for the machining operation may be assigned as the sorting code. The default ensures that all the engen1 operations are sorted first, then the setup operations and lastly all the point machining operations.

Special Settings

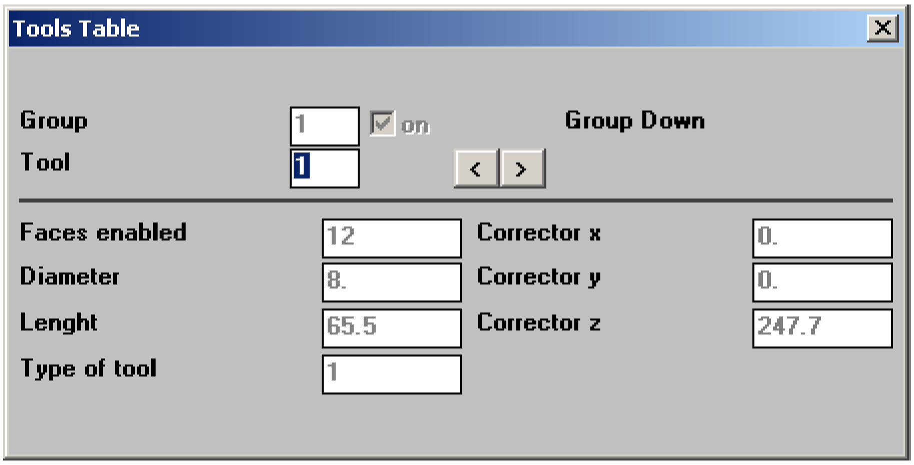
- **Use default settings:** if selected, the general optimization settings are assumed.
- **Assign sequences:** if selected, enables the assignment of the machining operation run sequences.

Hardware Exclusions

- **Hardware exclusions:** a list appears of the enabled machines and the units belonging to each of them. The machines and/or units may be disabled so that the optimizer does not use the tools belonging to the excluded units to optimize a piece.

2.9 Tool Table

Edicad displays the parametrics written in the standard TPA format. The tool table can be viewed by selecting the item in the ***Window->Tool Table*** menu, or by means of the **[CTRL+T]** or **[ALT+T]** button combination.



Tools Table

Group 1 ☑ on

Tool 1

Faces enabled 12

Diameter 8.

Lenght 65.5

Type of tool 1

Corrector x 0.

Corrector y 0.

Corrector z 247.7

The general data relevant to each tool are displayed (they cannot be modified!) in this window. These data include:

- **Machine** is the number of the Machine to which the tool belongs. It only appears if the plant consists of more than one machine.
- **Unit** is the number of the Unit of the tool table that you want to view. If the 'on' box beside it is active then the unit is physically present.
- **Tool** is the number of the tool belonging to the Unit. The two buttons next to it allow you to scroll the tool list and their details appear in the area below.
- **Work faces** on which the tool can work; if it contains '123456' then it is a universal tool.
- **Diameter** of the tool.
- **Length** of the tool.
- **Type of tool.**
- **x, y, z correctors** of the tool.

If the drilling head has more than one drill, the **[Multidrill Parameters]** button displays the following data :

- **x/y symmetry offset** is the distance between the head programming point and its centre.
- **Drill number** consists of 4 columns (#1-#4), which refer to the other 4 drills that may be part of the tool, in each of which the following parameters are displayed:
  - **Drill diameter** if equal to zero means that the drill is not fitted.
  - **x corrector** is the distance on the X axis of the drill from the centre of the head.
  - **y corrector** is the distance on the Y axis of the drill from the centre of the head.

2.10 Reprocessing

At any time all the programming of the Piece can be reprocessed by means of the ***Apply->Reprocess All*** command, whereas select the ***Apply->Reprocess Face*** command to reprocess the current face.

All the error or warning sequences are regenerated if any error conditions or possibility of uncertainties are present when the controls are made on all the parameters of the piece by the central processing unit.

3 Workings

3.1 General properties

3.1.1 Typologies

This chapter describes the commands used to insert and change machining operations and the commands which are linked to these functions.
Note that the last machining operation entered becomes the current one, therefore the next operation will be placed immediately above or below this one, depending on the insert mode that is active at the time, and then the new one becomes the current machining operation.
Machining operation refers to the basic block which can be programmed on the face.

Workings may be classed in the following typologies:

On point include all the workings may geometrically characterized by only the application point, as in the case, for instance, of the Drilling or Fitting instructions

Setup include all the profile initializing instructions: a profile is intended as a sequence of consecutive arcs or segments (see Arc and Linear workings). Also a Setup working is characterized by its application point.

Arc include all types of workings in circular interpolation, with the different specifications of parameters and geometric conditions

Linear include all the different types of workings in linear interpolation

Logical include all the workings defining a logical conditions testing (IF, ENDIF, ...) or a program variables assigning.

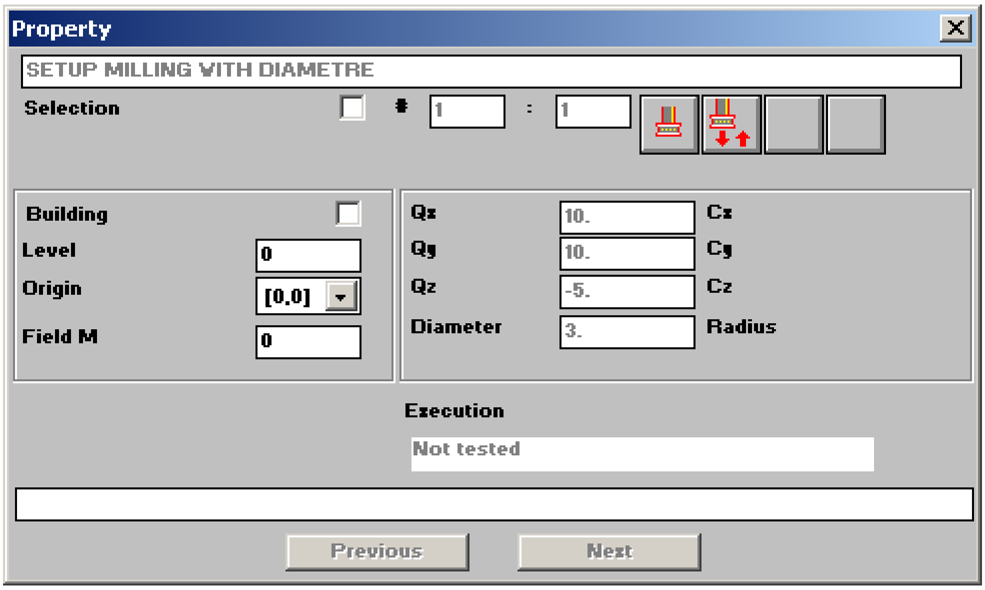
Macro include a most complex workings, normally customized according the specific Application.

Subroutines programmed by the User as an elementary workings synthesis, are used conveniently to represent a recurrent working layouts, as to be stored independently and recalled and inserted in main programs, when required.

3.1.2 Properties

The main properties of the current block can be viewed from the ***Edit->Property ...*** menu.

The window that appears displays the properties of the current machining operation. Several major properties may be modified and, furthermore, from this window you can go to the properties of the previous or next block, which allows you to easily check the properties of the nearby blocks and obviously modify them.



Property

SETUP MILLING WITH DIAMETRE

Selection ☐ # 1 : 1

Building ☐

Level 0

Origin [0,0]

Field M 0

Qx 10.

Qy 10.

Qz -5.

Diameter 3.

Cx

Cy

Cz

Radius

Execution

Not tested

This window includes:

- **Operating code and Work Name**.
- **Selection** indicates the selection status of the current block and allows you to select or unselect it.
- **Progressive number** of the current block (box marked with "#").
- **Total number of blocks** which make up the program (box marked with ":").
- **One or more icons** which provide a visual indication of the current machining type (point, setup, arc, line, logical, macro or subroutine).
- **Building** to which the current machining is matched, can be activated or deactivated by means of the control box next to it. If indicated as building, the machining is compiled but not executed.
- **Level** to which the current working is linked, can be modified by activating the control box next to it and entering the new level. It can only be modified on the profile for the Setup machining operation. This field is not always available.
- **Origin** of the current machining operation, can be modified by activating the control box next to it and entering the new origin. It can only be modified on the profile for the Setup machining operation. This field is not always available.
- **M Field**
- **Qx, Qy, Qz** are coordinates of the point of application of the machining operation (Qx, Qy, Qz).
- **Cx, Cy, Cz** are coordinates of the centre (Cx, Cy, Cz). They are only important for a circular interpolation.
- **Radius** is the value of the radius of the arc. It is only important for a circular interpolation.
- **Diameter** indicates the tool diameter, only for a point or setup machining operation.
- **Execution** shows the current logical condition status of the machining operation:
  - *Not tested* if the logical conditions are not applied
  - *Verified as active* if the condition to be verified is ON
  - *Verified as non-active* if the condition to be verified is OFF
- **Message Area** displayed in the following cases:
  - invalid code. May indicate the processing status (OK/NOK), specific status of the machining (induced)...
  - code excluded from the graphic figure (due to the effect of the visualization filters)

The **[Previous]** and **[Next]** buttons allow you to go to the properties of the previous machining operation or the next one which becomes the current operation.

3.1.3 Geometric Information

All the geometric information of the current block may be displayed with the command: **Edit ->Geometric Information**

The window that appears shows the geometric information of the current work. It is possible to switch to the information associated to the previous or next block by means of **[Previous]** and **[Next]** buttons.
The data given in the dialog box vary according to the type of machining referred to in the displayed information. These include:

- ***Operating code and Work Name***.
- ***Number of program line*** corresponding to working
- ***Total number of program lines***
- ***Working typology*** *and* ***main geometrical data*** represented in graphic form by buttons.
- ***Segment of profile****:* when enabled, it allows examining, for the expanded segments of profile, geometric data of each segment by means of buttons **[<]** and **[>]**.
- ***Basic geometrical data* as:** application point (Qx, Qy, Qz), tool diameter.
- ***Rotation, centre, radius*** in case of circular interpolations*.*
- ***Computed geometrical data*** (for profile segments): tangent to segment in the initial and in the final point, coordinates of intermediate point and tangent in this point (tangents on xy plane and expressed in degrees).
- ***Development of macros or subprograms*** the initial and the final point of working are indicated.
- ***Correct Elaboration*** box indicates the state of the elaboration.
- **Messages** area, displayed in the following cases:
  - invalid code
  - code without graphic representation (because of visualisation filters)

3.2 Introduction

3.2.1 Introduction

To insert a new machining operation on a face, go to the *Workings Palette*, and select the button which corresponds to the desired typology and then the machining selected from the operations presented.
At this point the dialogue box for data setting appears.
The palette varies from machine to machine depending on the user's requirements.
The machining typologies which correspond to the basic configuration of *Edicad* are listed below; the *Workings Palette*, however, only contains the operations that the machine is capable of executing. In this type of palette, when the milling operations are activated, a second palette appears which gives access to the various types of milling operations.

DRILLING

MILLING

Setup
Single lines
Chamfering
Single arcs
Fillets
Double arcs
Circles

POLYGONS

SAWING

INSERTIONS

FOR WOOD

SUBROUTINES

CODES CNC90

LOGICS

SPECIAL

3.2.2 Dialog box

The parameters which characterise the machining operation, whether they are geometrical or technological, are set through a *Dialog box*, (two if the technological ones are separate from the geometrical ones), which is different for each machining operation.
Normally, to make things simpler, the initial request is only for the geometrical parameters (and possibly the tool number) and the implicit assumption of the preset technological parameters (work speed, rotation speed, etc.): however, it is possible to recall a second window in order to set the technological parameters through a dedicated button.

The *Dialog box* may be provided with numeric fields - to be set in indirect or parameterized form, by means of the variables and mathematical operators described in the next chapter - or logic fields for enabling/disabling, which may be selected by simply clicking the mouse.

To recall the second *Dialog box* for setting Technological parameters, when permitted, select the [Technology] button.

Normally only the one **[Geometry]** button is present in the *Dialog box* for the *technological parameters*, to return to the first geometry *Dialog box* .

3.3 Edit

3.3.1 Assign technology

This command allows assigning geometrical blocks with codes related also to technology. Select the **Edi*t->Ass*ign technology** menu item.

Window *Assign technology* appears. It contains:

- **Blocks area:** for selection of workings involved in assignment:
  - *current*: if only the current block
  - *selection:* all the selected blocks
  - *face:* all blocks in the current face.
- **Code for geometric points** area, indicating the technological code selected for assigning to geometrical points.

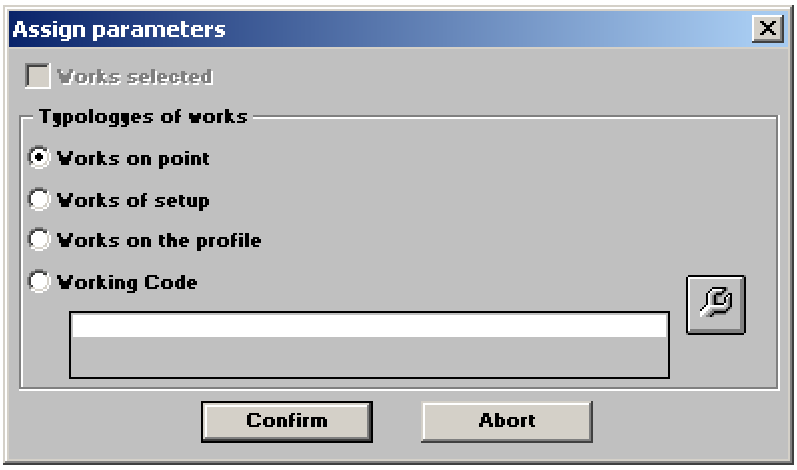
- **Code for geometric setup** area, indicating the technological code selected for assigning to geometrical setups.

- **Assign technological setups for geometric points :** all the geometric points are assigned a setup code.

For the geometric points or setup code choice select the bitmap for the corresponding machining. A dialog box opens with a list of the machining operations which are present and selectable: the machining code will be entered in the corresponding code zone.

3.3.2 Assign parameters

This command allows global assignment of parameters of homogeneous workings. Select the **Edit -> Assign parameters** item.
The *Overall Assignments* window appears. It contains:



Assign parameters

☐ Works selected

Typologyes of works

● Works on point

○ Works of setup

○ Works on the profile

○ Working Code

- **Works selected:** if selected acts on the selected machining operations.
- **Works on point:** the assignment is applied to point machining operations.
- **Works of setup:** the assignment is applied to setup machining operations.
- **Works on the profile:** the assignment is applied to profile machining operations.
- **Working Code:** if selected, code must be introduced in the box beside.

In case of Confirm a window is shown for the introduction of parameters to be assigned. For the assignment the following rules must be observed:

- unassigned fields are not modified
- it is possible to perform numeric or parametric assignments, as for a normal work assignment. Assignment of a parameter may be cancelled by introducing "=".
- in case of assignment by code, non numerical parameters (buttons, lists) are not allowed.
- in case of assignment of a subprogram, variables 'r' are not re-assignable.

When the button is pressed the **Workings Palette** allows you to select the working to which the overall assignments are to be made and the **Working selection bar**.

Within the *Selection Bar* some buttons are available for direct assignments.
In case of selection by type of working, the corresponding buttons are enabled only if there are workings selected.
These are the available buttons:

Point workings

Setup workings

Profile workings

Working Code

3.3.3 Assign Reference Origin

The reference Origin of a work may be changed by the command ***Edit->*****·Assign reference point.**

The reference origin is not used for editing the workings in Edicad. This parameter is used during the execution of the piece in the machine and can be managed differently according to the application. As this is a "custom" parameter, contact the machine manufacturer for further details.

The *Reference Assigning* window is recalled, showing the following areas:

- The **Blocs** area allows to select the blocs to be assigned, between the options:
  - **actual** if only the current bloc.
  - **selections** if all the selected blocs.
  - **face** if all the face bloc.
- The **Reference Origin** allows to select between four origins for the axes reference:
  - **[0,0]** *Bottom* (on the bottom left corner).
  - **[0,^]** *Top* (on the top left corner).
  - **[>,0]** *Right* (on the bottom right corner).
  - **[>,^]** *Top Right* (on the top right corner).

3.3.4 Level Assigning

This command allow to change the level a working belongs to, selecting the **Edit-> Assign level** menu item.

The ***Level Assign*** window is recalled, showing the following areas:

- The **Blocs** area allows to select the blocs to be assigned, between the options:
  - *actual* if only the current bloc.
  - *selections* if all the selected blocs.
  - *face* if all the face bloc.
- In the **Level** box, the corresponding value may be entered (0 for no level or a number from 1 to 8).

3.3.5 M Field

This command allows you to edit the M field value to which the working belongs from the ***Edit-> Assign field M*** menu.

The M field is a code which is used during the execution of a piece in the machine and can be managed in different ways according to the application. As this is a "custom" parameter, contact the manufacturer for further details.

The *Assign M Field* window appears, containing:

- **Blocks**: allows you to select which blocks are to be involved in the assignment, that is:
  - **actual** if only the current block.
  - **selections** for all the selected blocks.
  - **face** for all the face blocks.
- **M field**: allows you to set the desired value ranging from 0 to 65000.

3.4 Selection and unselection

3.4.1 Editing operations on the active face

All the typical editing features are available in the text manipulation. The following commands may operate or on the current bloc or on a group of blocs, according to the selection: cut, copy, find and replace. These operations can be carried out on several blocks simultaneously (of course, they must firstly be selected). The commands are as follow.

3.4.2 Selecting and unselecting the blocks on which to operate

Either a single block or several blocks may be selected to work on simultaneously.
When a block is selected, the asterisk (*****) appears in the *Status Bar* and the working becomes a different colour.
To select a block, proceed as follows:

- Move the cursor to the desired working.
- Press the **[CTRL]** button and click the left button on the mouse; the cursor turns into a hand with the finger pointing upwards, and the working appears in the colour which indicates that it has been selected.

To unselect a block, carry out the same operation on a block that has already been selected.
To select several blocks (referred to as *selected blocks*), enclose the relevant blocks in a rectangle.
The procedure is as follows:

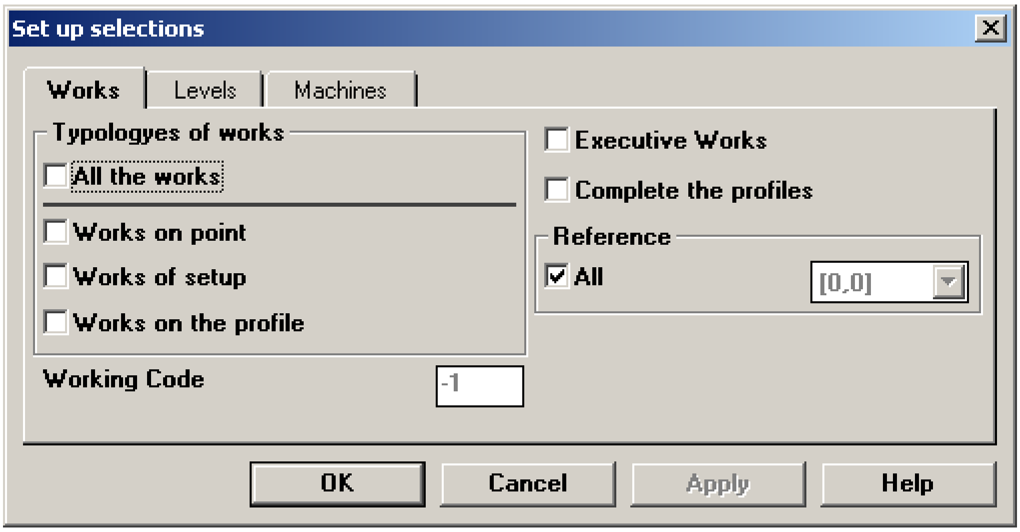
- Move the cursor to the point from which you want to construct the area.
- Press the **[SHIFT]** key and click the left button on the mouse; the cursor turns into a hand with the finger pointing upwards; both buttons must be held down until the operation is completed.
- Drag the mouse to construct the area.
- When the area is complete, release the button on the mouse to activate the selection.

To unselect several blocks simultaneously, simply use the ***Select->Unselect all*** command.
The selection is actuated by applying the filters and the settings assigned previously using the ***Set->Set selections*** command**.**
When dealing with a segment belonging to a profile the selection/unselection will involve the entire profile if the ***Tools->Apply profile*** functions are active.

3.4.3 Defining selection criteria

The workings are selected according to criteria that may be set as desired for each face of the piece; the unselection procedure also uses these criteria which are settable from the **Select->Selection of**
.
The dialog box offers several data pages which are described below in detail:

Workings



Set up selections

Typologies of works
□ All the works
□ Works on point
□ Works of setup
□ Works on the profile

Working Code -1

□ Executive Works
□ Complete the profiles

Reference
☑ All [0,0]

Typologies of works area:
All the works: by activating this box the selection will involve all the working types indicated in the **Working code** (if >0, otherwise all).
Works on point: only these works (Drilling, Fitting,..) will be interested
Works of setup: only these works (Setup) will be interested
Works on the profile: only the Arc and Segment instructions will be interested

Reference area:
All: if enabled, all the works may be selected, independently of their reference origin, otherwise the

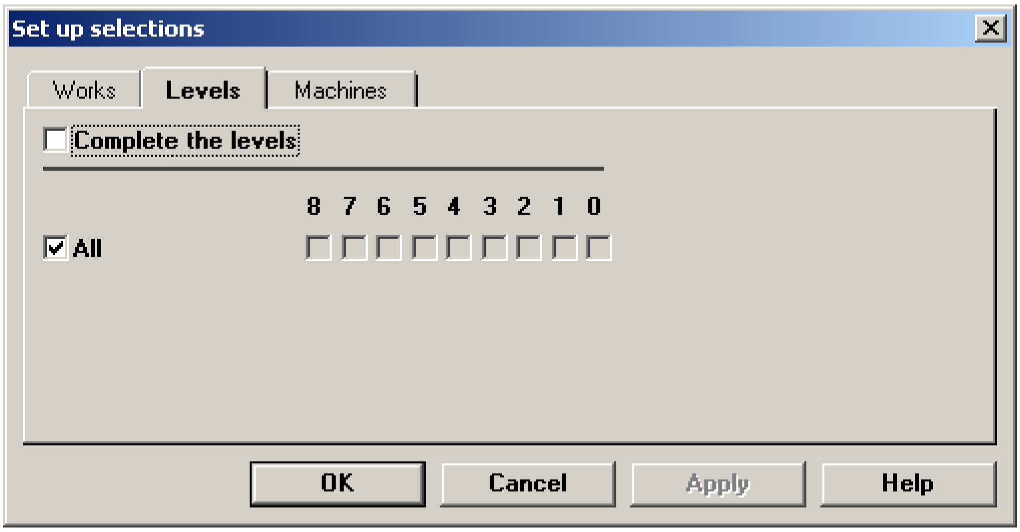
required origin must be entered:

[0,0]	Bottom (on the bottom left corner)
[0,^]	Top (on the top left corner)
[>,0]	Right (on the bottom right corner)
[>,^]	Top Right (on the top right corner)

Working Code: to enter the corresponding selection code: if the "-1" value is programmed, all work typologies will be interested
Executive Works: if enabled, all the logical instructions will be ignored for selection
Complete the profiles: if enabled, when some bloc of a profile has been selected, also all the other blocs will be automatically selected

N.B.: Several selection criteria may be used simultaneously.

Levels


Set up selections

□ Complete the levels

☑ All

8	7	6	5	4	3	2	1	0
□	□	□	□	□	□	□	□	□

Complete the levels: if the selection includes workings of the indicated level, check this box if you want other workings belonging to the same level, even outside the selected area, to be selected as well.
All: check this box if you want the selection to involve all the levels, or set the desired levels by checking the corresponding boxes.
Boxes numbered from '0' to '8': check the boxes corresponding to the levels relevant to the selection.

3.4.4 Selecting a working to be edited

The working to be edited must firstly become the current working; to do this proceed in one of the following ways:

- directly with the mouse;
- using the **[Cursor Up]** and **[Cursor Down]** keys to select the previous or next working and the **[Home]** and **[End]** keys to select the first or last working;
- using the commands on the Selection menu or the corresponding buttons on the Toolbar.

If using the mouse position the cursor on the working, or near it, then click with the left button, so that the working becomes the current one; then double click with the left button on the mouse to activate the dialog box containing the working data.

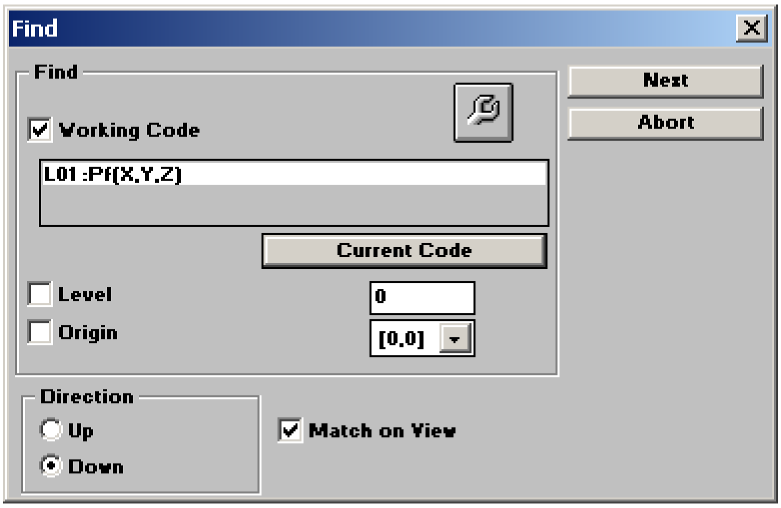
If using the buttons or the menu proceed as follows:

from the menu:

- **Select->Next corresponding block:** turns the next working, closest in terms of its x-y position, into the current one. The next command activated a search of the blocks starting from the current one downwards. It is particularly useful for making a working which is close to another one or which has the same x-y position as another one a current working, such as for example drilling, gluing, insertion and only one of which, having the same coordinates, must be placed in view.
- **Select->Go to block:** the window required for the insertion of the line number (block), which is to be made the current one.
- **Select->Next block:** the next working becomes the current one. The **[CTRL+D]** key combination is linked to the command.
- **Select->Previous block:** the previous working becomes the current one. The **[CTRL+U]** key combination is linked to the command.
- **Select->First block:** the first working of the face becomes the current one. The **[Home]** key is linked to the command.
- **Select->Last block:** the last working of the face becomes the current one. The **[End]** key is linked to the command.
- **Select->Setup block:** if it is a profile working this command makes the setup working the current one.

3.4.5 Find

The operation of workings search on the active face can be performed selecting the ***Edit->Find*** menu.
A dialog is shown where the search criteria are to be entered.



Find

Find

☑ Working Code

L01 :Pf(X,Y,Z)

Current Code

□ Level 0

□ Origin [0,0]

Direction

○ Up

● Down

☑ Match on View

Next

Abort

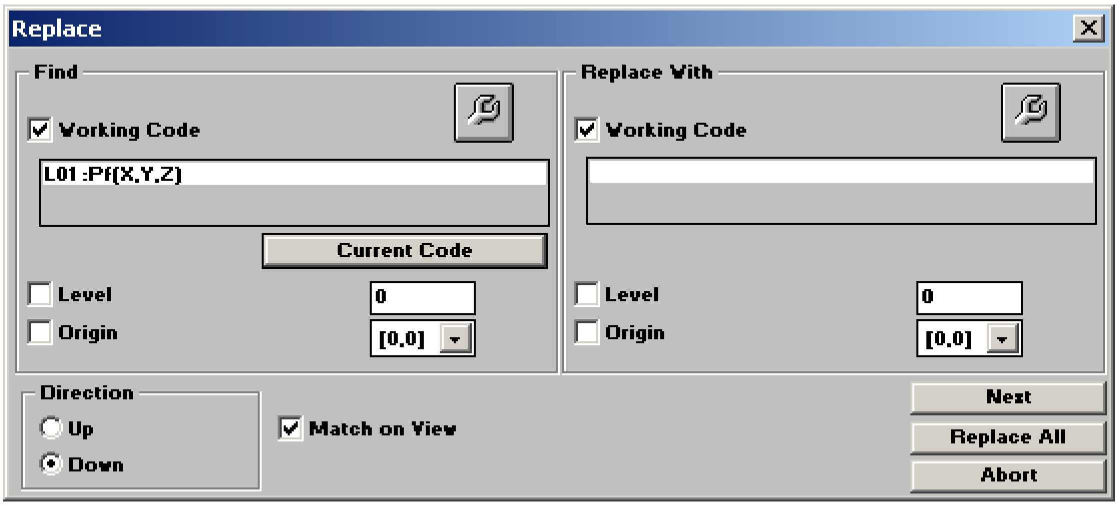
The criteria in detail are:

- **Working code:** activate the box if you wish to search by working code.
- : button to access the Workings Palette and select the working code to be used for the search.
- **Current code:** to use the code of the current working as the search code.
- **Level:** activate this box and indicate the level number if you wish to use this search criterion. This field is not always available.
- **Origin:** activate this box and indicate the origin, if you wish to use this search criterion. This field is not always available.
- **Direction up/down:** activate the desired search direction, back or forward, with respect to the current block.
- **Match on view:** activate this box if you wish to carry out a search only on the workings currently in view.

The **[Next]** button allows you to carry on with the search, the **[Cancel]** button ends the search operations.

3.4.6 Replace

The operation of replacing workings or properties of the workings on the active face can be made from the ***Edit->Substitute*** menu.
A dialog box appears for all the search criteria to be entered as well as the criteria relevant to the replacement to be made.



Replace

Find
- ☑ Working Code: L01 :Pf(X,Y,Z)
- □ Level: 0
- □ Origin: [0,0]

Replace With
- ☑ Working Code:
- □ Level: 0
- □ Origin: [0,0]

Direction
- ○ Up
- ● Down

☑ Match on View

Search criteria area:

- **Working code:** activate the box if you wish to search by working code.
- : button to access the Workings Palette and select the working code to be used for the search.
- **Current code:** to use the code of the current working as the search code.
- **Level:** activate this box and indicate the level number if you wish to use this search criterion. This field is not always available.
- **Origin:** activate this box and indicate the origin, if you wish to use this search criterion. This field is not always available.

Replace criteria area:

- **Working code:** activate the box if you wish to search by working code.
- : button to access the Workings Palette and select the working code to be used for the search.
- **Current code:** to use the code of the current working as the search code.
- **Level:** activate this box and indicate the level number if you wish to use this search criterion. This field is not always available.
- **Origin**: activate this box and indicate the origin, if this type of replacement is activated. This field may not be available.
- **Up/down direction:** activate the desired search direction, backwards or forwards, with respect to the current block.
- **Corresponding to view:** activate this box if you wish to carry out a search/replacement only on the workings currently in view.

The buttons allow you to establish how to make the replacement. When the **[Next]** button is pressed a window with the following buttons appears:

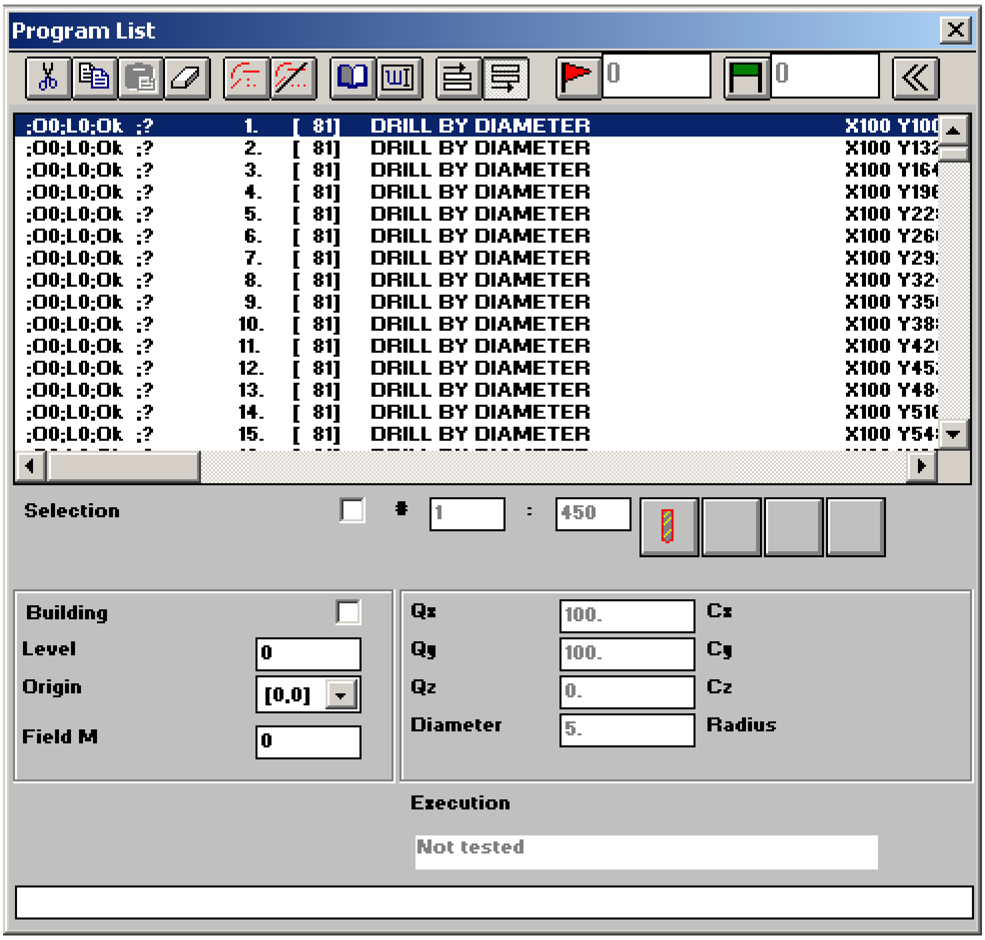
- **[Next]** allows you to skip the replacement of the working you have found, search for the next one and exit if there are no others.
- **[Replace]** allows you make the replacement, then search for the next working and exit if there are no others.
- **[Replace all]** automatically makes all the replacements possible and exits.
- **[Cancel]** stops the replacement operations.

The **[Replace all]** button automatically makes all the replacements possible. The **[Cancel]** button allows you to quit all the replacement operations carried out .

3.5 Program List

3.5.1 Composition

The list of blocks that comprise the program may be viewed for a complete view of all the working sequences introduced. The window also allows you to change and insert workings.
Access the list in the following way: select **Window->Program list**.
The command may also be activated with the [**ALT+L] and [CTRL+L]** key combinations**.**
A window appears containing the program list.



0 0

;O0;L0;Ok ;?	1.	[81]	DRILL BY DIAMETER	X100 Y10(
;O0;L0;Ok ;?	2.	[81]	DRILL BY DIAMETER	X100 Y132
;O0;L0;Ok ;?	3.	[81]	DRILL BY DIAMETER	X100 Y164
;O0;L0;Ok ;?	4.	[81]	DRILL BY DIAMETER	X100 Y19(
;O0;L0;Ok ;?	5.	[81]	DRILL BY DIAMETER	X100 Y22
;O0;L0;Ok ;?	6.	[81]	DRILL BY DIAMETER	X100 Y26
;O0;L0;Ok ;?	7.	[81]	DRILL BY DIAMETER	X100 Y29
;O0;L0;Ok ;?	8.	[81]	DRILL BY DIAMETER	X100 Y32
;O0;L0;Ok ;?	9.	[81]	DRILL BY DIAMETER	X100 Y35
;O0;L0;Ok ;?	10.	[81]	DRILL BY DIAMETER	X100 Y38
;O0;L0;Ok ;?	11.	[81]	DRILL BY DIAMETER	X100 Y42
;O0;L0;Ok ;?	12.	[81]	DRILL BY DIAMETER	X100 Y45
;O0;L0;Ok ;?	13.	[81]	DRILL BY DIAMETER	X100 Y48
;O0;L0;Ok ;?	14.	[81]	DRILL BY DIAMETER	X100 Y51
;O0;L0;Ok ;?	15.	[81]	DRILL BY DIAMETER	X100 Y54

Selection ☐ # 1 : 450

Building ☐
Level 0
Origin [0,0]
Field M 0

Qx 100. Cx
Qy 100. Cy
Qz 0. Cz
Diameter 5. Radius

Execution
Not tested

As you can see the window is divided into several sections which include text boxes, view areas and buttons, which are described below:

Button Areas or Toolbar

Consist of several buttons and two text boxes .
The buttons allow you to select the blocks included in the interval, to carry out editing operations on these blocks or on the current one, and to access the enter and edit commands.
The **[>>]** button allows you to activate and deactivate (**[<<])** the complete view of the Program List window .

List area

contains the programmed workings list, in the order in which they were inserted, every line of the list is comprised of several fields, each of which contains the main data of the working; the current working or block is the one which is highlighted and all the information relevant to it is displayed in

the areas below.
The data fields in the line, from left to right, are:

- **Selected working:** if it is selected the asterisk (*) appears, otherwise it is empty.
- **Origin:** origin number (from 0 to 4); this field is not always available.
- **Level number**: the level number (L00-L08) to which the working is linked; this field is not always available.
- **Diagnostics:** "Ok" if the interpretation of the working does not have any set diagnostics, otherwise "Nok"
- **Logical conditioning:** "?" if the logical conditionings are not applied, if applied then: "On" appears if the working is executive , otherwise "Off"
- **Line number:** is the progressive number in the face working list
- **Working code:** is the working code number
- **Working description:** description of the working
- **X, Y, Z coordinates:** working application co-ordinates

Other areas
contain the main properties or characteristics of the current working, and they include:

- **Selection:** indicates the selection status of the current block and even allows you to make the selection.
- **Progressive number:** of the current block (#).
- **Total number:** of blocks that comprise the program (:).
- **One or more icons:** which provide a visual indication of the type of current working (point, setup, arc, line, logical, macro or subroutine).
- **Building:** if active it indicates that a building working is applied to the current block.
- **Level:** to which the current block is linked.
- **Origin:** of the current block.
- **M field:** of the current block.
- **Qx, Qy, Qz:** coordinates of the point of application of the working.
- **Diameter:** of the tool, only if it is a point or setup working.
- **Cx, Cy, Cz:** coordinates of the centre. They are only significant in the event of a circular interpolation.
- **Radius:** radius of the arc. Only significant in the event of a circular interpolation.
- **Execution:** shows if the logical conditions must not be applied when the workings are executed, or which is the logical condition to apply:
  - Not tested if the logical conditions are not applied.
  - Verified as active if the condition to be verified is ON.
  - Verified not active if the condition to be verified is OFF.
- **Messages:** different, depending on the situation.

3.5.2 Inserting workings

Insertion in this work mode is by means of the **[Insert working]** icon, the new working is inserted in the program above or below the working, depending on the mode which is active at the moment, which may be selected by means of the icons **[Insert block in the previous position]** and **[Insert block in the next position**].
Consequently to make one or more insertions, proceed as follows:

- select the working which is to have the new one inserted above or below it, so that it becomes the current one;
- select the **[Insert work]** icon;
- chose the desired working code and type from the list of workings;
- enter the data relevant to the selected working;
- at the end the Working List box reappears to allow you to insert a new working or to cancel the insertion.

3.5.3 Modifying workings

Either the current block or all the blocks from the current one onwards may be modified. Proceed as follows :

1. select the working from which you wish to begin the edit procedure;
2. select the [Edit the parameters of the block] icon, or position the cursor on the desired block and press the ENTER key, or double click the current block;
3. edit the data;
4. at the end of the editing, if it is not the last one in the program, the next working automatically appears ready for the data to be edited;
5. the edit procedure can be interrupted by selecting the **[Cancel]** button or by pressing the **[ESC]** key.

The main properties of the current block may also be edited.

3.5.4 Editing workings

The blocks in the program may be edited as if they are normal texts, using the following commands: Cut, Copy, Paste, Delete, which are accessed through dedicated buttons.

All the editing operations refer to the current block or to the selected working block by means of the **[red flag]** and **[green flag]** icons, in the following way:

1. go to the first block in the interval;
2. select the button marked with a **[red flag]:** the working number appears in the box next to it;
3. go to the last working in the interval;
4. select the button marked with a **[green flag]:** the working number appears in the box next to it.

Or you can set the first and last block numbers directly by typing them into the two boxes.

A description of the editing operations follow:

Cut The blocks are eliminated from the program of the active face and copied to a temporary memory so that they can be re-inserted using the Paste command, either to another face or to another program.
The elimination is only carried out after the user gives an affirmative reply to a request for confirmation.

Copy The blocks are copied to a temporary memory so that they can be re-inserted using the Paste command, either to another face or to another program.

Paste The blocks which were previously copied to the temporary memory are re-inserted into the program, either before or after the current block, depending on the type of insertion that is active at the time. The command is useful for re-inserting the last cut block or for inserting blocks which were previously copied to other faces or to another program.

Delete The blocks are deleted from the program and lost.
The deletion is only carried out after the user gives an affirmative reply to a request for confirmation.

3.6 Parametric programming

3.6.1 Variables - Type "r"

In a program until a maximum of 300 'r' type variables, named from **r0** to **r299**, may be used.
Every variable may be programmed in a parametric format and is characterised by:

- **a typology**, defining its numerical format, between:
  - *float* numbers in floating point

- *integer* integer numbers (practically, the integer part of a float)
- *string* alphanumeric string of characters
- *flag* *boolean variable (for the moment, not yet available!)*

The *float* and *integer* typologies are managed in the same mode, but must be chosen conforming to their use as, for instance:
a variable must be used on boolean logic for an IF testing must be assigned with "integer" typology, since its value will be used for a '0' comparing.
The *float* typology must be used for the variables where the decimal part may be significant (as for *positions, angles*, and so on).
The *string* typology will be used for the parameter with non numeric meaning as, for instance, for the subroutines or macro *name* coding.

- **a *defining string*** (max 50 characters): allows to define the mathematical expression of the variable, when coded in parametric format.
- **a *variable value*** : (type *float*), corresponding to the mathematical result of the string expression.

In the case the variable has a numerical typology, this value is meaningful, because it corresponds to the solution of the string value, interpreted as a mathematical expression.
In the case the variable has a 'string' typology, this value is always assigned as '0', since not meaningful.

3.6.2 Variables - Type "j"

Up to 100 special variables, named from **j0** to **j99** are available.
Every "j" type variable may assume only a *float* value*.* These variables define a set of global variables may be seen and manipulated by the program, with the following characteristics:

- only one set of "j" variables is available for all the faces
- the numeric value may be reassigned into the program text, but this new value will be considered only in the current face
- no correlation may be assigned between these variables, when used on different faces
- every time the face part of a program is compiled, the complete set is reinitialised to '0'

3.6.3 Working parameters

Every working parameter is characterised by a type of format.
Some of the available format types are described below, in particular the ones involving direct programming (typology corresponding to data entry controls, not selection controls), with the distinction between :

- parameter with data entry that can be reduced to a numerical value.
- parameter with string type data entry.

An indirect, strictly numerical, whole type programming is managed for selection controls.
A numerical format type parameter generally allows parametric programming (in the viewed cases), in string format with a maximum of 50 characters. The string solution, interpreted as a mathematical expression, leads to the determination of the numerical value of the parameter.
A string format type parameter does not assume a significant numerical value (it is set at 0), the set string must assume a direct meaning. The maximum length of the set string can be up to 200 characters.
Principally, a string parameter relates to the assignment of the name of the subroutine or macroprogram.
An example of direct assignment in string format: **"c:\albatros\product\prg27.abc"** .

3.6.4 String type parameters

In any case it is possible to assign a string type working parameter or "r" variable in parametric form.
Two types of syntaxes are permitted:

a) "*r5:\albatros\ *r6*r7.prg"
b) "*pr[.....]".

Formalism a): "*r5:\albatros\ *r6*r7.prg
The "*rnn" expressions, where "nn" represents the numeric value of the r variable selection (from 0 to 299), have a parametric interpretation.
All the variables indicated must be assigned as string types and up to 5 replacements are managed, for example:
"porte*r28.*r29" determines the assignment of the string value of the variables r28 and r29 to the parameter in question. If r28="p007", r29="12", the development of the parameterization determines the development as "porte\p007.12".

Formalism b): "*pr[.....]".
Is interpreted rigidly:

- first character'*'
- "pr[...]" follows with the argument in square parentheses which can be assigned parametrically. The pr function argument must have a *nn numeric solution*, which corresponds to the r variable selection numeric value. The resulting variable must be assigned in string form.

This second formalism therefore allows a string type parametric programming through the solution of a numeric type parameterization. All the comments made in this section relating to a numeric parameter/variable parametric programming apply to the argument of r variable addressing (pr[..]) .
Example:
r20="*pr[r5+r7]" where:
r5 numerical type variable (integer or float) with assigned value of 5
r7 numerical type variable (integer or float) with assigned value of 12
==> *pr[r5+r7]=*pr[5+12]=*pr[17]

The r17 variable must be a string type, and the "r20" variable must assume a $ value of "r17".
In the example: the "r5" and "r7" variables cannot be string type variables, because they define a numeric argument.
Use of "*rnn" and/or "*pr[..]" strings not considered in the formats described here solve error situations.

NOTE: a string type parameter which corresponds to one of the points below is interpreted as a comment :
a) indicated field of comment
b) $ value set beginning with "//"

Use of string format for name of subroutine or macro
At this point a few explanations are required regarding the interpretation of a parameter string value, when it is used to assign a name to a subroutine or macroprogram. In general a pathname can be assigned, either explicitly or partially implicitly:

- the program (subroutine or macroprogram) name can be up to 8 characters.
- the program (subroutine or macroprogram) extension can be up to a maximum of 3 characters.
- indication of the name only searches the program along the path allocated to it by the system (dirprod+"\SUB" for subroutine, dircadcfg+"\MCR" for macroprogram). Example: "pippo.txt".
- indication of the name with simple directory addressing searches the program along the path allocated to it by the system (dirprod+\SUB for subroutine, dircadcfg+"\MCR" for macroprogram), with the addition of the subaddressing indicated. Example: "ante\pippo.txt". In the event of subroutines it loads the file : dirprod+"\SUB\"+"ante\pippo.txt".
- indication of the name with directory addressing preceeded by ".\" : idem previous case d) . Example: ".\ante\pippo.txt". In the event of subroutines it loads the file: dirprod+"\SUB\"+"ante\pippo.txt".
- indication of the name with directory addressing preceded by "..\" is only accepted for subroutines: it searches the program along the dirprod path, with the addition of the subaddressing indicated. Example: "..\ante\pippo.txt". It loads the file: dirprod+" \"+"ante\pippo.txt".
- indication of complete drive pathname: only searches the program according to the given path . Example: "c:\albatros\product.mio\ante\pippo.txt".

3.6.5 Special numeric format parameters

Also in the case of parameters characterised by a numerical typolgy, a special interpretation is provided, concerning the absolute or incremental positions programming.

Any "position" parameter in a work normally may be programmed as an absolute position (referred to the Origin) or an incremental position (referred to the previous point).

For instance, we can consider an arc programming on the (xy) plane, where:

a) (QX, QY) may be assigned in parametric format to define, in absolute or relative format, the arc final point
b) (QZ) may be assigned in parametric format to define, in absolute or relative format, the arc final depth
c) (QI) may be assigned in parametric format to define, relatively to the initial point, the X abscissa of the Center

Every one of these parameters, a special syntax allows *to force an absolute interpretation* of the passed numerical value, using the "*a;....*" expression.

For instance, we suppose to enter:

```
QX=100  QY=200        absolute
QI=a;200
```

==> the QX and QY parameters are an absolute position value
QI is programmed with an absolute center abscissa = 200, because forced by the syntax.

3.6.6 Table of the mathematical operators and functions

In the mathematical expressions supported by the Parametric programming, all the *symbolic* or *numerical* variables may be linked by a number of **mathematical** or **logical Operators** or **Functions,** may be expressed, as showed in the following table, in a "Extended Mnemonic" or "Synthetic" format.

Functions and Operators Table

SINGLE OPERAND FUNCTIONS

Mnemonic	**Synthetic**	**Description**
sin	s	Computes the sine of the argument (in °)
cos	c	Computes the cosine of the argument (in °)
tan	t	Computes the tangent of the argument (in °)
asin, as	d	Computes the arc-sine of the argument, returns the result in °
acos, ac	and	Computes the arc-cosine of the argument, returns the result in °
atan, at	f	Computes the arc-tangent of the argument, returns the result in °
abs	a	Computes the absolute value of the argument
pow	p	Returns the power of 2 of the argument
sqrt	q	Returns the square root of the argument (positive)
int	i	Returns the integer part of the argument
inv	v	Returns the reverse of the argument (1/x)
gr	g	Converts the argument from radians to degrees (°)

odd	Returns 1 if the integer part of argument is odd; otherwise 0
strlen	Returns the string length of the *rnn* variable contents, where nn is the argument of the function

FUNCTIONS MULTI-OPERANDS

Mnemonic	**Description**
getat[nn,np]	Returns the np character of the string of rnn variable, with: nn= address of the 'r' variable (1° argument) np= position of the character (significant from 1)
pown[nb,ne]	Computes the power nb[base] for ne[exponent]. Power is valid if included between 0 and 10 and is applied on the integer part.
hypot[c1;c2]	determines the hypotenuse of a triangle given the sides.
Ifelse[nc,n1,n2]	Minimum ternary operator; returns n1 if nc is different from 0, otherwise n2,
ifcase[nc1,nesp, nc2,n1,n2]	Complete ternary operator; evaluates the expression (nc1 ? nc2), with ? defined in *nesp*
case[nc;nc1:nv1;nc2:nv2; ... ;nvdef]	Condition test operator: evaluates the expressions (nc=nc1), (nc=nc2), ... and returns the "nv" value associated to the first expression that evaluates true.
min[n1,..,n30]	Returns the minimum between a max of 30 arguments
max[n1,..,n30]	Returns the maximum between a max of 30 arguments
ave[n1,..,n30]	Returns the average between a max of 30 arguments
sum[n1,..,n30]	Returns the sum of a max of 30 arguments
minr[n1,n2]	Returns the minimum between the values assigned to the r variables in the range (n1, n2). All the variables must have typology: float or integer
maxr[n1,n2]	Returns the maximum between the values assigned to the r variables in the range (n1, n2). All the variables must have typology: float or integer
aver[n1,n2]	Returns the average between the values assigned to the r variables in the range (n1, n2). All the variables must have typology: float or integer
sumr[n1,n2]	Returns the sum of the values assigned to the r variables in the range (n1, n2) All the variables must have typology: float or integer
minj[n1,n2]	Returns the minimum between the values assigned to the j variables in the range (n1, n2)
maxj[n1,n2]	Returns the maximum between the values assigned to the j variables in the range (n1, n2)
avej[n1,n2]	Returns the average between the values assigned to the j variables in the range (n1, n2)
sumj[n1,n2]	Returns the sum of the values assigned to the j variables in the range (n1, n2)
min$[n1,n2]	Returns the minimum between the values

assigned to the $ variables in the range (n1, n2)

max$[n1,n2]
Returns the maximum between the values assigned to the $ variables in the range (n1, n2)

ave$[n1,n2]
Returns the average between the values assigned to the $ variables in the range (n1, n2)

sum$[n1,n2]
Returns the sum of the values assigned to the $ variables in the range (n1, n2)

prfi[nm, ng, nt]
prfi[ng, nt]
prfi[nt]
Returns the tool Diameter , selected by: nm = machine (1÷10) (= 1 if missing) ng =group (1÷10) (= 1 if missing) nt = tool (1÷200)

prface[nm, ng, nt, nf]
prface[ng, nt, nf]
In the complete form (4 assigned parameters) restores 1 if the tool can work on the face "nf", if not, 0 . If only the first 3 parameters are assigned, the value of the tool parameter which indicates the mask of the tool work faces is restored. The *nm*, *ng*, *nt*, arguments are identical to the "prfi" function.The other argument is :nf = face number.

prattr[nm, ng, nt]
prattr[ng, nt]
prattr[nt]
Returns the value 1 if outfit is present; otherwise 0. The value is returned = 1 only in the case of a reference tool assigned for multipoint head. See also the "prfi" function, for the arguments description

profx[nm, ng, nt]
profx[ng, nt]
profx[nt]
Returns the X symmetry parameter of the selected tool. See also the "prfi" function,, for the arguments description

profy[nm, ng, nt]
profy[ng, nt]
profy[nt]
Returns the Y symmetry parameter of the selected tool. See also the "prfi" function, for the arguments description

prfin[nm, ng, nt, np]
prfin[ng, nt, np]
prfin[nt, np]
Returns the Diameter of the added point, numbered "np", for the selected tool. The parameter "np" may assume only the 1 to 4 values. See also the "prfi" function, for the arguments description

prcx[nm, ng, nt, np]
prcx[ng, nt, np]
prcx[nt, np]
Returns the X corrector of the added point, numbered "np", for the selected tool. The parameter "np" may assume only the 1 to 4 values. See also the "prfi" function, for the arguments description

prcy[nm, ng, nt, np]
prcy[ng, nt, np]
prcy[nt, np, np]
Returns the Y corrector of the added point, numbered "np", for the selected tool. The parameter "np" may assume only the 1 to 4

values. See also the "prfi" function, for the arguments description

prtip[nm, ng, nt]
prtip[ng, nt]
prtip[nt]

Returns the tool typology. See also the "prfi" function, for the arguments description

prtool[nm, ng, nt, nkind, ndefault, nconv]
prtool[ng, nt, nkind, ndefault, nconv]
prtool[nt, nkind, ndefault, nconv]

Restores the value allocated to the tool's "nkind" parameter, defined by *nm*, *ng*, *nt* (see"prfi" function description). nkind = parameter type of the assigned tool (obligatory parameter). ndefault = default value, if the parameter has not been configured. nconv = major conversion number on unit of measurement (2 or 3 significant); if the conversion number is invalid no error is signalled.

VARIABLE ARGUMENT

Mnemonic	**Synthetic**	**Description**
pi	w	pi greek (p = 3.1415..)
eps		Threshold (epsilon) for linear position, assumes a value conforming to the current unit of measure of the program: 0.001 for mm 0.001/25.4 for inch
cnq		Conversion factor for linear position, assumes a value conforming to the current unit of measure of the program: 1 for mm 1/25.4 for inch
l		Piece X Dimension
h		Piece Y Dimension
s		Piece Z Dimension
lf	x	Face X Dimension (only for a face programming)
hf	y	Face Y Dimension (only for a face programming)
sf	z	Face Z Dimension (only for a face programming)
prgrun		Program execution environment: 1 = *Edicad* running 0 = *Optimize running*
prgn		Flag of Normal execution: 1 = normal(always verified in programming) 0 = mirror
prgx		Flag of X mirrored execution: 1 = X mirror (never verified in programming) 0 = different execution
prgy		Flag of Y mirrored execution: 1 = Y mirror (never verified in programming) 0 = different execution
prgxy		Flag of execution mirror XY: 1 = mirror execution XY 0 = different execution
r0-r299		Piece Variables (only variables of typology: float or integer)
$0-$99		Text auxiliary Variables (valid only in a macro)

j0-j99	Global Variables (not valid in a macro)
pr[.]	Reference to a piece variable (only variables of typology: float ed integer). The parametric format is compulsory (use of brackets [.]): returns the value of the variable **rn**, where n=computed value between the [.] brackets.
p$[.]	Reference to an auxiliary variable in a macro text. The parametric format is compulsory (use of brackets [.]): returns the value of the variable **$n**, where n = computed value between the [.] brackets.
pj[.]	Reference to a global variable in a program text. The parametric format is compulsory (use of brackets [.]): returns the value of the variable **jn**, where n=computed value between the [.] brackets.
face	Face number (only for face programming)
'ch'	Returns the numerical decimal value corresponding to the ASCII coding of the 'ch' character. Valid codes: from 32 (' ') until 126 ('~').

OPERATORS

Mnemonic	**Description**
+	Addition
-	Subtraction
*	Multiply
/	Division
%	Module of the division
#	Integer of the division
?	Step Adjust
&	Logical bit to bit AND of the integer part of the operands, resetting the decimal part of the result
|	Logical bit to bit OR of the integer part of the operands, resetting the decimal part of the result
(..)	Level of parenthesis
[..]	Argument Delimiters for a function
.	Separator from integer to decimal part of a numerical argument
, ;	Separators between arguments in a multi-operands function.

3.6.7 Single Operand Functions

For every function, either the mnemonic extended or the synthetic coding is accepted.
The argument of a function may be passed in numerical or parametric format:

- numeric argument: cos60
- parametric argument : sqrt[r5+70]

As arguments of a function, the following items may be used:

- the piece dimensions (l, h, s)
- the piece variables (r0, r1, r2...), (j0, j1, j2 ...)
- the face dimensions (lf, hf, sf)
- the parenthesis "( )"
- the functions

The argument in parametric format must be embedded in the brackets [ ].
A multiple Functions nesting is allowed as, for instance:
sqrt[pow[r40]+pow[r41]]

In the Functions, negative arguments must be embedded compulsory in the brackets [ ] as, for instance:

sin[-45]

equivalent to the numerical format to: sin 315

We can consider any single function:

sin **returns the sine of the argument**
cos **returns the cosine of the argument**
tan **returns the tangent of the argument**
The argument of the functions must be in degrees. No error conditions are provided.

asin **returns the arc-sine of the argument**
acos **returns the arc-cosine of the argument**
atan **returns the arc-tangent of the argument**
For the first two functions (asin, acos), the value of the argument must be included between - 1 and 1. The returned value is
included between 0 and 180°.

abs **returns the absolute value of the argument**
Returns the absolute value of the argument. No error conditions are provided.

pow **computes the square (power of 2) of the argument**
sqrt **returns the square root of the argument**
Computes the square root of the argument. Argument must be positive (>=0).

int **returns the integer part of the argument**
Returns the integer part of the argument (cut-off). No error conditions are provided.

inv **returns the reverse of the argument**
Returns the reverse of the argument (1/x). The argument cannot be zero.

gr **converts the argument from radiants to degrees**
Converts the argument from radiants to degrees (1 radiant=(180/p) °). No errors occur.

Odd **tests if a value is even or odd**
Returns 1 if the integer part of the argument is odd; otherwise 0

strlen **returns the length of the string of the rnn variable,**

where nn=argument of the function.

Returns the number of characters of the string stored in the r type variable addressed by the nn argument.

The argument must be included between 0 and 299.

The variable is searched before the current development (another r variable assigning, parameter or working variable assigning, subroutine or macro development).

If the variable is unassigned, a 0 value is returned.

Notice: the value returned refers to the original string length. This must

be considered if the variable has a string typology: the STRLEN function do not resolve an eventual parametric expression then dont operate on the result of the string development.

Ex.: the selected variable (of string type) r5 is assigned as : "submio*r4",   (where r4="alpha")

if the parametric expression should be resolved : r5 = "submio\alpha" then "strlen[5]" returns 10 corresponding to the length of the string r5:: "submio*r4".

3.6.8 Multi operand Functions

In this case, only the extended mnemonic code is accepted.
The function operands must be embedded in the brackets, divided by the "," or ";" characters.
Every operand may be entered in numerical or parametric format:
a) numerical arguments: min[12, 15, 28]
b) also parametric arguments: min[12, abs[r4], 25, r5+r7]
About the argument parametric formalism, all the single argument functions consideration are still valid.
Some functions require a fixed argument number, as : getat, minr,...;
Other functions accept a variable number of arguments, from a minimum of 1 to a maximum of 30.
An error message is provided in case of :

- function without arguments
- function with an invalid arguments number

We can consider every function in detail:

getat[nn,np]

returns a required character from the rnn variable string
Returns the decimal value (corresponding to the ASCII coding) of the **np.**th character of the selected **rnn** variable string.
nn (1° argument) = sequence number for the rnn variable addressing. The value of nn must be included between 0 and 299.
np (2° argument) = sequence number of the character in the string (significant from 1).
The variable is sampled before the current development , as in the case of the STRLEN function.
If the rnn variable has not been assigned, a 0 value is returned
The arguments of the function must be 2.
Ex.:

```
if assigned as    r5="r2+r3"
getat[5,2]              returns '2'=50
getat[5,10]             returns 0.
```

pown[nb,ne]

Power
The function performs a power, where:
nb (1° argument) = base
np (2° argument) = exponent (must be included between 0 and 10, and is considered only for its integer part).
Particular cases:
a) nb#0 and ne=0 returns 1
b) nb=0 returns 0
Only 2 arguments are allowed 2.

Ex.:

```
pown[5,2]  =  5 * 5 =25
pown[5,3.5]  =  pown[5,3]  = 5 * 5 * 5 = 125
pown[5, 0]  =  1
pown[0, 5]  =  0
```

hypot[c1,c2]

determines the hypotenuse of a triangle
The function restores the hypotenuse of the rectangle triangle which has assigned legs (c1, c2).

ifelse[nc,n1,n2]

ternary operator
The function evaluates the first argument (nc): if different from 0 it returns n1, otherwise n2.
The arguments of the function must be 3.

ifcase[nc1,nesp,nc2,n1,n2]

complete ternary operator
The function evaluates the condition (nc1 ? nc2), with ?=nesp:

if the condition is verified true it returns n1, otherwise n2.
The arguments nc1 and nc2 are numerical or parametric
The nesp argument is interpreted for assigning the condition between nc1 and nc2:
value 0 corresponds to < (smaller)
value 1 corresponds to <= (smaller or equal)
value 2 corresponds to > (greater)
value 3 corresponds to >= (greater or equal)
value 4 corresponds to = (equal)
value 5 corresponds to <> (different)
The corresponding symbolic forms may also be assigned to argument esp, instead of the numerical value.
Example: condition " greater or equal " may be assigned with value 3 or with ">=".
The relation of difference may be expressed with "<>" or with "#".
The arguments of the function must be 5.
Examples:
ifcase[5, >=, 12, 3, 25] = 25
ifcase[5, <, 12, 3, 25] = 3
ifcase[5, <>, 12, 3, 25] = 3

case[nc;nc1:nv1;nc2:nv2; ... ;nvdef] **condition test operator**
This function tests the appearance of a condition, from the assigned ones, and restores the assigned value to the condition that is verified as valid.
The arguments are:
nc = value to be evaluated; it may be numeric or parametric.
nc1 = value to coincide with the other 'nc' values; it may be either numeric or parametric.
nv = value restored by the function if 'nc' = 'nc1' coincide; it may be either numeric or parametric.
nvdef = restored default value if no correspondence is detected; if not assigned value 0 is restored. It is not obligatory and if it is assigned it is placed as the last argument.
The character separating 'nc' and 'nc1' is '**;**', whereas the character between 'nc1' and 'nv1' must be '**:**'.
The maximum number of cases is 9, including 'nvdef'.
Example:
case[h;100:r0;200:h-100;l:l/2;h]
The example demonstrates the following conditions:
if h=100 the value of the 'r0' variable is restored,
if h=200 the datum value of 'h-100' is restored,
if h=l the datum value of 'l/2',
if no correspondence is detected the value assigned as 'nvdef' is restored, that is 'h' is restored.

min[n1,..,n30] **returns the minimum value between the arguments**
max[n1,..,n30] **returns the maximum value between the arguments**
ave[n1,..,n30] **returns the average value between the arguments**
sum[n1,..,n30] **returns the sum of the arguments**
A maximum of 30 arguments are allowed 30.
Ex.:
min[5, 12, 3, 25] = 3
max[5, 12, 3, 25] = 25
ave[5, 12, 3, 25] = (5 + 12 + 3 + 25) /4 = 11.25
sum[5, 12, 3, 25] = 5 + 12 + 3 + 25 = 45

minr[n1,n2] **returns the minimum between the values of the "r" variables in the range (n1, n2)**
maxr[n1,n2] **returns the maximum between the values of the "r" variables in the range (n1, n2)**

aver[n1,n2] **returns the average between the values of the "r" variables in the range (n1, n2)**

sumr[n1,n2] **returns the sum of the "r" variables in the range (n1, n2)**

Only 2 arguments, for these functions, are allowed.
All the addressed variables must have float or integer typology. The n1 and n2 arguments
must be included in the range: r0 to 299.
These functions may be used only on the "r" variables assigned on the main text (program or macro).

minj[n1,n2] **returns the minimum between the values of the "j" variables in the range (n1, n2)**

maxj[n1,n2] **returns the maximum between the values of the "j" variables in the range (n1, n2)**

avej[n1,n2] **returns the average between the values of the "j" variables in the range (n1, n2)**

sumj[n1,n2] **returns the sum of the "j" variables in the range (n1, n2)**

Only 2 arguments, for these functions, are allowed.
The n1 and n2 arguments must be included in the range: j0 to j99.
These functions may be used only to assign a working parameter on the main
program text (not a macro) or in the subroutine development.

min$[n1,n2] **returns the minimum between the values of the "$" variables in the range (n1, n2)**

max$[n1,n2] **returns the maximum between the values of the "$" variables in the range (n1, n2)**

ave$[n1,n2] **returns the average between the values of the "$" variables in the range (n1, n2)**

sum$[n1,n2] **returns the sum of the "$" variables in the range (n1, n2)**

Only 2 arguments, for these functions, are allowed.
The n1 and n2 arguments must be included in the range: $0 to $99.
These functions may be used only to assign a working parameters of the primary text (not macro-program) or in the subroutine development.

prfi[nm,ng,nt] **returns the Tool Diameter**
prfi[ng,nt]
prfi[nt]

Returns the diameter of the tool selected by the following arguments:
nm = machine number (1 to 10).
ng = group number (1 to 10).
nt = tool number (1 to 200).

prface[nm, ng, nt, nf]
prface[ng, nt, nf]

In the complete form (4 assigned parameters) restores 1 if the tool can work on the face "nf", if not, 0 .
If only the first 3 parameters are assigned, the value of the tool parameter (which indicates the mask of the tool work faces) is restored, if the tool does not exist it restores the 0 value.
c

prattr[nm,ng,nt] **verifies if the outfit is present**
prattr[ng,nt]
prattr[nt]

Returns the 1 value 1 if present, otherwise returns 0.
Value will be 1 only in the case of the reference tool

assigned for a multi-point head.
The arguments have the same meaning as the "prfi" function.

profx[nm,ng,nt] **returns the X symmetry parameter for the addressed tool**
profx[ng,nt]
profx[nt]
profy[nm,ng,nt] **returns the Y symmetry parameter for the addressed tool**
profy[ng,nt]
profy[nt]

Returns the X or Y symmetry parameter for the addressed tool.
The arguments have the same meaning as the "prfi" function.
The function may return a not zero value only in the case of multi-point head.

prfin[nm,ng,nt,np] **returns the diameter of the "np" added tool-point**
prfin[ng,nt,np]
prfin[nt,np]

Returns the diameter of the "np" tool-point (np = 1 to 4).
The arguments have the same meaning as the "prfi" function.
The function may return a not zero value only in the case of multi-point head.

prcx[nm,ng,nt,np] **returns the X corrector of the "np" added tool-point**
prcx[ng,nt,np]
prcx[nt,np]
prcy[nm,ng,nt,np] **returns the Y corrector of the "np" added tool-point**
prcy[ng,nt,np]
prcy[nt,np]

Returns the X or Y corrector of the "np" tool-point (np = 1 to 4).
The arguments have the same meaning as the "prfi" function.
The function may return a not zero value only in the case of multi-point head.

prtip[nm,ng,nt] **returns the Tool typology**
prtip[ng,nt]
prtip[nt]

Returns the Tool typology.
The arguments have the same meaning as the "prfi" function.

prtool[nm, ng, nt, nkind, ndefault, nconv]
prtool[ng, nt, nkind, ndefault, nconv]
prtool[nt, nkind, ndefault, nconv] Restores the value allocated to the tool's "nkind" parameter.
The *nm*, *ng*, *nt*, arguments are identical to the "prfi" function.
The other arguments are :
nkind = parameter type of the assigned tool (obligatory parameter).
ndefault = default value, if the parameter has not been configured.
nconv = major conversion number on unit of measurement (2 or 3 significant); if the conversion number is invalid no error is signalled.
'nconv ' is used if the program is written in inches ([inch]):

- assign nconv=2 to convert a dimension parameter ,
- assign nconv=3 to convert a speed parameter.

The minimum number of parameters is 2 and the minimum format is:
prtool[ntool;nkind]; no default value is allocated and the conversion is not made.

3.6.9 Variable Arguments

The variable arguments using, in a mathematical expression, allows a true parametric programming in the work definition, because any parameter may change according the current value of the used variable.

In the work parameters programming , the (lf, hf, sf) arguments assume the current face dimensions values.

3.6.10 Operators

Some mathematical operators deserve a study in depth.

% module of the division

Ex.: 100 % 7 = 2
Computing procedure for the result :
a) executes the division: 100 / 7 = 14.2857
b) break up the decimal part of the result: 14.2857 - 14 = 0.2857
c) multiplies by the divisor: 0.2857 * 7 = 2

# integer of the division

Ex.: 100 # 7 = 14

Computing procedure for the result :
a) executes the division: 100 / 7 = 14.2857
b) break up the integer part of the result: integer (14.2857) = 14

? step adjust

Ex.: 100 ? 7 = 7.14285

Computing procedure for the result :
a) executes the division: 100 / 7 = 14.2857
b) break up the decimal part of the result: 14.2857 - 14 = 0.2857
c) multiplies for the divisor: 0.2857 * 7 = 2
d) divides the module by the integer of the division: 2 / 14 = 0.14285
e) adds to the divisor: 7+ 0.14285 =7.14285

The *a)* to *c)* points perform the module computation.
If the result of division is less than 1, this function returns a dividend.
Ex.: 10 ? 15 = 10 {results: 10/15= 0.6666}

The **?** function then returns a correct divisor, modified as to obtain an integer result from division.

& AND

Ex.: 10.456 & 3.56 = 2

| OR

Ex.: 10.456 | 3.56 = 11

Boolean bit to bit AND / OR operate only on the operand interger part.

A *"Division by zero"* error occurs, in the / , % , # and ? operations, when a zero denominator is detected.

Parenthesis may be nested without limits. Ex.: "l2*((r0+r3)*(r4-r1))".

In some cases, the multiply character '*****' may be neglected since implicitely assumed, as:

...)(...	equivalent to	...)*(...
...)3...	equivalent to	...)*3...
...)l...	equivalent to	...)*l...
...3(...	equivalent to	...3*(...
...l(...	equivalent to	...l*(...

3.6.11 Operating Limits

- In the piece variable assigning (i.e. r40 variable):
  a) the Face Dimensions parameters (lf, hf and sf) cannot be used
  b) face number "face" cannot be used
  c) In the current variable expression, only the previous "r" variables (in this case from r0 to r39) may be used
  d) No Functions or Variables, including a "j" variables, may be used

- In the work parameters assigning, when inserted on a face of a program text and in the "r" variable assigning into a Subroutine (always in the text of program):
  a) the (lf, hf and sf) arguments assume always the current face (in programming) dimensions.
  b) The *minr, maxr, aver* and *sumr* Functions cannot be used
  c) No Functions or Variables including a "j" variables may be used.

3.7 Subroutines

3.7.1 Definition

A subroutine is a normal part program usually defining a recurrent works layout, typical of the User applications; then it may be convenient to be stored separately to be recalled and embedded in a general "main" programs, when required. Consequently:

- it appears on the programs directory
- may be run in execution by itself
- writing and/or modifying is not protected by password

The works instructions sequence, included in a subroutine, may have:

- a completely fixed structure, without a parametric programming needing
- a parametric structure, as to be adjusted, in some geometrical or technological feature, by a reassignable variables passed by the recalling main program.

The parametrical subroutines using is the most general and interesting mode for a structured programming.
Edicad manages in a fully automatic mode the subroutine parametric programming process, as we can see in the **[Rnn]** typology button description.
In the case a subroutine, already stored and recalled in many part program, should be heavily modified in its variables structure, User is suggested to verify the previous subroutine using and to save again the interested programs, after compiling.
A particular case of the parametric programming includes a not assigned (at subroutine level) variables using: in this case the variable value is defined only by the recalling main program.
Example:
if the variable r0 is passed from the recalling program to the subroutine, it will be sufficient do not assign the r0 value in the subroutine text: in this case, when the r0 parameter is referred on the subroutine body, the recalling program original value is automatically assumed, without needing to assign an internal variable to copy the external program variable.

This feature may result very convenient, but is better to limit them only to the absolutely necessary cases. Indeed User may forget to leave undefined the required variables (designed for a direct value transfer from main) in the subroutine text; if locally defined, an heavy alteration of the subroutine behaviour may occurs.
To avoid this error, a good rule may be to document, in a comment string, all the variables meaning.
In the subroutine using, the only constraint is the subroutine and the recalling application face existence.

A subroutine may include a macros.
The subroutine calling logic is not recursive: then a subroutine cannot recall itself!

3.7.2 Development

A subroutine is for all aspects a standard program. It may be executed as written, according to the same interpretation logic used in phase of writing.
Graphic view of a subroutine absolutely respects the working sequence.
Consider for instance a working cycle executing 10 point based workings, with the following data:

a) X position of first reference =r0

b) Y position of all references =r1

c) depth Zp of all strokes =r2

d) incremental pitch for next strokes, to be applyed to axis X=r4

e) point base working code 81

f) tool selection by type (=r5) and diameter (=r6)

Each data is expressed in parametric form, subroutine call allows assignment of data as listed.
All parameters should be qualified as reassignable (r0, r1, r2, r4, r5, r6).
Subroutine codes are of general use and "subroutine variables" are automatically reassigned.

3.7.3 Programming rules

Some rules should be followed for best efficiency in developing and using subroutines:

a) when defining a Subroutine, assign as reassignable all and only the variables that shoulod be modifyed when the subroutine is called;

b) when selecting the variable type, set the most suitable, as follows:

- set a *float* type for a coordinate
- set an *integer* type for a counter

c) comment meaning of variables

d) use non reassignable variables to set data that will probably be reassigned in the future (subroutines will be more easily espanded)

e) use non reassignable variables to set data used as recursive. Consider for instance a subroutine with reassignable variables r0, r1, r2, r3, where point based workings are defined at a vertical position QY computed as an average between the four parmeters. It may be efficient to set a value in a non reassignable variable and use it as current value:
r10=(r0+r1+r2+r3)/4 and set qUOTE y =r10

All subroutine parameters aare computed with a recursive backward criterium, starting from the nearest call level.

Consider the following example:

Prg PIPPO

```
r0=100;w          <-- variable r0:  reassignable, set value=100
r1=200=w          <-- variable r1:  reassignable, set value=200

face 1

[sub] name=sub001 (r0=50)          <-- call subroutine "sub001", face 1, assigning variable r0=50
```

Subroutine **SUB001** has the following text :

```
r0=80; w          <-- variable r0:  reassignable, set value=80
r1=...            <-- variable r1: NON ASSEGNATA
r5=r0/2;          <-- variable r5:  NON reassignable, set value=r0/2

face 1
```

```
[81] qx=r0; qy=r1
[87] qx=r5;
```

Program PIPPO solves :

```
face 1

[sub] name=sub001  (r0=50)              :..
       :[81] qx=r0=50;  qy=r1=200;
       :[87] qx=r5= 25;
```

where:

```
working [81]:          q.x assumes value r0 assigned callig 50;
                       q.y =r1 reads r1 from PIPPO ==> qy=r1=200

working [87]:          q.x assumes r5 from SUB001, assigned with r0 valid (50) ==> qx=r5=r0/2=25.
```

3.7.4 How to recall a subroutine

A subroutine call *inserting* in a main program may be performed or by the Works list {*Apply / Works*} or by the {*Apply / Subroutine*} command: in the two cases, the *Subroutines List* is accessed including all the available codes:

- From 2010 to 2019: special codes for a graphic selection of a subroutine. These 10 codes may be used to customise the subroutine calling modality: for instance, a code may be dedicated to a call with translation, another code for a call with translation and rotation, and so on.
- From 4001 to 5000: for a direct subroutine calls.

Subroutines are stored in specific environment, defined by the "dirprod" item in the [TPA] section of the"tpa.ini" file, with the sub-addressing "\SUB" item. The subroutine name assigning allows, in any case, to define a different addressing path, as examined in the documentation regarding parametric programming.

3.7.5 How to set the call for a subroutine

The dialog box, to program a subroutine calling, may require some different data, according to the subroutine calling code: normally, in any case, the following information must be entered:

- *Execution conditions:* include a logical test between two expressions as to force a logical conditioning to the subroutine execution.
- the *subroutine Name* (box: *Sub*); including a special **[>>]** button, as described later.
- *X1, Y1, Z1* are the positions of a translation positioning, in absolute or relative mode, to define the real application point along the X, Y and Z axes.
- *rel,* to enable the relative positioning.
- *Face* define the number of the face which the subroutine may be used on: if missing, by default the face N. 1 is assumed. The setting is different for "Subroutine induced calls", if they exist in the system, and it is as follows: indicate the subroutine face number to be used; if it is not indicated, for each subroutine face with programmed workings a call is applied to the current program. The subroutine induced calls are described in a paragraph below.
- ° defines the rotation positioning angle:
  a) any value may be programmed
  b) rotation may be applied only on the XY plane
  c) with an arc programmed on a plane different from XY, the rotation will be accepted only if, between the works list, the "arc on XYZ plane" code is available.

- *rel* **|<-** if selected with active **rel** parameter and preceding a macro or sub-program: the application point is referred to the initial application point of previous working.

Between the geometrical transformations for application, we have:

- *Locking point*: if enabled, in the subroutine the works origin is forced on the previous application point. If the Locking parameter is enabled (value= 1):
  a) it prevails on the translation positioning by the P1 (X1, Y1, Z1) point
  b) it links the following milling path with the previous, neglecting the internal Setup instruction.

  In this last case, the tool rising up and down movements, on the locking point,
  are erased.
  For the milling paths junction, with the consequent setup erasing, the following conditions are required:
  a) the subroutine must start with a Setup instruction (or with an application code followed by a setup)
  b) the previous work (immediately before the subroutine call) must be included in a profile (then or a Setup or an arc or a linear segment), also if it's a last instruction of a macro or another subroutine
  c) the current subroutine and the previous work must be subjected to the same logical conditions
- *Mirror X* and *Mirror Y:* to be enabled if a mirrored execution is required.
  With the mirror x or the mirror y (not x+y) requirement, in the subroutine developping, all the tool correction senses are reversed (right <--> left).
- *reverse*: if activated, reverses the execution of a program.

Command buttons: **[>>]**, **[Rnn]**, **[OK]** and **[Abort]**.

- The **[OK]** button allows to confirm all the entered data, exiting from the dialog window.
- The **[Abort]** allows to exit, aborting all the entered data
- The **[>>]** button, on the side of the subroutine name (*Sub*) box, allows to select the required subroutine from the usual *File Open* window, then researching them between all the available drive and directories. In the case of a standard searching directory (dirprod+\"SUB" and its subaddressing), the pathname is presented, in relative form, on the *Sub* box, to warrant a stores portability.
- Excluding the *modifying* or *inserting* session, the **[>>]** button selection causes a complete subroutine pathname displaying: this may be particularly convenient in the case of the subroutine name parametric programming.
- The **[Rnn]** button allows to enter ad/or verify the reassignable variables used on the subroutine.
  In the case of working instruction *modifying* or *inserting,* the **[Rnn]** button selection opens a special window including all the variables may be reassigned on the current subroutine.
  This command has no effect, in the case of:
  a) reassignable variables list void
  b) subroutine pathname unassigned or incorrect.

  Variables are listed with the following data:
  a) the variable name (i.e.: r10)
  b) the typology (i.e.:. float)
  c) its stored expression as loaded from subroutine, in the case of insert, or the last entered, in the case of modifying mode.
  d) the comment

  The assigned value, for every variable, is not displayed.
  Every variable must be confirmed, in numerical or parametric format, without missing fields.
  Excluding in the *modifying* or *inserting* mode, with the **[Rnn]** button selection, the used variables are listed, in this case with their computed value.
- Button **[REPEAT]** gives access to the dialogue window for the parameters required by sub-program repetition.

Parameters relate to *Repetition* depend on one of the two repetition modes defined for the sub-programs, i.e.:

a) repetitions with free allocation
b) repetition with matrix allocation

3.7.6 Subroutine with Repeat with free allocation

In case of **Repeat with free allocation** the dialog window contains:

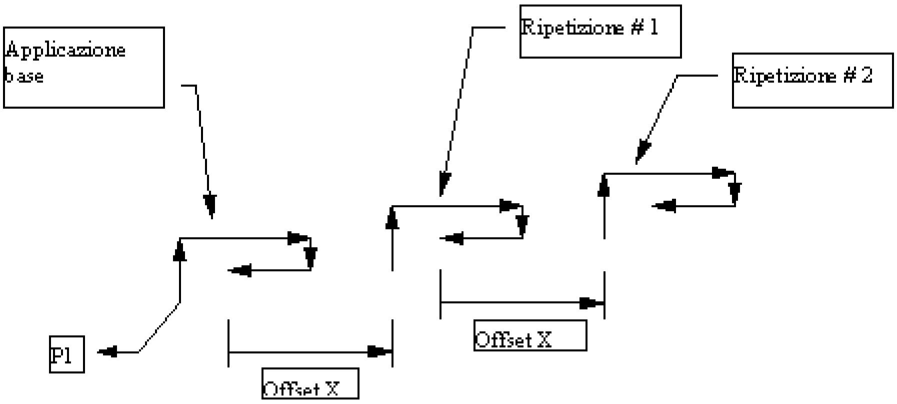
- *Repetitions* number of repetitions to be executed in addition to basic execution.
- *Offset X* offset positioning coordinate to be applied to X at each repetition (applied in relative mode).
- *Offset Y* offset positioning coordinate to be applied to Y at each repetition (applied in relative mode).
- *Offset Z* offset positioning coordinate to be applied to Z at each repetition (applied in relative mode).
- *rel* |<- when selected, offsets are applied to initial point of previous repetition.
- *Link point* when selected links each repetition to the final point of the previous.
- *dA°* angular increment at each repetition.
- *Button* **[PGUP]** returns to previous window.

Application rules:

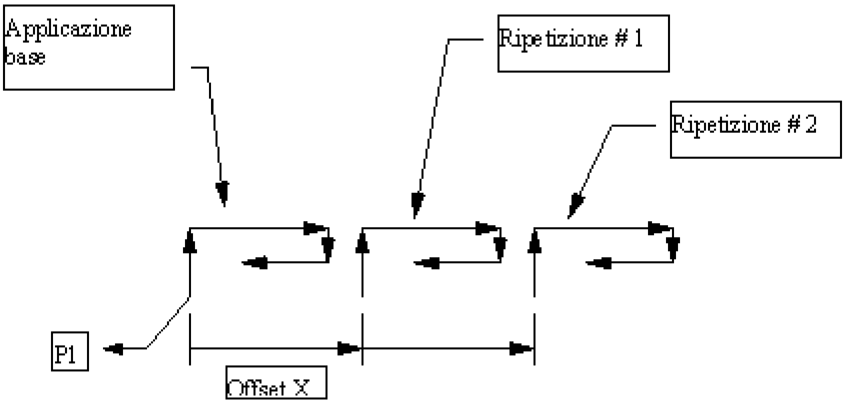
a) minimum value of repetitions is 1;
b) angular rotation increment *dA°* is added at each repetition, starting from the initial value assigned in *A°* (first window of working). Example: repetition # 2 has a rotation of (*A°* + 2 * *dA°*);
c) mirrorings assigned to basic application are also applied to repetitions (x mirror: the application of *Offset X* is inverted; y mirror: the application of *Offset Y* is inverted).

Example :
Repetitions=2
Offset X=100 Offset Y=0
dA°=0

The figure represents the sequence generated by setting *rel* |<- = 0.
P1 is the basic application point.
Total number of applications is: 1+2.



The figure shows the development resulting from the active *rel* |<- setting :



3.7.7 Subroutine with repetition with matrix allocation

In case of **repetition with matrix allocation** the dialog window contains:

- *Lines:* number of lines of matrix.
- *Columns:* number of columns of matrix.
- *Offset X:* offset positioning co-ordinate to be applied to X at each repetition (applied in relative mode).
- *Offset Y:* offset positioning co-ordinate to be applied to Y at each repetition (applied in relative mode).
- *rel* |<- *:* when selected, offsets are applied to initial point of previous or reference repetition.
- Button **[PGUP]** returns to previous window.

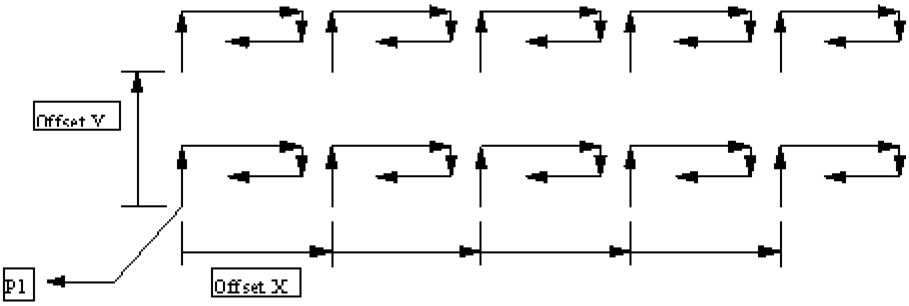
Application rules:

a) repetitions are applied only if no inversion of subprogram is requested;
b) minimum value of repetitions is 1;
c) rotation and mirroring assigned to basic application is also applied to repetitions (x mirror: the application of *Offset X* is inverted; y mirror: the application of *Offset Y* is inverted).
d) Executes a number of applications equal to *Lines*Columns*, including the basic application.

Example:

Columns=5	Lines=2
Offset X=100	Offset Y=80

The figure represents the sequence generated by setting *rel* |<- active.
P1 is the basic application point.
Total number of applications is: 5*2 (basic application is enclosed in the repetition cycle).



3.7.8 Subroutine induced call

When recalling a subroutine without assigning a face, even the workings programmed in the subroutine for the other faces of the piece are automatically applied to the corresponding faces. This mode of application by the subroutine is called **Subroutine induced call**, and it is only active if the system is set for it.
This type of call respects the following rules:
a) they are queued after the workings of the relevant faces, consequently the workings which are directly programmed on the face can only be inserted above the induced calls;
b) they cannot be modified directly on the face;
c) they are modified automatically by modifying the main call;
d) if the main call has a non verified logical conditioning, even the induced calls are not carried out;
e) in the application of the induced calls, the placing values X_1 and Y_1 are only applied if there is a physical correspondence on the piece between the two axes;
f) in the application of the induced calls, the *relative* and *point anchoring* selections are deactivated .
g) the induced calls are also applied to the fictive faces.

3.8 Drawing Functions

3.8.1 Foreword

The Editor program provides various functions for entering and modifying drawings.
A drawing is a set of geometric elements such as: points, segments and arcs, polylines and splines.
A polyline is an ordered sequence of arcs and segments preceded by an initial point.

A drawing may be introduced in two basic ways:

- by digitising
- by construction

3.8.2 Typing in

You can obtain a drawing by typing in a sequence of points.
A point is characterised by geometric dimensions which indicate its position in space and by typological dimensions which describe the relation between the points and impose conditions to the curve described by them.
In order to describe arcs and lines the points typed in must contain typological information called "point typology".
The drawing function is enabled from the ***Applie->Profile*** menu.
The commands in the ***Draw*** menu are activated***.*** To end the typing in session of a profile select the command a second time.
The profile starts with an initial point and can combine and alternate angular points with normal points. It ends when an isolated point is entered or an initial point which begins a new profile.

The machining operations entered with this command are queued behind the last machining operation on the face. In this mode, the options in the View menu stay active, and the *Grid* option is the most important with the possibility of SNAP.

Supposing, for simplicity's sake, that the data entry devices are the keyboard and the mouse, as soon as the command is given, the dimensions of the first point of the new profile can be acquired. In fact, the video still shows the window with the face on which the machining operations are to be applied. The dimensions of the mouse appear on the *Toolbar* and the type of the next point to be entered is highlighted on the same, in the first of the four graphic indicators.

You can change the point type, according to the rules given below, with the command {*Draw / Type*}; the possible types are included in the options available with this command.
You can take a shortcut to choosing the type by using the Drawing Palette, enabled with the {*View / Drawing Palette*} command.

If you don't change it, the type of the next point is selected and propagated automatically.
If the *Alternated point insertion sequence* {*Set / Drawing aids (Graphic settings)*}is enabled, an ANGULAR point follows a NORMAL point or an ANGULAR point.

To confirm the insertion of the point, move to the position selected and press the left mouse button.

Once you have inserted the point, according to the type, the geometric entity associated with it will appear. A marker can then be displayed, in the case of an isolated point, or, otherwise, a segment or an arc. In profile entry mode, geometric machining operations are stored for:

- Point machining operations
- Set-up machining operations

while machining operations are stored for:

- Linear machining operations (L01)
- Arc machining operations (A01 or A04)

All these machining operations are identified with level 0. Following each machining operation of a geometric type, you can assign a technology with the command {*Edit / Assign technology*}.

3.8.3 Construction

Construction is the typical methodology of a CAD system where points, lines and arcs are defined with absolute dimensions or as particular geometrical locations such as intersections, tangents, offsets, and so on.

In the *Draw* menu four options are highlighted: POINT, LINE, ARC, CIRCLE corresponding to the geometric entities to be entered. Each of these options has further options corresponding to the ways in which these entities can be defined and built.

For all the options you can identify some common operations.

Positioning and selecting a geometrical entity are always subordinate to the type of positioning enabled at that moment.
The various types have been described above, but they are recalled briefly below:

- **Mouse:** the geometrical entity is positioned and/or selected by means of the mouse
- **Cartesian:** the geometrical entity is positioned in a Cartesian dimension entered by means of the keyboard and is selected with its progressive number.
- **Polar:** the geometrical entity is positioned in a polar dimension entered by means of the keyboard and is selected with its progressive number.
- **Point with Mouse:** the geometrical entity is positioned at the same height as the machining operation application point closest to the mouse position; it is selected with the mouse.
- **Progressive point:** the geometrical entity is positioned at the same height as the machining operation application point indicated, by means of the keyboard, with its progressive number; it is selected with its progressive number.

4 Tools

4.1 General tools

4.1.1 Introduction

The general tools operate on the current working or on the selected ones. If the working belongs to a profile (that is, a setup, arc or linear working), the tool is applied to the whole profile .

4.1.2 Translation

Translation of the working is enabled from the ***Tools->Translate*** menu .

This command executes a translation of the selected workings or the current working.
The translation is assigned according to the positioning mode on the Selections bar which is active at the time.

- **positioning with the Cartesian system**: the x, y, z coordinates are assigned in the dialog box in absolute or relative mode.
- **positioning with the polar system**: the x, y centre coordinates and the z coordinate are assigned in the dialog box in absolute or relative mode. The values of the module and the angle are assigned in absolute mode.
- **positioning with the mouse**: the movement only involves the absolute x and y coordinates, which obtained by simply pointing the mouse.

In the event of absolute translation in position, the current block is moved to the assigned position. The remaining blocks involved then translate as well.
In the event of relative translation, all the blocks involved are moved by the indicated offsets, while the parametric programming of the modified coordinates are maintained.
In the event of point workings, subroutines or macros:

- the translation does not involve changing the application coordinates in the event of relative programming.
- for subroutines or macros: the point of application is translated, even on the empty set coordinate/s.

4.1.3 Horizontal centring

The centring of the working is enabled from the ***Tools->Center->Horizontal centring*** menu .

This command translates the selected blocks or the current block along the x axis, until the x coordinate of the selected working is positioned half way along the length of the panel.

4.1.4 Vertical centring

The centring of the working is enabled from the ***Tools->Center->Vertical centring*** menu

This command translates the selected blocks or the current block along the y axis, until the y coordinate of the selected working is positioned half way up the height of the panel.

4.1.5 Rotation

Rotation of the working is enabled from the ***Tools->Rotate*** menu.

This command rotates the selected workings or the current working.
If the working belongs to a profile: the profiles, the subroutines and the macros in point anchoring mode are not rotated.
The centre of rotation is assigned to the plane of the face (xy) through absolute or relative

programming. Relative programming indicates the movements to be assigned with respect to the current working.
The angle of rotation can also be assigned in absolute or relative mode.
Rotation is not valid if the absolute angle and centre coincide with the current point of application.
An arc in the xy or yz plane cannot be rotated.
With these settings only the special subroutine case is solved: in this case the final angle of rotation is taken from the subroutine call rotation parameter.
For subroutines or macros:

- rotation is applied to the application point, even for the non set coordinate/s.
- it applies the rotation assigned to the angle of rotation.

4.1.6 Symmetry

The mirror axis is assigned to the plane of the face (xy), according to the active positioning mode in the Selection bar:

- **positioning with Cartesian or polar system:** the coordinates which define the axis of symmetry are assigned in absolute mode in a dialog box.
- **positioning with the mouse**: the coordinates of the axis of symmetry are assigned using the mouse pointer on the face.

Before applying the geometric transform, the user is asked if the workings involved should be copied: if the reply is affirmative then a Copy is made and automatically Pasted. The duplication is not made if the command is applied to selections with individually selected profile sections or in the event of a profile with request for point anchoring. The copied workings are inserted below the blocks involved in the command, without ever breaking up a profile. The symmetry is applied to the inserted blocks.
If the working belongs to a profile: the profiles, the subroutines and the macros in point anchoring mode are not mirrored.

The parametric programming of the modified coordinates are maintained.
For subroutines or macros:

- the transform is applied at the point of application, even for the non set coordinate/s.
- for subroutines: it also applies to the mirror parameter/s.
- for macros: it also applies to the assigned parameters with the rule of symmetry .

Horizontal symmetry
The horizontal symmetry of the working is enabled from the ***Tools->Mirroring->Horizontal symmetry*** menu***.***
This command creates a mirror image of the selected workings or the current working along the horizontal axis.

Vertical symmetry
The vertical symmetry of the working is enabled from the ***Tools->Mirroring->Vertical symmetry*** menu***.***
This command creates a mirror image of the selected workings or the current working along the vertical axis.

Generic Symmetry
The generic symmetry of the working is enabled from the ***Tools->Mirroring->General symmetry*** menu.
This command creates a mirror image of the selected workings or the current working along a generic axis.

4.1.7 Repetition

The repetition command of one or more workings duplicates them.
The new workings are inserted on the face according to the following criteria:

- for application to the current profile: below the original profile.
- for application to the current working: below it.
- for application to selected blocks, with the current working selected: below the selected workings.

- for application to the selected blocks, with the current working not selected: below the current working.

In any case: if the application of the rules described above result in invalidating a profile (e.g. point workings inserted in the middle of a profile), the insertion would take place in any case below the profile in question.

The parametric programming of the modified coordinates are maintained.
For point workings, subroutines or macros:

- the translation does not alter the application coordinates in the event of relative programming.
- for subroutines or macros: the point of application is translated, even on the empty set coordinate/s.

Single repetition
The single repetition of a working is enabled from the ***Tools->Single repeat*** menu.

This command makes a duplicate of the selected workings or the current working. The position of the new workings is assigned according to the active positioning mode on the Selections bar

- **positioning with the Cartesian system**: the x, y, z coordinates are assigned in the dialog box in absolute or relative mode.
- **positioning with the polar system**: the x, y centre coordinates and the z coordinate are assigned in the dialog box in absolute or relative mode. The value of the module and the angle are assigned in absolute mode.
- **positioning with the mouse**: the movement only involves the absolute x and y coordinates, which obtained by simply pointing the mouse.

Multiple repetition
The multiple repetition of a working is enabled from the ***Tools->Multiple repeat*** menu.
Copies on the current face the selected workings for a number of times specified by **[On Number of]**. To determinate the workings positions **[Delta X]**, **[Delta Y]** and **[Delta Z]** offsets have to be specified.

4.2 Profile tools

4.2.1 Profile cutting

A profile can be cut from the ***Tools->Cut*** menu ***.***

This command cuts the profile at the indicated point only if the current working and the next working assign profile segments (arc or line).
A setup is inserted below the current working with:

- code and setting obtained from the original profile setup
- application in the current point.

Below the added setup the profile maintains the previous development. Specifically:

- if, originally, the profile path, after the new setup point, was some geometrical linked with the previous path (i.e. continuity of tangent) a new code will be assigned as to erase this link.
- all the linear or circular blocs, assigned for closing on the setup, will be reassigned as to conserve the original path.

4.2.2 Reversing direction

Direction reversal of a profile is enabled from the ***Tools->Reverse direction*** menu.

This command reverses the path of the current profile. A copy of the profile may be requested: in this case the reversal is applied to the profile which is inserted. Reversal of a profile involves reassigning operative codes for the profile segments.
The command is not active for profiles with request for point anchoring.

4.2.3 Move Setup

In a closed profile the Setup can be moved from the ***Tools->Move Setup*** menu ***.***

The user is asked if the Setup point should be moved to the current working; if the reply is negative then a dialog box appears and the x and y coordinates of the point to which the Setup is to be moved are entered.

4.2.4 Profiles Junction

To join two consecutive profiles select the command from the ***Tools->Unit*** menu.

The command is only enabled if the current working is part of a profile and if the previous profile follows it.
The suggested operating modes are:

- translates the profile: the second profile is translated so that the setup is moved to the last point (x,y,z) of the preceding profile. The setup of the second profile is then eliminated .
- inserts a joining segment: a linear segment is inserted between the two profiles: the segment's first point coincides with the last point of the preceding profile and its last point coincides with the setup of the second profile. The second profile's setup is then eliminated.

4.2.5 Linearize an arc

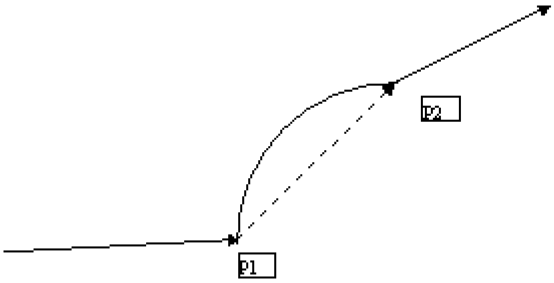
To linearize an arc select the command from the ***Tools->Change arc to line*** menu ***.***

This command allows to substitute a circular arc with a linear segment.

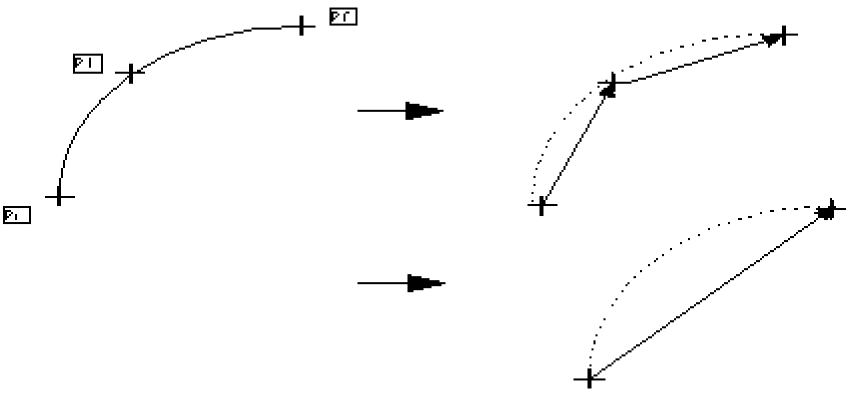
Two different cases must be considered:

- replacement only involves the current block, which must have an arc typology.
  replacement involves the whole profile to which only the current block belongs with an active **Apply to profile** tool, or with the current block which coincides with the setup of a profile, or with a selected profile. In this case the profile must be scanned starting from the setup.

The geometrical result is showed in the figure:



In the drawing, the arc between points P1 and P2 is replaced with a linear segment.
A special case is indicated by the code COPA04: three-point arc. If the rotation of the arc is set in automatic mode (to pass through the assigned point P1): it is possible to linearize it in a single segment or in two segments (from the first point to the intersection point, and from the intersection point to the last point). The geometric transformation is shown in the figure:

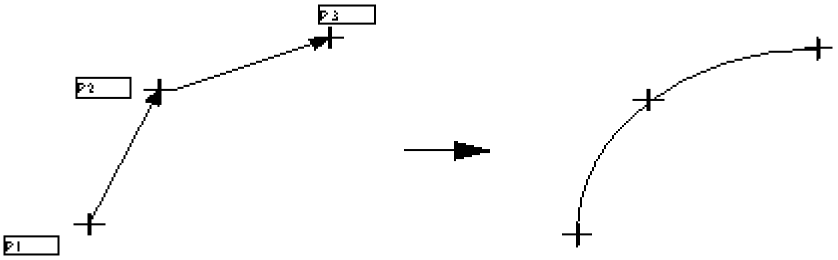


The choice of the type of transform is confirmed by the operating code on the first segment of the COPA04 type profile: three-point arc.

4.2.6 Three-point arc

To change a corner in a three-point arc select the command in the ***Tools->Arc three points*** menu ***.***

This command allows to substitute two consecutive segments with a three-point arc; the operating code COPA04 must be available. The replacement only involves the current block and the next one, which must be the single line type. The geometric transformation is shown in the figure:



The working corresponding to the segment which ends with P2 is eliminated from the list and the working which ends with P3 is assigned :

- a three-point arc code.
- the intersecting point of the arc is P2 and the last point is P3.
- the rotation of the arc remains in automatic mode. The three original points must be distinct and not in line.

4.2.7 Filleting an edge

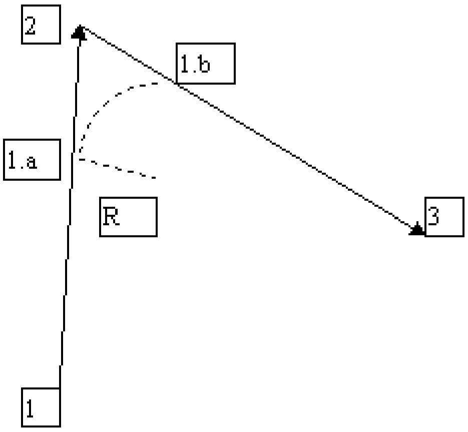
To fillet an edge select the command from the ***Tools->Connect corner*** menu.

This command fillets an edge consisting of two profile segments (applicable to one or more points), by assigning a value to the filleting **[Radius]**.
The second segment can also be assigned as an expanded segment (e.g. chamfer or fillet), as long as the initial segment is linear.

Two different cases must be considered:

- replacement only involves the current block, which must have an arc typology.
- replacement involves the whole profile to which only the current block belongs with an active **Apply to profile** tool, or with the current block which coincides with a profile setup, or with a selected profile. In this case the profile must be scanned starting from the setup.

If the edge consists of two linear segments, the geometric transformation is shown in the figure:



In the drawing, the two original linear segments follow the profile on the points: 1 ,2 and 3.

After the application of the fillet (with radius R), the profile between points 1 and 3 is assigned on the segments :

- 1-> 1.a a linear segment
- 1.a-> 1.b circular segment (with radius R, tangent to the two original segments)
- 1.b-> 3 linear segment

The first two segments are assigned a fillet code (COPA29):

- edge set in (2).
- fillet radius R (assigned by the operator).
- intercepted straight line identified by points (2) and (3).

The next segment (ending in (3)) may undergo changes caused by the need to keep the development of the profile unchanged.

In the generic case of an edge between two non-linear segments (arc - line, line - arc, arc - arc):

- an arc is inserted between the two segments (COPA01) which is tangent to the two segments.
- the first segment of the original edge undergoes changes caused by the need to change the last point.
- the second segment of the original edge undergoes changes caused by the need to change the starting point and to keep the development of the profile below unchanged.

The fillet arc can be assigned a **[Speed].**

4.2.8 Chamfering an edge

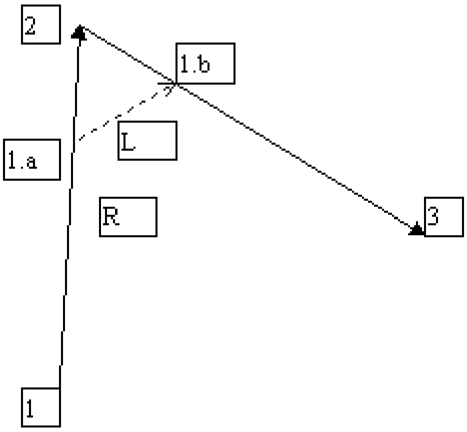
To insert a chamfer on an edge select the command from the ***Tools->Round off corner*** menu.

This command chamfers an edge between two profile segments (applicable at one or more points), by assigning the **[Length]** value of the segment to be chamfered.

The segment inserted can be one of the following working types:

- COPL18: solves the case of a detached assigned length on the original linear segments.
- COPL19: solves the case of assigned length for the chamfer.

The next figure shows the geometrical result:



In the drawing, the two original linear segments follow the profile on points: 1 2 3. After application of the chamfer, the profile between points 1 and 3 is assigned on the segments :

- 1->1.a a linear segment
- 1.a-> 1.b a linear segment
- 1.b-> 3 a linear segment

The first two segments are assigned with a chamfer code (COPL18 or COPL19):

- edge set in (2).
- length segments (1.a 2) and (2 1.b) or segment length (1.a 1.b) as assigned by the programmer.
- intercepted line identified by points (2) and (3).

The next segment (ending in (3)) may undergo changes, caused by the need to keep the development of the profile unchanged.
The chamfer segment may be assigned a **[Speed]**

4.2.9 Apply entry

To insert an entry segment in the current profile select the command from the ***Tools->Insert start block*** menu.

This command allows to insert an entry path into the current profile.
The command is not active for profiles with a request for point anchoring.
The type of path may be selected between:

- Cartesian linear segment: the x and y coordinates for the segment start are assigned with relative or absolute positioning and z variation along the path;
- polar linear segment: the length of the segment, the angle and the z variation on the segment are assigned. For the angle, a value is proposed corresponding to the final tangent of the first segment of profile.
- circular segment: the x and y coordinates of the first point are assigned (in absolute or relative) and the z variation on the segment. The arc is made as a tangent with the programmed start of the profile.

4.2.10 Apply exit

To insert an end segment to the current profile select the command from the ***Tools->Insert exit block*** menu.

This command allows to insert an exit path to the current profile.
The command is not active if below the profile there is a request for point anchoring.
The type of path may be selected between:

- Cartesian linear segment: the x and y coordinates for the segment end are assigned with relative or absolute positioning and z variation along the path;
- polar linear segment: the length of the segment, the angle and the z variation on the segment

are assigned. For the angle, a value is proposed corresponding to the final tangent of the last segment of profile
- circular segment: the x and y coordinates of the arrival point are assigned (in absolute or relative) and the z variation on the segment. The arc is made as a tangent with the programmed start of the profile.

4.2.11 Linearize Z

To activate the Linearize the Z variation select the item from the **Tools*->Make Z linear*** menu.

This command makes it possible to apply a linear distribution of the Z depth included between the two profile coordinates. The variation could also be assumed automatically by the profile coordinates **[Assume coordinates from Profile]**: in this case the z depth of the setup and the final segment of the profile are maintained, and the linearization is applied between these two extremes.

4.2.12 Bloc dragging

To drag the working select the command from the ***Tools->Drag*** menu***.***

This command drags the current working or the selected workings to another position on the face. The overall xy area defined by the blocks involved in the translation, is indicated on the screen by a rectangle with dashed edges and the pointer changes into a hand. To move the blocks, position the pointer inside the area, press and hold the left mouse button as you drag the area to the new location, then release the mouse button.

4.2.13 Direct Measure

To carry out a direct measurement between two points select the command from the ***Tools->Measure->Directe*** menu***.***

A segment appears on the screen with one end fixed to the point of application of the current working, whereas the other end is dragged by the movement of the mouse. Choose the point at which to establish the distance between the two points, displayed in a box that contains the following information:
- **Distance**: measures the direct distance between the two selected points.
- **Delta X**: variation of the x coordinate between the first and second selected point.
- **Delta Y**: variation of the y coordinate between the first and second selected point.
- **Delta Z**: variation of the z coordinate between the first and second selected point.
- **Inclination**: angle of inclination, expressed in degrees, formed by the line that joins the two points.

4.2.14 Measure along the profile

To carry out a measurement of a profile segment select the command from the ***Tools->Measure->On profile*** menu.

The item is only enabled for the workings belonging to "profiles".
The command acts differently depending on the type of positioning that has been set:
- **positioning with mouse:** a segment appears with one end fixed to the point of application of the current working, whereas the other end is dragged by the movement of the mouse. Choose another point on the profile to establish the distance between the two points along the curvilinear coordinate, which is displayed in a window.
- **Cartesian positioning:** a box appears showing the progressive number of the profile segment to be measured. Once the segment has been selected, a box shows the distance.

4.2.15 Filleted path

A filleted path can be constructed by means of the command ***Build->Smoothed path***.

The command is not active if the profile has segments which are not correctly defined.
A new path is constructed by using the current one as a base and applying to it the requested milling-radius correction value.
Firstly a box appears to allow the user to enter the construction data, that is :

- **Radius** is the correction value.
- **Contouring** if activated, the profile is constructed without fillets; otherwise it inserts the fillet arcs.
- **Right** if activated, the profile is constructed with a RH correction; otherwise with a LH correction.

4.2.16 Break up a profile

A profile can be broken up by means of the command ***Build->Massimized path***

This command breaks up the segments of a profile into segments of assigned maximum lengths. It is applied to the current block or to the profile.
The command is not active if the profile has segments which are not correctly defined.
Set the following in the dialog box:

- **Max length of segments:** set the maximum length of the profile pieces.
- **Apply length in 3d:** if selected, evaluates the length of the xyz segment, otherwise only for the xy plane.
- **Distributes the remainders**: if selected, calculates the number of pieces in which to break up each individual segment and distributes the remainder over these .
- **Linearizes the arcs:** if selected, breaks the arcs up into sections that are linearized. Arc only If selected it only applies the command to the arc segments; it only appears if applied to the profile.

4.2.17 Minimize a profile

To minimize a profile select the command ***Build->Minimized path*** menu.

This command minimizes a profile by evaluating the geometric continuity of all the consecutive segments. It is applied to the current block or to the profile.
The command is not active if the profile has segments which are not correctly defined.
Enter the **[angle]** value of the maximum tolerated angulation between consecutive linear segments in the box which appears on the screen. The command is not applied to the segments: arc in non-xy plane, circles, expanded workings. The minimization is carried out equal to z. The maximum tolerated angulation angle is assigned: consecutive linear segments which fit into the space between a cone with the assigned maximum angle are joined together. Consecutive arcs which belong to the same arc (same: centre, rotation, sum of paths that do not overlap) are joined together.

4.2.18 Insert connections

To insert connections in a profile select the command from the ***Build ->Path with connections*** menu.

This command inserts connections in a profile in segments assigned maximum lengths. These are generally applied to through closed profiles (the tool passes through the thickness of the piece) to prevent a part of the panel from detaching during the work cycle. The command is applied to the profile to which the segment belongs. It is not applied if the profile has invalid segments or arcs in a plane other than the xy plane.

The connections consist of a segment (linear or circular) in which the tool:

- rises to the indicated depth (leaves a residual thickness)
- executes a working segment (linear or circular), equal to the assigned length

- goes back down to the programmed original depth
- proceeds to the next connection.

The first box allows you to choose the distribution mode for the connections:

- manual to distribute the connections by pointing the mouse in the position in which they are to be inserted.
- automatic for a homogeneous distribution of the connections (specifiy the number) on the profile.

The second window is for entering other relevant data required for the completion of the command:

- **x and y coordinates** set the position of the first connection to be inserted (only in manual mode and with cartesian positioning active).
- **Number of connections** number of connections, max. 255 (only in automatic mode).
- **Length of the connection** set the length in the xy plane.
- **Residual thickness** set the thickness that the tool leaves in the piece when executing the connection. A positive value is accepted, the maximum value possible is equal tot he value of the face thickness. If for example the thickness of the face is 15 mm, with a residual thickness of 3 mm the z coordinate of the connection is calculated as : -(15+3) = - 12 mm.
- **Tool compensation** if selected, modifies the kength of the connection in order to take into account the overall dimension of the tool, indicated on the profile setup; if the compensation exceeds the tool diameter it is considered an error .

5 Errors

5.1 Introduction

The term ***error*** refers to the incorrect use of parameters identified by the processor during the compilation of the piece. This situation cannot be remedied automatically so the processing freezes. The errors include:

- invalid operating code (that is, it is not present among the *available workings*)
- inexistent subroutine or macro
- subroutine or macro with inexistent application face

The term ***warning*** refers to a warning that the compiler provides for the user, in the event of incorrect programming of some parameters which are not compatible with the geometry of a working or even an incorrect parametric programming. The compiler solves these incongruencies with defaults, therefore these *warnings* are not considered errors and they do not interrupt the normal operation or running of the program in the machine.

5.2 Parametric programming error list

In the Parametric Programming, an incorrect parameters or expressions use may cause a different errors typologies: in the following table all the possible errors are listed, classed with their *Number* code, the relative *Message* and the possible *Origin.*

Message	**Origin**
61: Too long String	The max limit has been overcome, that is to say: • 200 characters for the "string' type parameters defining • 50 characters in the other cases
62: Invalid synthax	Any type of synthax erros
63: Division by zero	• operators of division • function: inv (Reverse)
64: Negative Square Root	Function: sqrt
65: $ type Variable used to code a numerical variable	Variables or parameters of numerical typology
66: Flag Variable flag used in the parametric expression	Variables or parameters of numerical or string typology
67: Unassigned variable use	Use of a String type variable or parameter not assigned
68: Incorrect definition of a $ Parameter	Variable or parameter of string type (a not $ type variable has been used)
69: A max number of reassignable variables overcome	Variable "r" of program
70: Value of sine or cosine non included between -1 and +1	Functions: asin, acos
71: Result exceeding the float range	The final result of a parametric expression final exceeds the max float limit
72: Incorrect indirect $ Parameter parametric expression	String type variable or parameter, in the case of a numerical "r" variable addressing
73: The used parametric expression is invalid for a reassignable variable	Parametric assigning of an "r" variable of program or macro, with: • reassignable variable entered • param. expression dependent from an "r" variable
74: Variables $(nn) unusable on a program text	$ Variable or addressing to a $

	variable not in a macro working parameter
75: Variables j(nn) unusable on a macro text	"j" Variable or addressing to a "j" variable not in a program or subroutine working parameter.
76: Invalid variable addressing	Numerical variable or parameter, with invalid (pr, pj, p$) addressing
77: Variables j(nn) not usable on "r" type variable	Variable"r" of program
78: $ Parameter with too many *rnn (maximum 5)	String variable or parameter
79: $ Parameter $ with incorrect *rnn and/o *pr[.] using	String variable or parameter
170: Too many operands Function (max 30)	Functions with a variable number of operands: (min, max,..)
171: Function without operands	Functions multi-operands
172: Function with incorrect operands number	Functions with a fixed operands number: (minr, ...)
173: Invalid "r" variable index	• invalid using range for "r" var. (range: 0÷299) • on a string variable or parameter, with invalid "r" variable addressing (range: 0÷299)
174: Invalid "j" variable index	invalid using range for "j" var.(range: 0÷99)
175: Invalid $ variable index	invalid using range for $ var. (range: 0÷99)
176: Invalid exponent in a power	Function: pown
177: Cumulative Functions on "r" variables not accepted	Functions: minr, maxr. aver, sumr, used not for a "r" variable assigning
178: Reassignable variable range overcome	"r" Variable of program: invalid reassignable variable index

5.3 Program Errors

The following table resumes all the possible errors messages may occurs during a program compiling, including the error *Number*, the related *Message* and the possible *Origin.*
If only the error Number is listed, this error type is not yet used: otherwise, if the Origin colomn is marked by the [*] character, this error has been present but not yet managed in the current release.

Remarks :

[Notice 1] The errors detected during the text reading, cause the program opening with the same conditions as a new: then clearing all the variables, works and forcing an elementary geometry

[Notice 2] The error message occurs only in the "general piece" session

[Notice 3] The error condition must be removed, otherwise the program cannot be saved

Message	**Origin**
60÷79	Errors generated by the Parametric programming
80: Error in the program variables elaboration	[Notice 2]
81: Error in the face 'nn' compiling	[Notice 2]
82: Error in the logical conditions compiling, on the face 'nn'	[Notice 2]
83: Error in the contraints verifying, on the face 'nn'	[Notice 2]
84: Error in the face 'nn' optimization	[Notice 2] not detected by the *Edicad* program
85: Error in the Tool radius correction on the face 'nn'	[Notice 2]. [*]
86: Error in the special controls, on the face 'nn'	[Notice 2]. [*]
87: Fatal Error in the face 'nn' compiling	[Notice 2]. [Notice 3]

	• invalid operating code (not present): (see also Error 92 on the face) • insufficient memory for a correct program interpretation: (see also Error 101 on the face) • subroutine or macro code elaboration failed for an incorrect correspondence (invalid pathname or face selection, or failure in reading)
88: Default Setup inserted on a profile	A profile, not headed by a Setup, has been corrected by an automatic defaul coded Setup inserting
89: Max Number of works in the list	The list of works of the face cannot be enlarged (max limit: 30000)
90: Incorrect Drawing assigning	[*]
91: Incorrect Profile assigning	In elaboration, in the case of subroutine o macro including a profile not headed by a setup instruction
92: Invalid # code	[Notice 3] In the message, # is substituted by the working code
93: Parameter #: incorrect synthax	[*] In the message, # is substituted by the parameter nam
94: Parameter #: invalid (%) value	[*] In the message, # is substituted by the parameter name and *%* by the real value
95: Parameter #: too long	[*] In the message, # is substituted by the parameter name
96: Number of ENDIF greater than IF	Logical conditions instructions: verify if every IF statement has been close by the corresponding ENDIF instruction
97: Number of ENDIF less than IF	
98: Invalid Code following an open IF statement	Logical conditions instructions: verify the typology of the work following the open IF statement
99: Invalid face Number	In the subroutine or macro compiling: an invalid face number (<1 or >32) is detected
100: Parameter #: enter a value between %1 and %2	Incorrect entering of: • level (0 and 8); • origin (0 and 3). in the message, # is substituted by the parameter name, *%1* and *%2* are substituted by the range
101: Impossible to expand the work by insufficient memory	[Notice 3]
106: Technological Priority # not linked	In the message, # is substituted by the technological priority code
111: Radius computed infinite	Work on a profile: an infinite value for the radius has been produced. Codes: 2204, 2205, 2206, 2207
112: Invalid entry Tangent	Work on a profile: evaluation of the input tangent on a point
113: Invalid exit Tangent	Work on a profile: evaluation of theexit tangent on a point
114: Invalid final Point	[*]
115: Null Radius	Work on a profile: evaluation of null radius
116: Invalid Arc	Work on an arc type profile: incorrect or insufficient geometrical conditions for arc defining
117: Invalid intercepte line	Work on a profile: missing or

	incorrect intercept line defining.
118: Evaluated point external to the strokes	Work on an extended profile (linear or circular smoothings): solution is external to the reference strokes.
119: Intersection point missing	Work on an extended profile (double arc): no real solutions in the two arc intersection.
121: Invalid Arc: not distiguishable points	Work on an arc type profile: incorrect or insufficient geometrical conditions for the arc defining, owing to a not distinguishable points
122: Invalid Arc: aligned points	Work on an arc type profile: incorrect or insufficient geometrical conditions for the arc defining, owing to aligned points. Three points arc codes.
125: Invalid subroutine name	Subroutine code processing: all these errors prevent the subroutine application
126: Subroutine coincides with the program	
127: Subroutine # missing	In the message, # is substituted by the subroutine name
128: Invalid subroutine release	
129: Code % cannot be used in a subroutine	In the message, % is substituted by the working code
130: Unassigned Face on a subroutine	
131: Subroutine reading failed	
132: Unassigned subroutine name	
133: Variables reading in a subroutine failed	
139: Invalid macro name	Macro code processing: all these errors prevent the Macro application
140: Macro # missing	In the message, # is substituted by the Macro name
141: Invalid Macro release	
142: Code % cannot be used in a Macro	In the message, % is substituted by the working code
143: Macro reading failed	
144: Unassigned Macro name	
145: Unassigned face in the Macro	
146: Macro variables reading failed	
150: Number of ENDFOR greather than the FOR statements	Logical conditions application: any FOR statement requires a corresponding closing ENDFOR instruction
151: Number of ENDFOR less than the FOR statements	
152: Instruction FOR: unassigned initial expression	Logical conditions application: verify all the FOR instruction assigning.
153: Instruction FOR: unassigned conditional expression	
154: Instruction FOR: unassigned loop expression	
155: Parameter #: enter the $nn format	FOR instruction compiling: requires to assign the # parameter with the indicated formalism. in the message, # is substituted by the parameter name.
156: Parameter #: invalid typology	FOR instruction compiling: signals an incorrect typology assigning for the # parameter. in the message, # is substituted by the parameter name

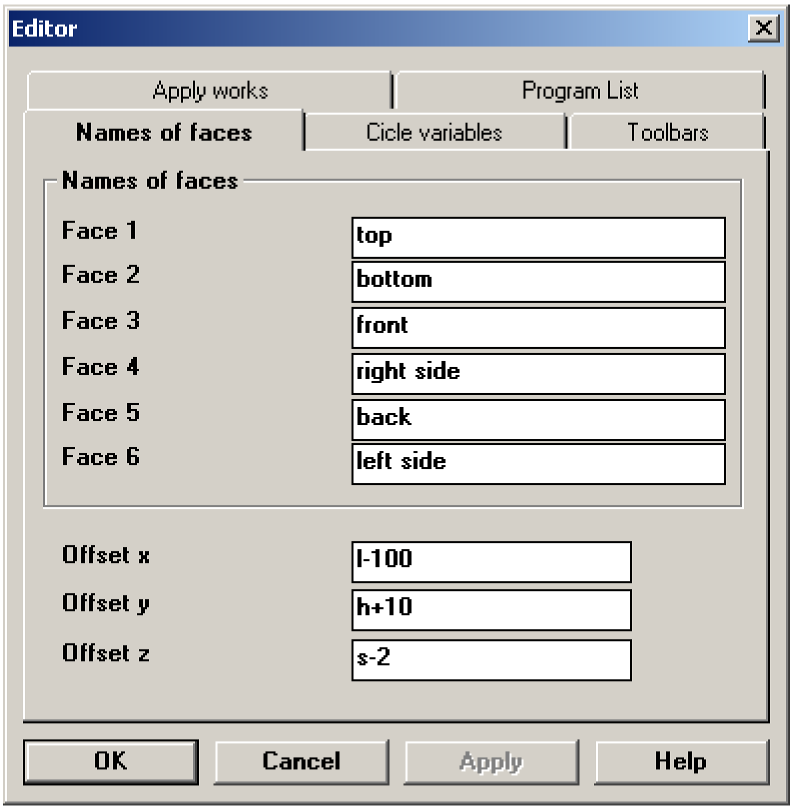
157: Number of FOR instructions exceeding the max value (=50)	Phase of application of the logical conditions in the macro compiling: too may FOR cycles has been detected.
158: Number of iterations in a FOR cycle exceeding the max value (=1000)	Phase of a macro compiling: too many interations in a FOR cycle has been detected
159: Instruction ENDIF used to close a FOR cycle	[Notice 3]. Phase of application of the logical conditions: incorrect closing statement for a FOR or IF cycle.
160: Instruction ENDFOR used to close an IF cycle	
170÷180	Deriving from a parametric programming
181: Unassigned Tool Radius	}
182: The correction exceeds on the arc	}
183: Indefinition in a line-line intersect.	}
184: Indefinition in a line-arc intersect.	}Deriving from a tool radius correction procedure
185: Indefinition in an arc-line intersect.	}
186: Indefinition in an arc-arc intersect.	}
187: Error on corretion on the #xy plane	}
200: Unattended End of file	[Notice 1]. In an ascii format file reading.
201: Closure section $1 not found	[Notice 1]. In an ascii format file reading: $1=name of the section not closed.
202: Geometrical section undefined on the extruded piece	[Notice 1]. In an ascii format file reading.
203: List of face undefined	[Notice 1]. In an ascii format file reading: the text reading ends without the face assigning sections.
204: List of face incongruent with the piece shape	[Notice 1]. In an ascii format file reading. the face sections is not compatible with the declared face number.
205: Number of piece cornersincongruent	[Notice 1]. In an ascii format file reading
206: Number of face corners incongruent	
207: Number of the real faces incongruent	
208: Number of the fictives faces incongruent	
209: Face $1: invalid identifier	In an ascii format file reading: an invalid face section identifier detected
210: Work $1: invalid identifier	In an ascii format file reading: an invalid work code detected in the work section.

6 Settings

6.1 Setting the programming environment

Edicad makes it possible to customize the programming environment both at a graphic level and a data setting level for the execution of pieces on the machine. The dialog box consists of several pages, each of which is used to specify different characteristics. It is activated from the ***Settings->Editor*** menu.
The pages are described below in detail:

Names of faces

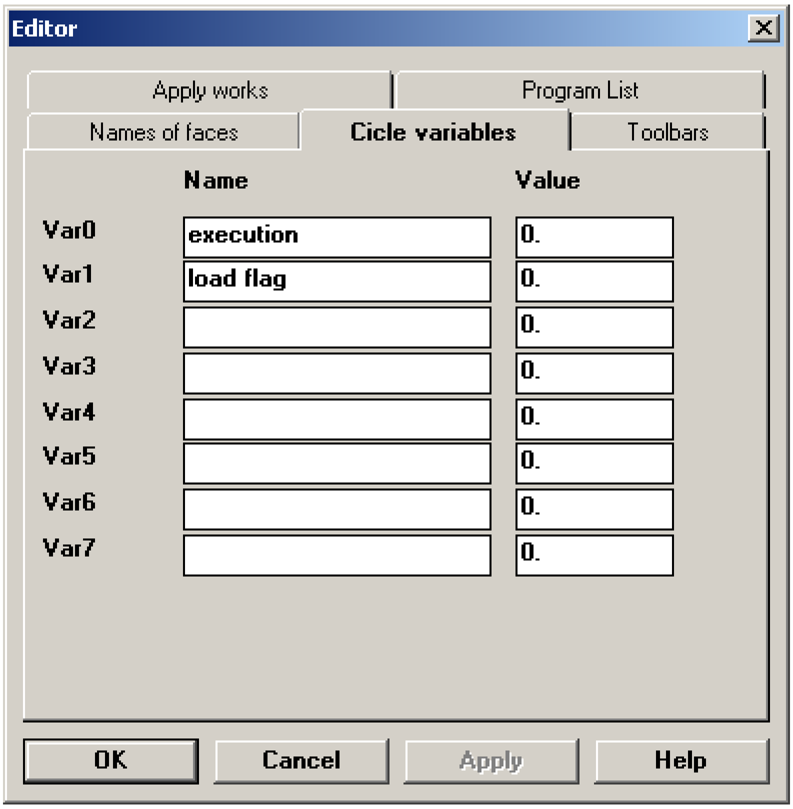


Face 1: top
Face 2: bottom
Face 3: front
Face 4: right side
Face 5: back
Face 6: left side
Offset x: l-100
Offset y: h+10
Offset z: s-2

- **Face 1 to Face 6** are the names of each of the basic faces applied to a new program.
- **Offset X, Y, Z** are the Offsets of the piece which are assigned by default to a new program. They may be changed through the ***Edita->Dimensioni*** menu. The offsets can also be set in numeric type parametric form or according to the dimensions of the piece (e.g.: 100+1). The operators "+ - * /" are accepted; the dimension cannot be a denominator. The calculated value of a parametric expression may be displayed next to the data entry box.

Cycle Variables

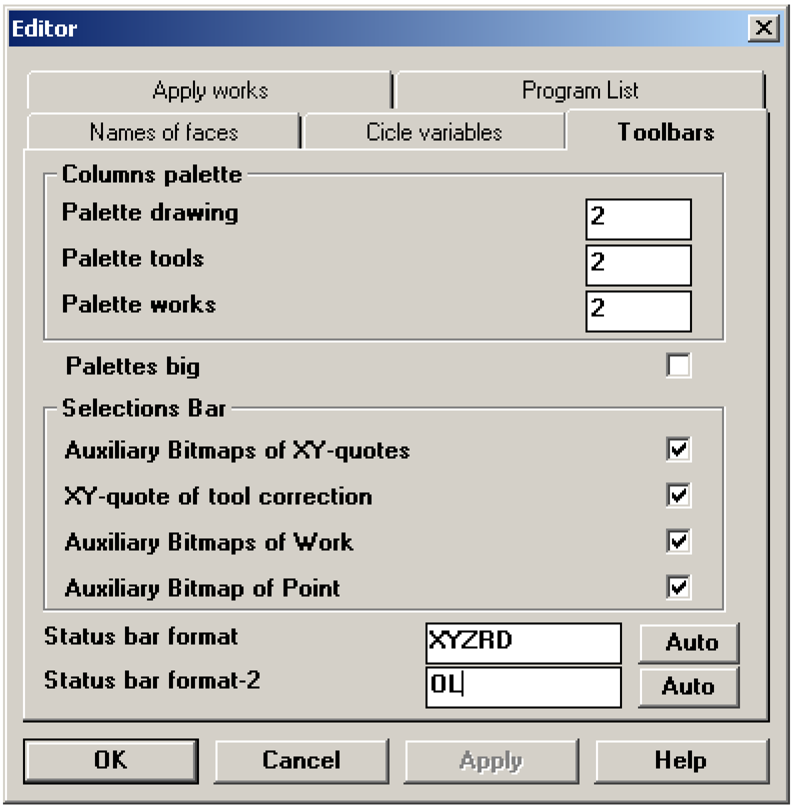
These variables are used by the PLC to distinguish the execution of the piece from another. Their meanings change for each machine.



	Name	Value
Var0	execution	0.
Var1	load flag	0.
Var2		0.
Var3		0.
Var4		0.
Var5		0.
Var6		0.
Var7		0.

- **Name** is the name of each of the general variables of a program (max 20 characters). When a new program is opened the name appears in the ***Edit->Dimensions*** dialog box.
- **Value** is the value given to each of the general variables during the creation phase of a new program.

Toolbars



Columns palette

Palette drawing 2

Palette tools 2

Palette works 2

Palettes big □

Selections Bar

Auxiliary Bitmaps of XY-quotes ☑

XY-quote of tool correction ☑

Auxiliary Bitmaps of Work ☑

Auxiliary Bitmap of Point ☑

Status bar format XYZRD

Status bar format-2 OL

The first items in the dialog box are used to customize the palette columns. The term palette refers to a button box used for the quick activation of commands. It can be moved around the screen. If, for example, you want the tool palette to appear on a single line, or all the bitmaps of the buttons placed next to each other horizontally, then you must assign a greater number of columns or a number equal to the total number of bitmaps for the palette.

- **Drawing palette** is the number of columns (from 1 to 50) on which to view the drawing palette.
- **Tool palette** is the number of columns (from 1 to 50) on which to view the tool palette.
- **Working palette** is the number of columns (from 1 to 50) on which to view the working palette.
- **Palettes big** if selected, the palettes are larger than normal.

- **Coordinate auxiliary bitmaps** enables or disables the display of the mouse position bitmap and the tool correction coordinate bitmap in the Selection Bar next to the corresponding coordinate display fields.
- **Tool correction coordinates** enables or disables the display of the tool correction coordinates in the Selection Bar .
- **Working auxiliary bitmaps** enables or disables the display of the three working auxiliary bitmaps in the Selection Bar .
- **Point auxiliary bitmaps** enables or disables the display of the point type bitmaps in the Selection Bar. It only appears when operating on a profile. They are, from the left:
  - working type (point, setup, etc.)
  - profile tool movement, represented by one of the following icons:
    - tool down in setup
    - tool down then up (isolated setup)
    - profile segment followed by tool up
    - intermediate profile segment
    - setup not carried out due to dummy point anchoring

- **Status bar format** is the format of the first area of the Status Bar that can be customized, an area that can contain data such as 1 to 15 values. These are indicated one below the other by means of the character that distinguishes them, that is :

x , X	for X position
y , Y	for Y position
z , Z	for Z position
i , I	for the centre X coordinate of an arc in the XY or XZ plane
j , J	for the centre Y coordinate of an arc in the XY or YZ plane
k , K	for the centre Z coordinate of an arc in the XZ or YZ plane
r , R	for the arc radius
a , A	for the position of the axis of rotation
c , C	for the position of the axis of rotation
b , B	for the position of the pitch axis
w , W	for W axis position
v , V	for V axis position
d , D	for tool diameter (for point or setup workings)
m, M	machine
g , G	group
t , T	tool

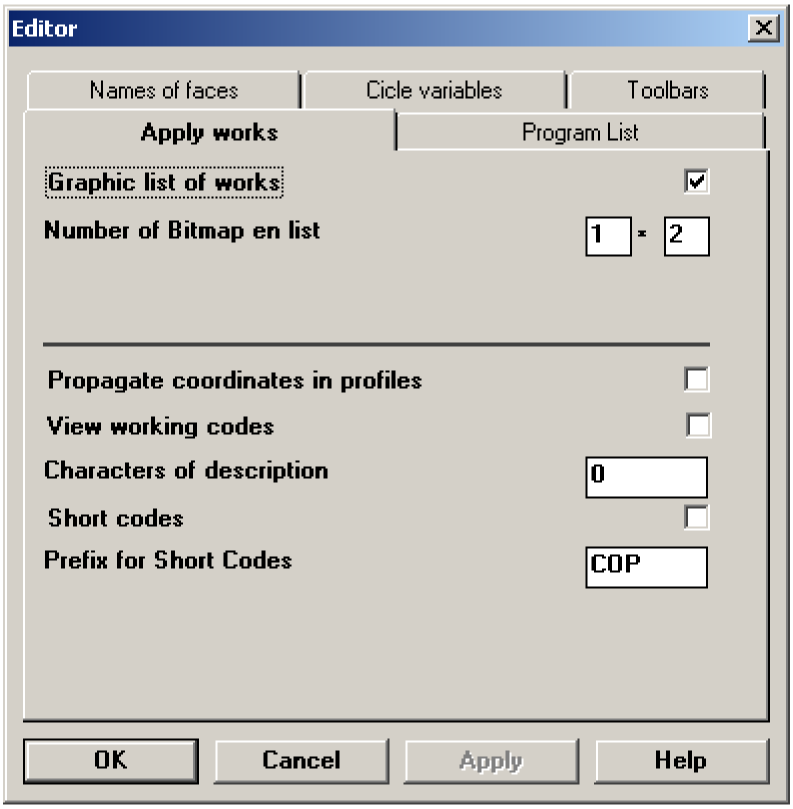
The **[Auto]** button automatically activates a display of the XYZRD parameters if there is a Z axis, otherwise the fields XYRD are displayed.

- **Status bar format-2** is the format of the second area of the Status Bar that can be customized, an area that cannot be assigned, and that contains data relevant to the following parameters:

l , L	level
o, O	origin

The **[Auto]** button automatically activates a display of the parameters configured by the manufacturer.

Apply Works



Graphic list of works ☑

Number of Bitmap en list 1 * 2

Propagate coordinates in profiles □

View working codes □

Characters of description 0

Short codes □

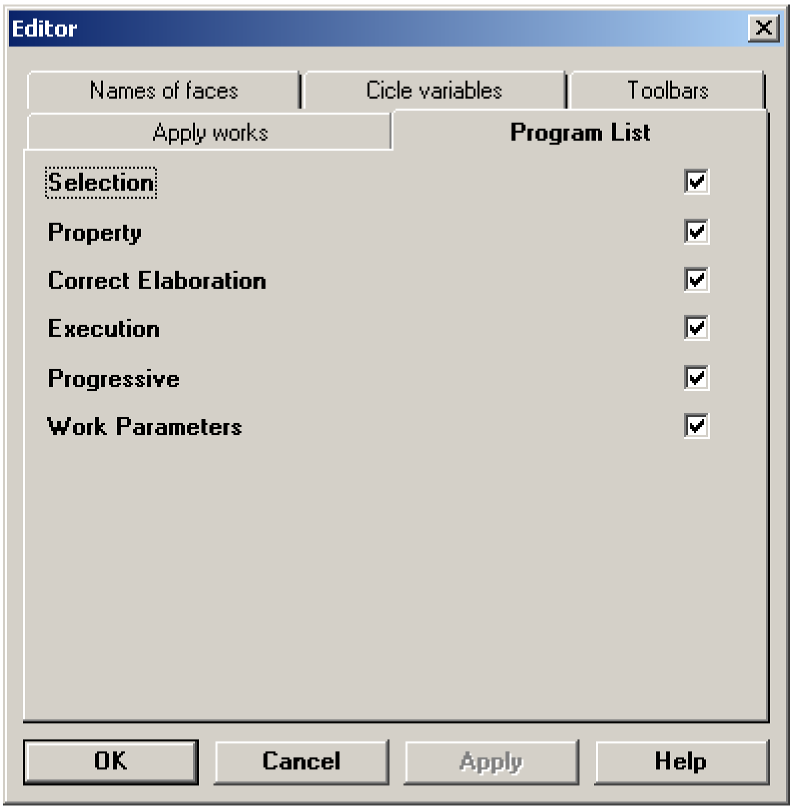
Prefix for Short Codes COP

- **Graphic list of works** enables or disables the graphic menu of the workings as an alternative to the text format menu.

- **Graphic list bitmaps** is the number of figures of the graphic menu that are displayed simultaneously. The lists are sorted by ascending subtype order or, by subtype, by progressive code.
- **Propagate coordinates in profiles** enables or disables the settings of the preceding block during the insertion phase of a new block below a profile working (setup, arc or linear) for the X, Y, Z positions.
- **View working codes** enables or disables the display in the Status Bar, apart from the name of the working, the corresponding operating code.
- **Characters of description** requires an indication of the maximum number of characters that are to be displayed in the "Program List". The maximum number is 61.
- **Short codes** if activated, the Short Code takes the place of the Working Description.
- **Prefix for Short Codes** requires an indication of the characters that make up the prefix for the short code (example: 'COP').

Program List

The items enabled in this window are displayed in the Program List window .



Selection ☑
Property ☑
Correct Elaboration ☑
Execution ☑
Progressive ☑
Work Parameters ☑

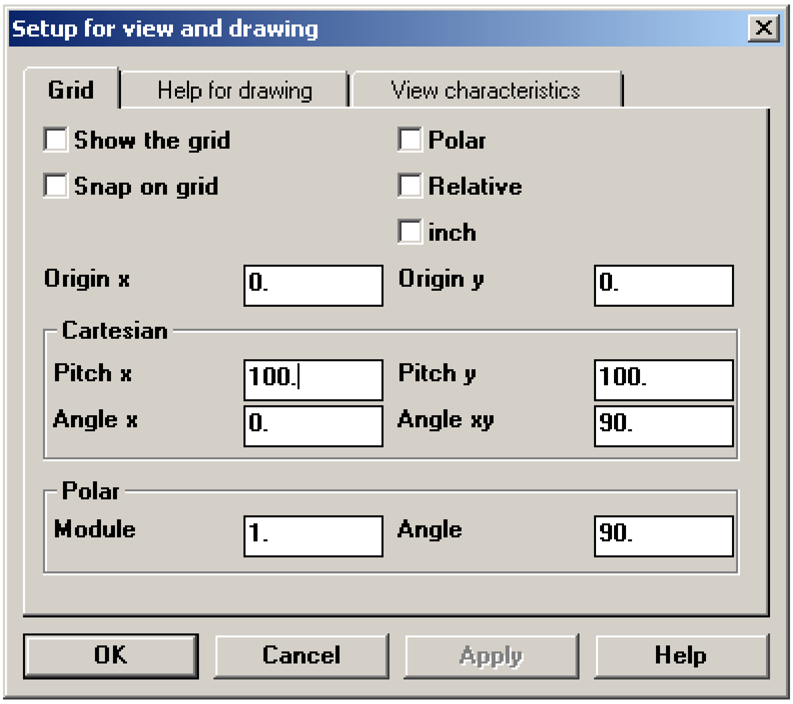
- **Selection** if selected, the selected program blocks are marked with an asterisk.
- **Property** if selected, the properties of the block are displayed.
- **Correct Elaboration** if selected, the block indication OK or NOK appears.
- **Execution** if selected, indicates whether or not the block will be executed.
- **Progressive** if selected, the progressive number of the program block is displayed.
- **Work Parameters** if selected, the working parameters are displayed.

6.2 Graphic settings

In Edicad it is possible to set the viewing characteristics of the piece in the work window and configure help tools for the programming operations from the ***Set->Setup for view and drawing ...*** menu.

Grid

By setting a grid a graphic grid appears to help with the insertion of the workings.



□ Show the grid □ Polar

□ Snap on grid □ Relative

□ inch

Origin x: 0. Origin y: 0.

Cartesian

Pitch x: 100. Pitch y: 100.

Angle x: 0. Angle xy: 90.

Polar

Module: 1. Angle: 90.

- **Show the grid:** enables or disables the view of the grid in Face View.
- **Snap on grid:** influences the insertion of the coordinate on the intersection of the grid; this is only in the event that the positioning button is enabled in the "with mouse" mode.
- **Polar:** enables the polar representation, based on the "Module" and "Angle" parameters. If disabled, viewing is assumed cartesian.
- **Relative:** the grid origin is placed on the last inserted point. If disabled, the origin will be assumed absolute.
- **Inch:** if enabled, all the "Origin x and y", "Cartesian" and "Module" positions are expressed in inches.
- **Origin x** and **Origin y:** X and Y coordinates of the grid origin.

The following parameters are assigned according to whether the grid is expressed in polar coordinates or in cartesian coordinates:

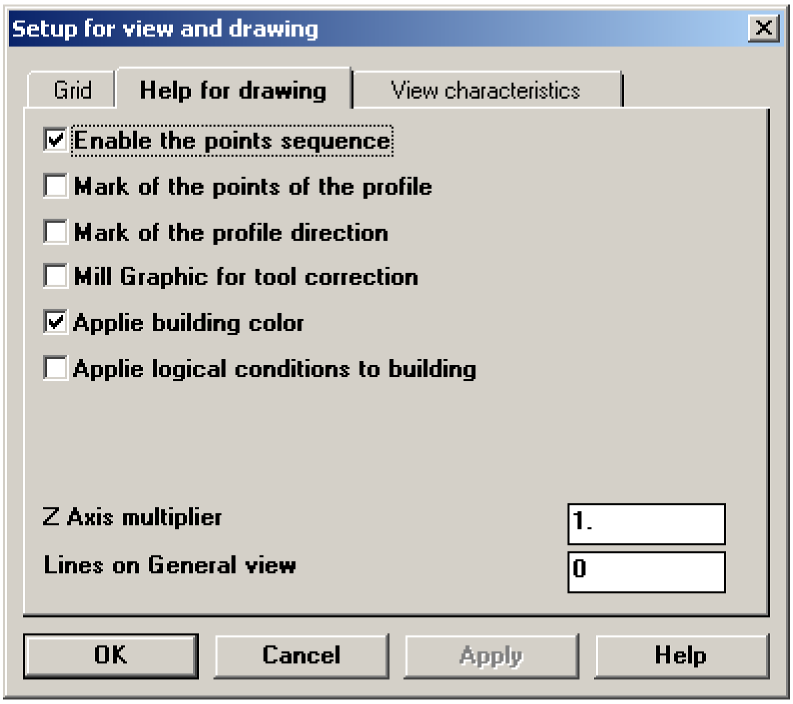
In the case of Cartesian grid

- **Pitch *x*** and **Pitch y:** grid pitches along X and Y.
- **Angle x** and **Angle xy:** define, the first, the slope angle of the x axis and, the second, the angle between the two axes.

In the case of Polar grid

- **Module:** is the pitch along the grid radius.
- **Angle:** is the grid angular step.

Help for drawing



Setup for view and drawing

☑ Enable the points sequence

□ Mark of the points of the profile

□ Mark of the profile direction

□ Mill Graphic for tool correction

☑ Applie building color

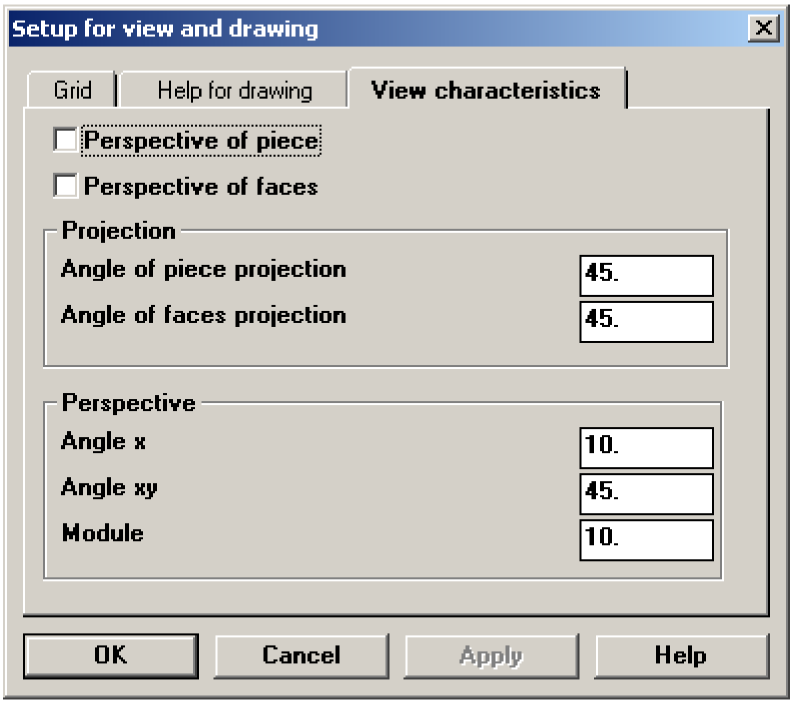
□ Applie logical conditions to building

Z Axis multiplier: 1.

Lines on General view: 0

- **Alternate sequence of points:** related to the profile building. When enabled, forces a compulsory Angular - Normal- Angular sequence on the point inserting.
- **Mark of the point of the profile:** enables or disables the profile points visualization. The visualization can be enabled or disabled on each face by the ***View->Points of profile*** command.
- **Mark of the profile direction:** enables or disables the profile direction visualization. The visualization can be enabled or disabled on each face by the ***View->Direction*** command.
- **Mill Graphic for tool correction** if activated the cavity is viewed as a result of the mill-radius p rocess.
- **Applie building colour** if enabled, the building working is displayed with the colour assigned to the building workings. Otherwise it is displayed in the colour defined for the working that it represents.
- **Applie logical conditions to building** if activated, the logical conditions are applied to the building working, otherwise they are assigned as non-executive. If the logical conditions are verified in the event of Logical Conditions View the building workings are displayed.
- **Z axis multiplier** is a number which, when multiplied by the value of the Z axis, expands the three-dimensional graphic display for a better view. This is because normally there does not exist a proportion of dimension between the X. and Y axes and the Z axis and in the three-dimensional display there is the risk of not managing to appreciate the depth.
- **Lines on General view** dimensions the face list in the General View. If it is equal to zero it displays the real faces of the piece.

View characteristics



Setup for view and drawing

□ Perspective of piece

□ Perspective of faces

Projection

Angle of piece projection: 45.

Angle of faces projection: 45.

Perspective

Angle x: 10.

Angle xy: 45.

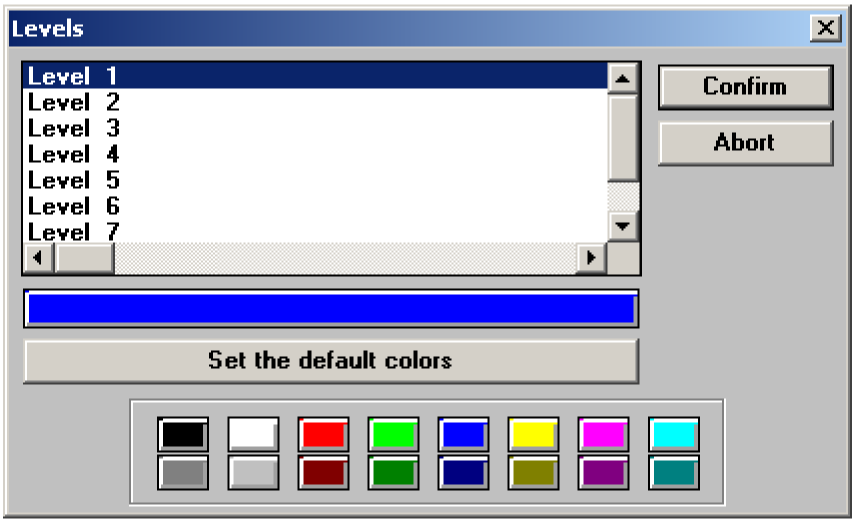
Module: 10.

- **Perspective of piece** when selected, shows the piece in perspective, in the general piece view. When not selected, piece is shown in projection.
- **Perspective of faces** when selected, shows the face in perspective, activating a 3D view. When not selected, face is shown in projection.
- **Angle of piece projection:** defining the angular view of the piece in the Main view Window.
- **Angle of faces projection:** defining the angular view of the face in the Face view Window.
- **Angle x** angle of X axis with reference to the horizontal (values: 0°÷360°).
- **Angle xy** rotation angle of piece (values: 0°÷180°), starting from: 0 = top view, 90 = front view, 180 = opposite view.
- **Module** (values: 1÷10.000). Represents a hypothetical distance from where the perspective is viewed.

6.3 Levels Setting

Every working instruction may be associated to a specific "level", then allowing to collect some of them on the base of the colour, improving readability, or to select only the works pertaining to a specific level.
System provides 9 levels: a level 0 (assumed by default) and other 8 (from 1 to 8) which may be characterised by a specific colour in the work displaying: if the work is marked with level N. 0, it will be displayed with its predefined colour.. Select in the **Settings** menu the **Levels** item.

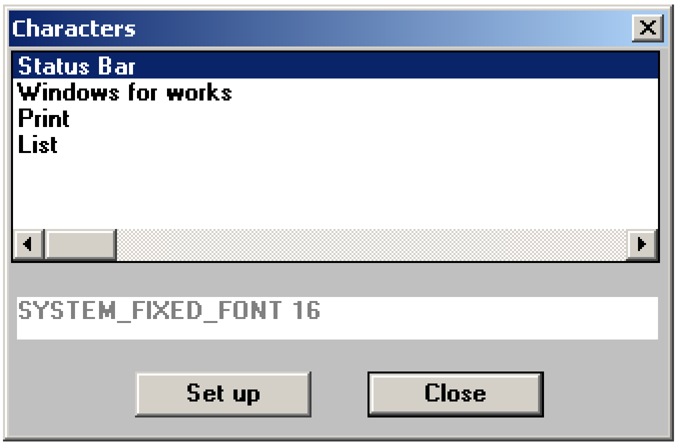
The following dialog box appears:



A preferred colour may be assigned to each level. To return to the default situation simply select the button [Set default colours].

6.4 Characters Setting

Any text typology used in displaying, may be characterized by a character font.
The list of environments in which the character font may be changed appears in a dialog box. The change can be made by selecting the relevant font and double-clicking it, or by means of the **[Set up]** button. The standard Windows box with the selection of character types appears; the style and size can also be set in this box.

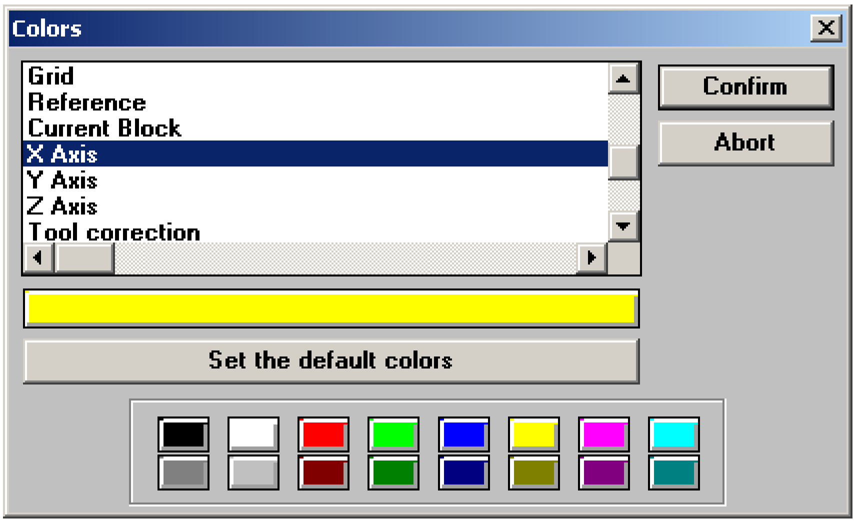


SYSTEM_FIXED_FONT 16

6.5 Colours Setting

The colours of the several windows of Edicad and the different construction typologies may be modified, as to customize the working environnement

The dialog window allows the following colour selection:



:

The colour corresponding to each of the items in the list is displayed in the colour bar below the list. Select one of the displayed colours in order to choose a new colour to allocate to the highlighted item. To return to the default situation simply select the **[Set the default colours]** button**.**

Edicad

Il presente manuale è di proprietà della T.P.A S.p.a.
Ne è vietata la duplicazione non autorizzata.
La società si riserva il diritto di modificarne il contenuto in qualsiasi momento

T.P.A S.p.A Tecnologie e prodotti per l'automazione
Via Carducci 221
20099 Sesto S. Giovanni (MI)
Tel. 0236527550
Fax. 022481008
www.tpaspa.it