Department of

Mechanical Engineering

LAB MANUAL

CAD/CAM

B.Tech–5th Semester

KCT College OF ENGG AND TECH.

VILLAGE FATEHGARH DISTT.SANGRUR

List of experiments

1&2. Part programming on CNC machine and execution of part program for machining given profile.

3. Programming of robots for various applications.

4. Part Modeling using some of the Modeling techniques.

5. Component Assembly in CAD and Generating & Modifying drawings

6. To learn Simple Auto lisp Programs.

7. To Create Blocks and External Reference.

8. To Study Simulation Packages

LAB IN-CHARGE **HOD ME**

EXPERIMENT NO.1 & 2

AIM: PART PROGRAMMING ON CNC MACHINE & EXECUTION OF PART PROGRAMME FOR MACHINING GIVEN PROFILE.

The operation of an NC machine is controlled by a program written in NC code, called NC program, which consists of a series of statements, or blocks, specifying the operations to be executed and cutter motion to be realized by the NC machine in order to machine a specific part. On classic metal-cutting machine tools, a machining process plan is usually provided and should be followed by the operator in order to make the part to required dimensions and tolerances. Correspondingly, an NC program is the translation of a machining process plan from English into NC codes that are understandable to the NC machine-controller system.

Although there are NC code standards, such as international standard ISO 6983/1, controllers made by different manufacturers use different NC codes. An NC part program consists of the following:

1. A declaration of the program number.
2. Statement defining the origin of the coordinate system and the kind of coordinates (incremental or absolute) used to describe the tool position or path in the program.
3. Statement defining the operations, other than tool movements, to be performed by the NC machine, such as turning the coolant and spindle on or off, setting the spindle speed and feed rates, and changing or selecting the tool(s).
4. Statement defining the movement and position of the tool.
5. Statement defining the operations to be performed by the controller, such as "Stop reading and execution", "Stop reading and execution and rewind the memory" and "call a subprogram".

THE BLOCK FORMATS OF NC PROGRAM

There have been different ways to specify NC words in a statement (or block), depending on the design of the NC controller on e of the earliest NC formats is fixed sequential. Here, a specific sequence is required for every word in a block, and all the words that are used by the controller are required in block whether or not they specify meaningful changes in the NC machining state, each word consists of a number specifying a state (e.g. position in the x direction, feed rate, or spindle speed) of the NC machine. Therefore a statement is composed of fixed number of numerals that are listed in a fixed order and that represent different NC words. For example, if the order of specifying NC words is:

Statement No.-motion type -X-coordinate- Y-coordinate- Z-coordinate -feedrate-spindle speed-miscellaneous functions (e.g., spindle on off).

Then a statement in fixed sequential format may appear as:

003 01 1.00 2.00 3.00 2.0 500 3

The second type of format is Tab sequential format. In this format all the information has to be in a particular sequence but various elements are separated by the code TAB. In case the information remains same as in the earlier block, then it needs not to be repeated but the code TAB is used in succession. For example:

31TAB 81TAB 235000 TAB 115000 TAB....

38TAB 300000 TAB TAB.....

This means that the operation code 81 and coordinate Y 115000 are to be repeated in the second step.

WORD ADDRESS FORMAT

This is the third type of format. The modern CNC controller system used word address format. Each word in a statement consists of a character identifying the meaning or address of the word and a number representing its contents. Each element of information is prefixed by a alphabetic character which the controller understands for conveying it to the particular register. In this format the sequence is not necessary since all information is labeled by an alphabet. Further, repetition is not necessary since the controller in the absence of any information, take it from the earlier block. This eliminates the chances of any error and also shortens the program to large extents. Thus, saving time in preparing the program and its subsequent processing, e.g.

N30 G99 G21 G40 G50 S300

PREPARATORY FUNCTION (G WORD)

The two digit G command is programmed to setup the control to perform an automatic machine operation. A list of frequently used G codes is given in the Table No.1.

MISCELLANEOUS FUNCTION (M WORD)

An M Word is used to initiate auxiliary functions particular to the machine. One M code can be programmed within one program block together with other part program information.

TOOL ADDRESS FUNCTION (T WORD)

The T word used in conjunction with "M06" is used to call up the required tool on an automatic indexing turret machine, and to achieve its tool offsets.

SPINDLE SPEED ADDRESS FUNCTION (S WORD)

- In constant surface speed mode (G96) the four digits S Word is used to command the required surface speed in either feet or meters per minute.
- In direct R.P.M. mode (G97), the four digit S Word is used to command the spindle speeds incrementally, in R.P.M., between the ranges available for the machine.
- Prior to entering constant surface speed mode (G96) the S Word is used to specify a speed constraint, the maximum speed you wish the spindle to run at. To set this restraint the S Word is programmed in conjunction with the G 50 word.

FEED RATE ADDRESS FUNCTION (F WORD)

The F Word is used to specify the feed required for the machine according to the speed of the spindle and the material of the work part to be machined.

- In G99 mode the F Word is used to command feed/rev.
- In G98 mode the F Word is used to command feed/min.
- In G76 mode the F Word specifies the lead (pitch) of the thread.

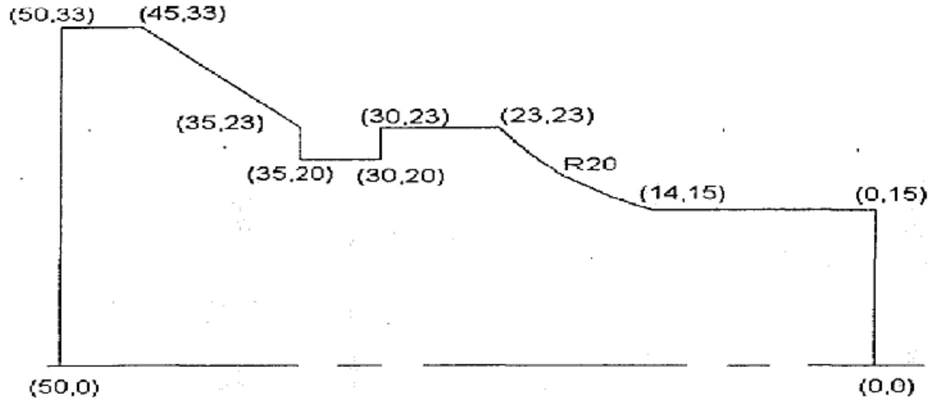
Some important G and M codes which are frequently used are shown in

Table No. 1

G00	Positioning (Rapid Traverse)
G01	Linear Interpolation (Feed)
G02	Circular Interpolation CW
G03	Circular Interpolation CCW
G20	Inch data input
G21	Metric data input
G27	Reference Point return check
G28	Reference point return
G32	Thread cutting
G34	Variable lead thread cutting
G70	Finishing cycle
G71	Stock removal in turning
G74	Peck drilling in Z-axis
G75	Grooving in X-axis
G76	Thread cutting cycle
G90	Absolute Mode
G92	Thread cutting cycle
G96	Constant surface speed control
G97	Constant surface speed control cancel
G98	Feed per minute
G99	Feed per revolution
M00	Program Stop
M01	Optional Stop
M02	Program re-set
M03	Spindle forward
M04	Spindle reverse
M05	Spindle Stop
M06	Auto Tool change
M07	Coolant 'A' on
M08	Coolant 'B' on
M09	Coolant off
M10	Chuck open
M11	Chuck close
M13	Spindle Forward and Coolant on
M14	Spindle Reverse and Coolant on
M30	Program reset and rewind

G00	Positioning (Rapid Traverse)
G01	Linear Interpolation (Feed)
G02	Circular Interpolation CW
G03	Circular Interpolation CCW
G20	Inch data input
G21	Metric data input
G27	Reference Point return check
G28	Reference point return
G32	Thread cutting
G34	Variable lead thread cutting
G70	Finishing cycle
G71	Stock removal in turning
G74	Peck drilling in Z-axis
G75	Grooving in X-axis
G76	Thread cutting cycle
G90	Absolute Mode
G92	Thread cutting cycle
G96	Constant surface speed control
G97	Constant surface speed control cancel
G98	Feed per minute
G99	Feed per revolution
M00	Program Stop
M01	Optional Stop
M02	Program re-set
M03	Spindle forward
M04	Spindle reverse
M05	Spindle Stop
M06	Auto Tool change
M07	Coolant 'A' on
M08	Coolant 'B' on
M09	Coolant off
M10	Chuck open
M11	Chuck close
M13	Spindle Forward and Coolant on
M14	Spindle Reverse and Coolant on
M30	Program reset and rewind

Part Progran No. 1




```
[BILLET X 67 Z 60
(*****CUTTING CONDITIONS*****
N10 G99 G21 G97 S2000 M13
(*****TOOL CALL*****
N20 M06 T01
(*****FACING OPERATION*****
N30    G00   X 66.00 Z 2.00
N40    G01   Z 0.00 F 0.125
N50    G01   X -1.00
N60    G00   X 66.00 Z 1.00
(*****ROUGHING OPERATION*****
N70    G71   U 1.50 R 0.50
N80    G71   P90 Q120 U 0.5 W 0.5 F 0.125
N90    G01   X 30.00
N100   G01   Z -14.00
N110   G02   X 46.00 Z -23.00 R 20.00
N120   G01   Z -30.00
```

EXPERIMENT NO.3

AIM: Programming of robots for various applications.

Steps	Move or signal	Comments
Step 1	1,1	Start at home position
Step 2	8,1	Move to wait position
Step 3	WAIT 11	Wait for press to open
Step 4	8,8	Move to pickup point
Step 5	SIGNAL 5	Signal gripper to close
Step 6	8,1	Move to safe position
Step 7	SIGNAL 4	Signal press to actuate
Step 8	1,1	Move around press column
Step 9	1,8	Move to tote pan
Step 10	SIGNAL 6	Signal gripper to open
Step 11	1,1	Move to safe position

Each step in the program is executed in sequence, which means that SIGNAL and WAIT commands are not executed until the robot moved to the point indicated in the previous step.

Above written programs seems to look like computer programs. This is for convenience in our explanation of the programming principles. The actual teaching of the moves and signals is accomplished by leading the arm through the motion path and entering the non motion instructions at the control panel or with the teach pendant. In the majority of industrial applications today, robots are programmed using one of the lead through methods. Only with the textual language programming do the programs read like computer program listings.

The operation of the gripper was assumed to take place instantaneously so that its actuation would be completed before the next step in the program was started. Some grippers use a feedback loop to ensure that the actuation has occurred before the program is permitted to execute the next step. A WAIT instruction can be programmed to accomplish this feedback.

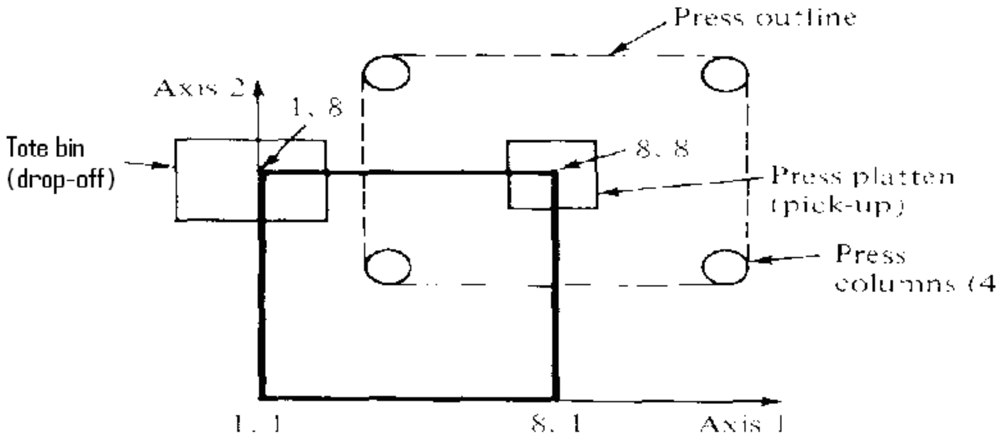


Figure: Robot work space for press unloading operation

An alternative way to address this problem is to cause the robot to delay before proceeding to the next step. In this case, the robot would be programmed to wait for a specified amount of time to ensure that the operation had taken place. The form of the command for this second case has a length of time as its argument rather than an input line. The command

DELAY X SEC

Indicates that the robot should wait X seconds before proceeding to the next step in the program

EXPERIMENT NO.4

AIM: Part Modeling using some of the Modeling techniques.

Procedure for Pro-E modeling

Step 1:- Draw rectangle with specify dimension l60mm x 100mm.

Step 2:- Extrude with specify dimension 12mm.

Step 3:- Draw two rectangles with dimension 20mm x 100mm on its corners.

Step 4:- Extrude with specify dimension 100mm.

Step 5:- Fillet the corners of these rectangles using round tool with dimension 50 mm.

Step 6:- Select side plane as a reference.

Step 7:- Draw circle with diameter 30.

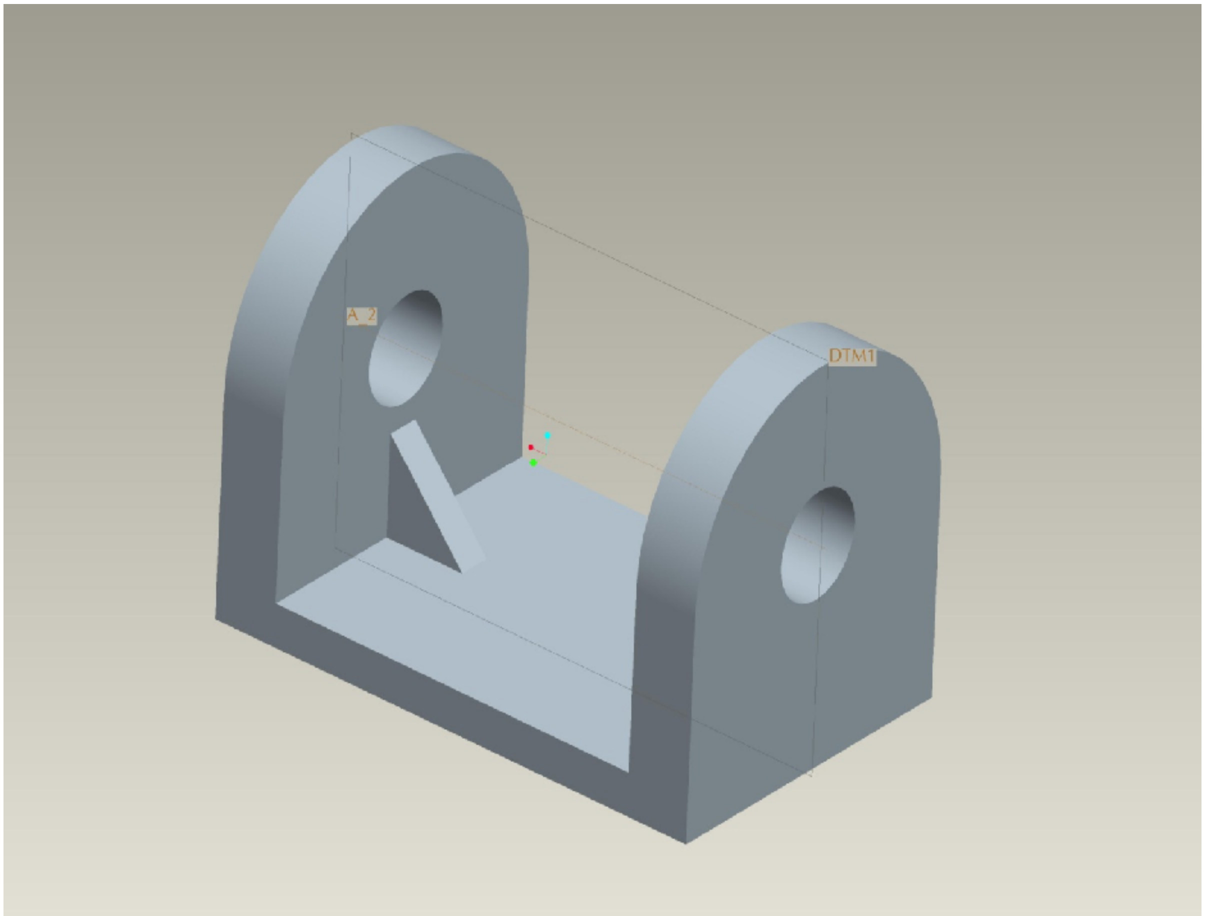
Step 8:- Cut Extrude the circle using through all option.

Step 9:- Draw a plane exactly in middle of a bigger rectangle

Step 9:- Draw a preparatory open sketch on this work plane.

Step 10:- Use Rib tool to draw rib on its edge with 10mm thickness

Step 11:- Save the file


A_2
DTM1

EXPERIMENT NO.5

AIM: Component Assembly in CAD and Generating & Modifying drawings

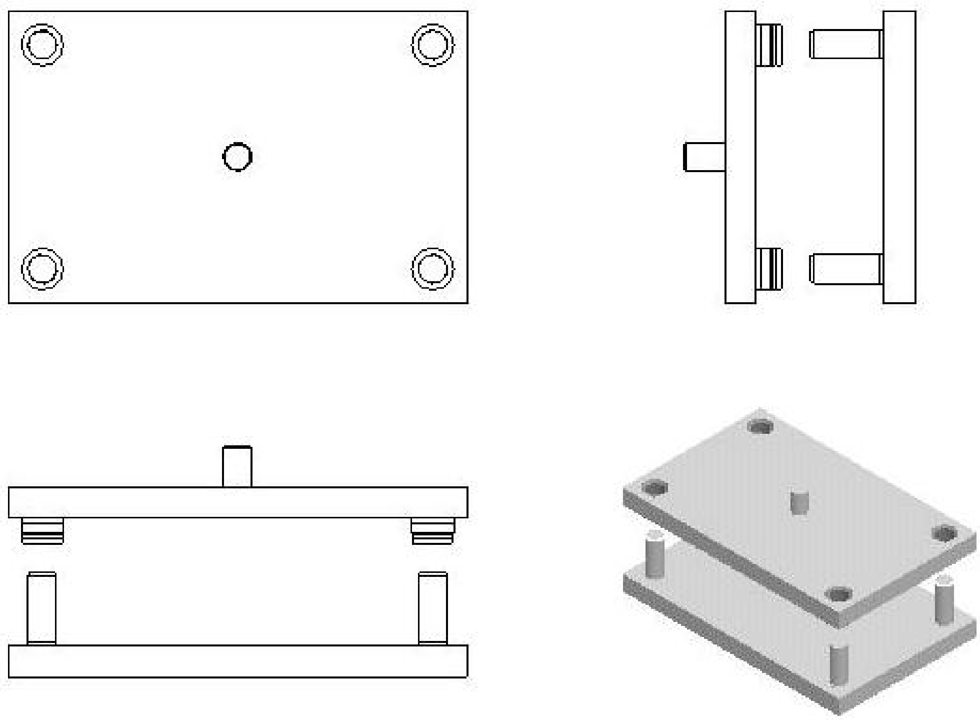


Figure: Mold Assembly

EXPERIMENT NO.6

Aim: To learn Simple Auto lisp Programs.

A program to select a Line entity, a Text entity, and then rotate the text entity based on the angle of the line.

```
;LTR.lsp - select a Line entity and a Text entity then
rotate the text entity based on line angle.

(defun C:LTR()

;turn off the system echo
(setvar "cmdecho" 0)

;display a message on the command line
(princ "\n Select LINE with Correct Angle.")

;let the user select one entity
(if (setq eset(entsel))
   (progn

      ;get the entity name from the entsel command
      (setq en(car eset))

      ;get the DXF group codes of the selected entity
      (setq enlist(entget en))

      ;check to see if a LINE was selected
      (if(= "LINE" (cdr(assoc 0 enlist)))
        (progn

          ;display a message on the command line
          (princ "\n Select TEXT to Match Line Angle.")

          ;let the user select one entity
          (if(setq eset2(entsel))
            (progn

              ;get the starting point of the line
              (setq ept1(cdr(assoc 10 enlist)))

;get the end point of the line
              (setq ept2(cdr(assoc 11 enlist)))


              ;get the angle from the end points of the line
              (setq ang1(angle ept1 ept2))

              ;get the entity name from the entsel function
              (setq en2(car eset2))
```

```
            ;get the DXF Group Codes of the entity
            (setq enlist2(entget en2))

            ;change the angle in the text entities DXF
group codes
            (setq enlist2(subst (cons 50 ang1)(assoc 50
enlist2)enlist2))

            ;update the text entity
            (entmod enlist2)
         )

         ;if the second entity wasn't selected...

         (princ "\n Select Text Entity Please. Program
Aborted.")
       )
     )
     ;if the first entity wasn't a line

     (princ "\n Enitity selected was not a LINE. Program
Aborted.")
    )
  )
  ;if the first entity wasn't selected

  (princ "\n Nothing selected. Program Aborted.")
)
;reset the system echo

(setvar "cmdecho" 1)

;suppress the last echo
(princ)
)

;End of Program
```

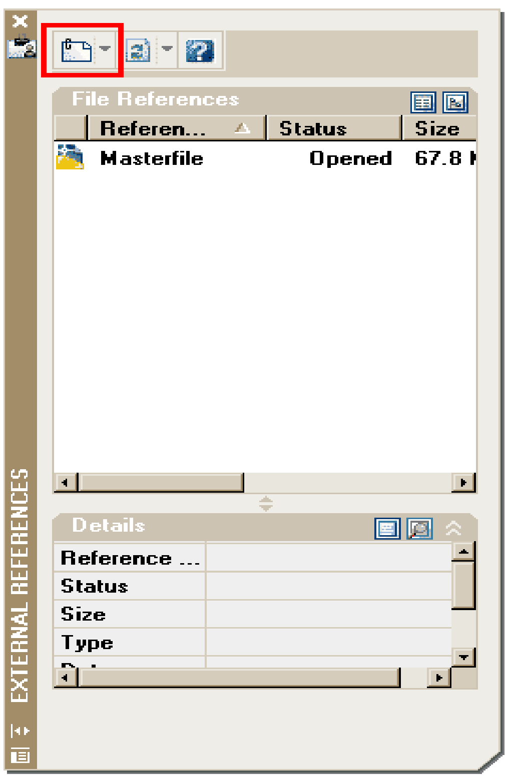
EXPERIMENT NO.7

Aim: To Create Blocks and External Reference.

An X-ref is an 'external reference' to another AutoCAD drawing file. One file can reference many other files and display them as if they were one. These are used in larger projects for many reasons:

- They keep the file sizes down.
- They allow many users to work on individual components of a project.
- Every time an X-ref is loaded, it is the most recent version of the drawing.
- X-ref's can be updated, added, or unattached from the main drawing at any time.
- You can X-ref drawings that they themselves X-ref other drawings (nesting).

In these days of networks and the Internet, many projects are produced this way. People from anywhere in the world can collaborate on a project in real time.



The first thing to do is to *ATTACH* the X-ref. This means that you are linking another drawing to your current one. Do this by starting the **XFEF** (**XR**) command. When you start the command, you will see the palette shown in figure

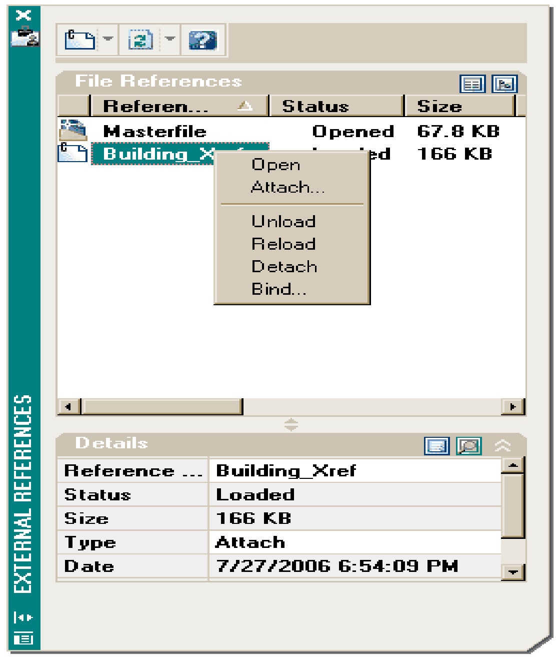
ATTACH – Attaches another X-ref.

DETATCH – Detaches the selected X-ref.

RELOAD – Updates the selected X-ref - use this if the Xref was changed.

UNLOAD – Removes the X-ref, but retains the reference for future use.

BIND – Permanently attaches a loaded X-ref, so that it is part of the drawing.



EXPERIMENT NO.8

AIM: To Study Simulation Packages

Costly and laborious prototyping hinders a design team, resulting in compromised schedules and budgets. Through Pro/ENGINEER Mechanica, we can better understand product performance, and accommodate the digital design accordingly -- all without needing a specialist's FEA-background.

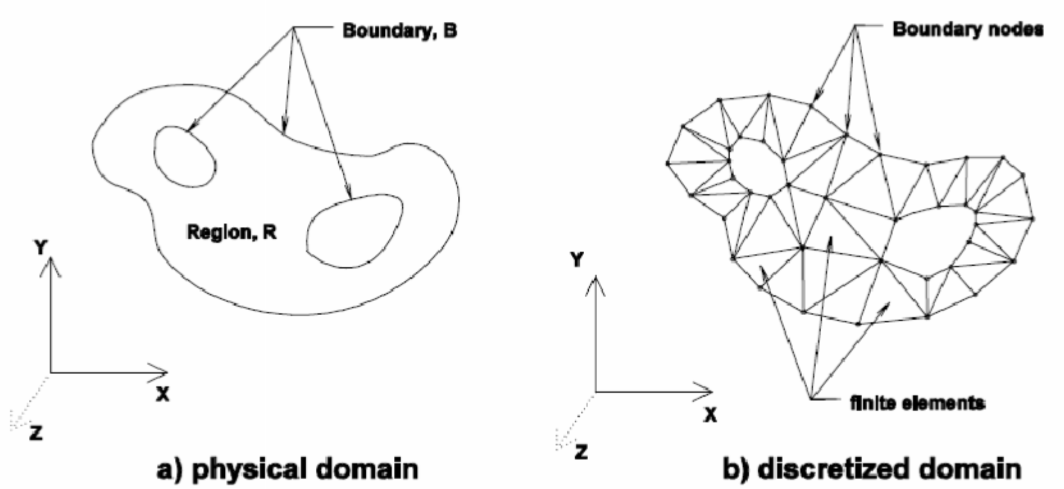


Figure 1 The problem to be solved is specified in a) the physical domain and b) the discretized domain used by FEA

For a two dimensional problem, the governing physical law or principle might be expressed by a partial differential equation (PDE), for example

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$$

that is valid in the interior of the region R. The solution to the problem must satisfy some boundary conditions or constraints, for example T = T(x,y), prescribed on the boundary B. Both interior and exterior boundaries might be present and can be arbitrarily shaped.

Note that this governing PDE may be (and usually is!) the result of simplifying

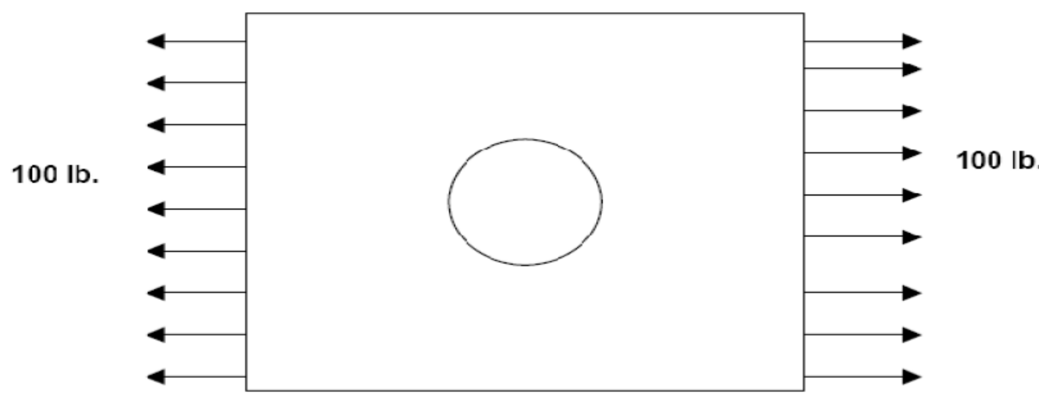
Note that this governing PDE may be (and usually is!) the result of simplifying assumptions made about the physical system, such as the material being homogeneous and isotropic, with constant linear properties, and so on.

In order to analyze this problem, the region R is discretized into individual finite elements that collectively approximate the shape of the region, as shown in Figure 1(b). This discretization is accomplished by locating nodes along the boundary and in the interior of the region. The nodes are then joined by lines to create the finite elements. In 2D problems, these can be triangles or quadrilaterals; in 3D problems, the elements can be tetrahedral or 8-node "bricks". In some FEA software, other higher order types of elements are also possible (e.g. hexagonal prisms). Some higher order elements also have additional nodes along their edges. Collectively, the set of all the elements is called a finite element mesh. In the early days of FEM, a great deal of effort was required to set up the mesh. More recently, automatic meshing routines have been developed in order to do most, if not all, of this tedious task.

Starting from the simplified geometric model, there are generally several steps to be followed in the analysis. These are:

1. Identify the model type
2. Specify the material properties, model constraints, and applied loads
3. Discretize the geometry to produce a finite element mesh
4. Solve the system of linear equations
5. Compute items of interest from the solution variables
6. Display and critically review results and, if necessary, repeat the analysis

We have applied 100 pound load on both sides below



Now in Pro Mechanical we shall consider only structural analysis

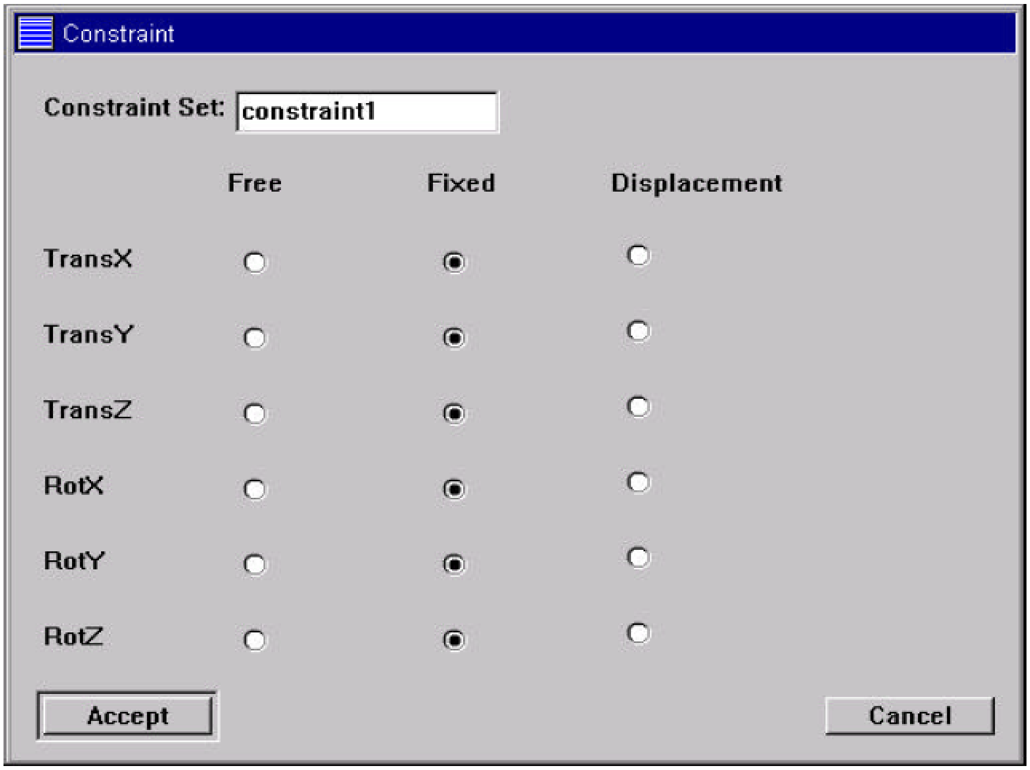
- Go to the ***Menu Bar*** and under ***Applications*** click on ***Mechanica***.
- You should get a screen in which will tell you the ***Principal Units*** used for your model. Click on ***Continue.***
- You will now get a menu manager as shown below. Since we are concerned with only structural analysis click on ***Structure***.

You will see that there are 5 subdivisions under ***Mec Struct.*** These are the various steps to be followed in the same sequence as they appear.

a. Model: Here we set up the various constraints, loads, materials, convert it into a shell model if necessary, etc.

- **Applying constraints**: Click on ***Constraints*** and then on ***Create*** and then ***Face/Surface*** (because you are applying constraints on the fixed surface).
- Pick the surface that is fixed and then say ***Done Sel***. You will now see a window as shown below.

- Notice that all the degrees of freedom are fixed, which means that the face/surface is fixed in all directions.
- Be sure that all the buttons corresponding to ‘fixed’ are set ‘on’ and then click on ***Accept.***



- You will now get another window asking you to describe the constraint. You could either describe it or leave it blank and say ***Accept.***
- Once this is done you should see a triangle on the surface where the constraints have been applied.

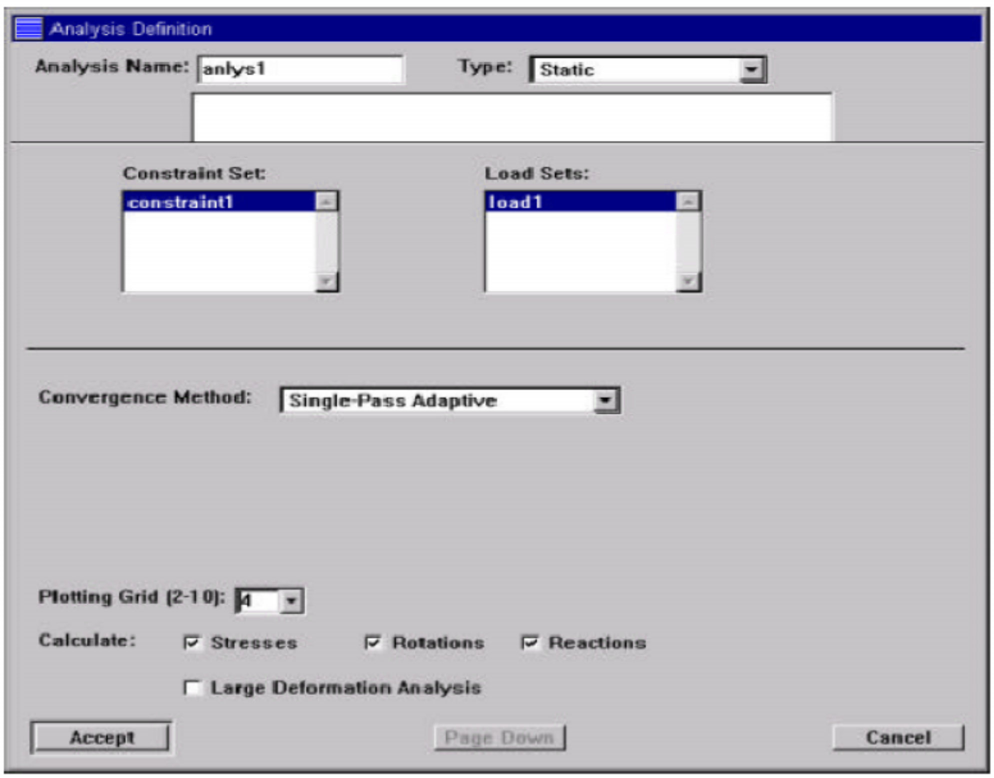
- **Applying loads:** Click on ***Loads*** and click ***Create***.
- You can create different kinds of loads as listed in the menu. But in this problem we want a point load of 1000lbf. Hence click on ***Point***.

- The force will be displayed as an arrow on the model. If the display does not show a tensile force, then click on ***Edit*** and select the arrow. Make the required changes.

- **Applying Material Properties:** Pro-mechanica has an in built library of materials. Click on ***Materials*** and say ***Assign*** and ***Part***. Pick the part and say ***Done Sel***. You will now see a window as shown below. Select ***Steel*** and click on ***Accept***. It will display the material properties of steel. Click on ***Accept*** once again.

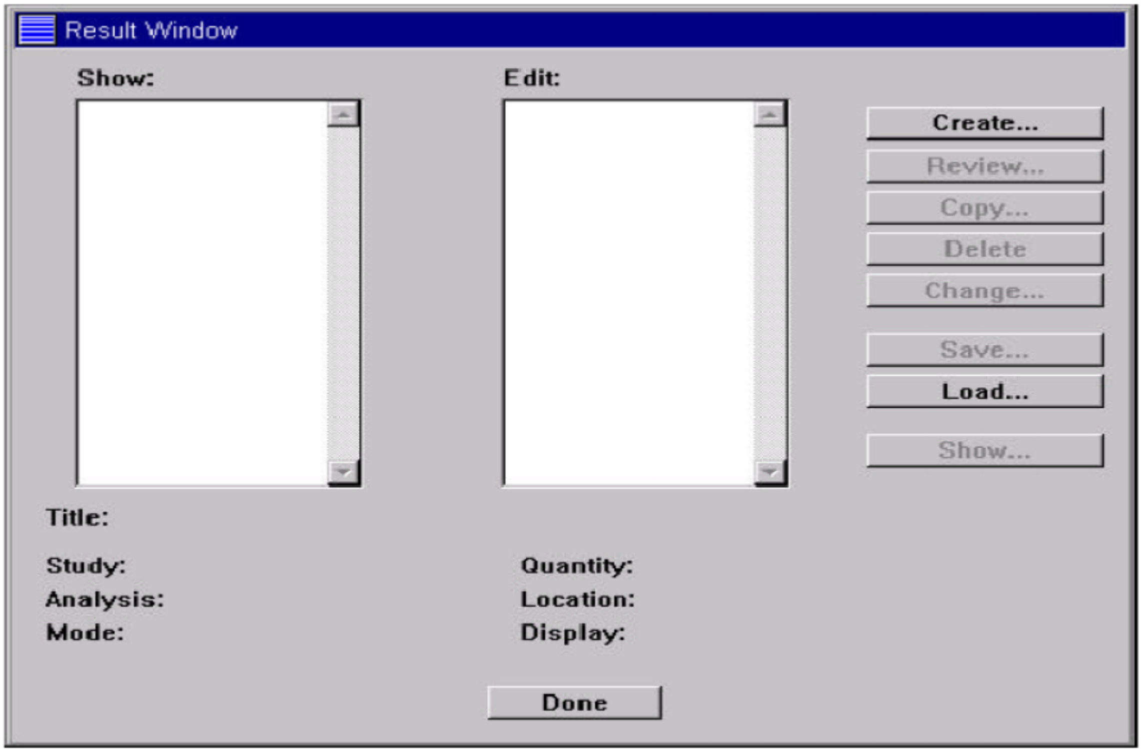
You have now set the constraints, set the loads and the material properties. Hence you are ready to enter the second subdivision under Structure - Analysis.

b. Analysis: Click on ***Analysis*** and you will see a window as shown.



Now after it has finished the analysis, click on ***Summary***.

d. **Results:** Having run the analysis you should now be able to view your results. Click on ***Results*** and a window pops up as shown below.



- Click on ***Create*** and name the first result window as "***vonmises stress"***.
- Select the directory as ***anlys1*** and click ***Accept***. You will see a window as shown below.