printed 25 set 16

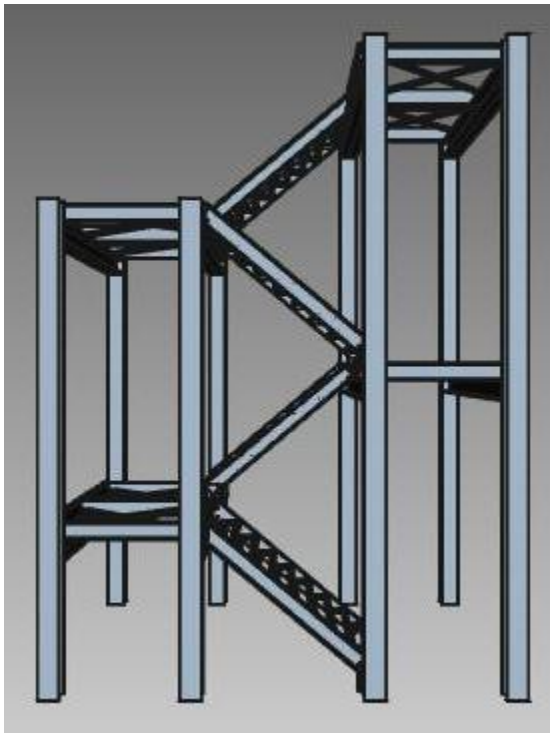
PYPETOOLS is the new workbench added to the Flamingo set.
As the name may suggest, it mainly deals with tubes, elbows, flanges and similar. It's a direct continuation of the FrameTools job but there is also a fundamental difference: it does not work (only) on Arch's *Structure* objects but it works on customized classes of objects (i.e. "Pipe", "Elbow" and "Flange": more on this later) defined inside *pipeFeatures.py*. That's the reason of the "y".
This don't mean that you won't be able to intersect beams with the command "extendTheTubes2intersection()", as well, or move any object that has a circular edge with "rotateTheTubeEdge()" (you CAN actually do that) but for my purposes and future development I preferred to define my own simplified "FeaturePython" objects.

So the workflow of PYPETOOLS reduces to two point:

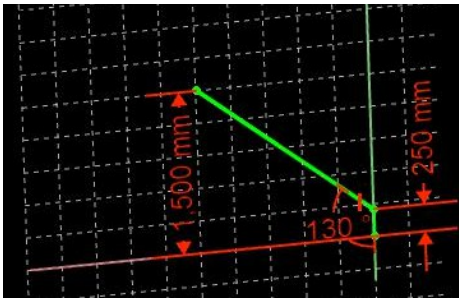
1. Create with ***Sketch*** or ***Draft*** workbenches the basic route of the piping or with ***Part*** and ***PartDesign*** create an auxiliary solid whose edges will be the axis of it.
2. With ***PypeTools*** workbench insert piping elements and modify them as needed.

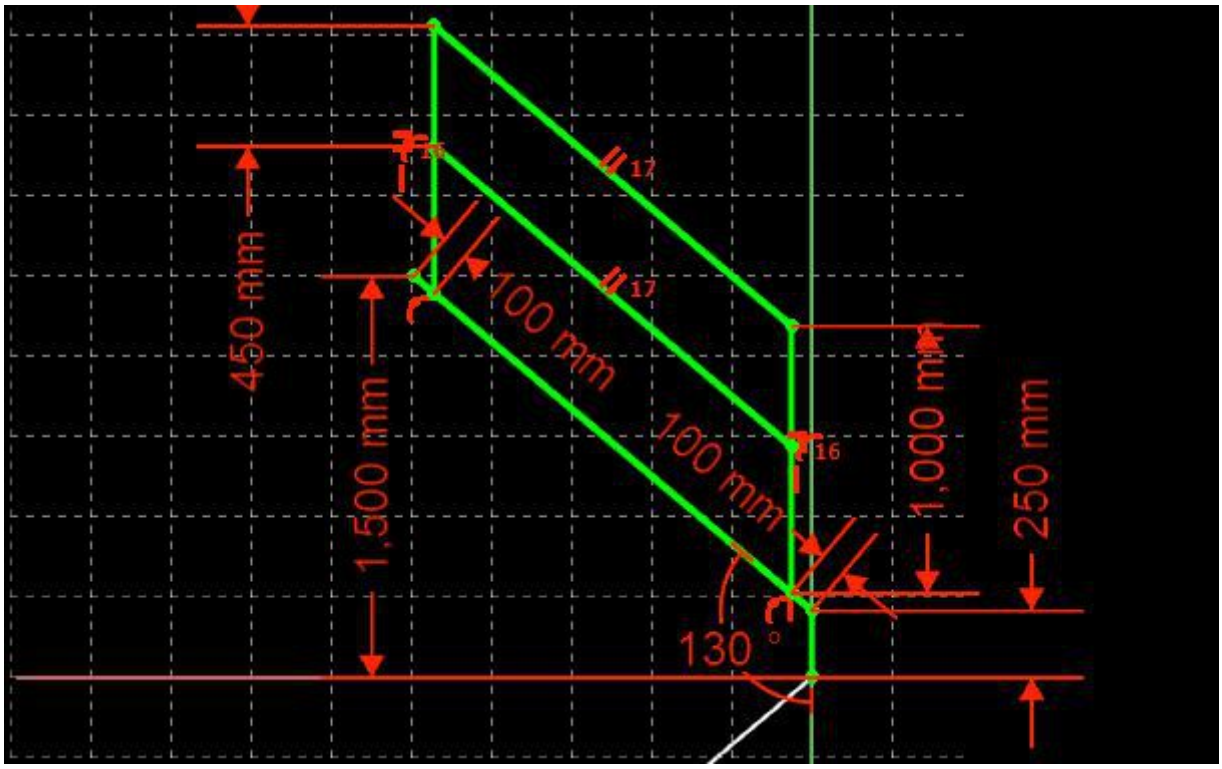
Now, for example it's worth to recall the tutorial of FRAMETOOLS where we have drawn a kind of firefighting stair. At the end it didn't look so safe because I got the feeling that something was missing... the handrails!

In this tutorial we are going to draw them with the aid of the new workbench; it is not the real task it is designed for but actually handrails are made by structural pipe, so here we go.

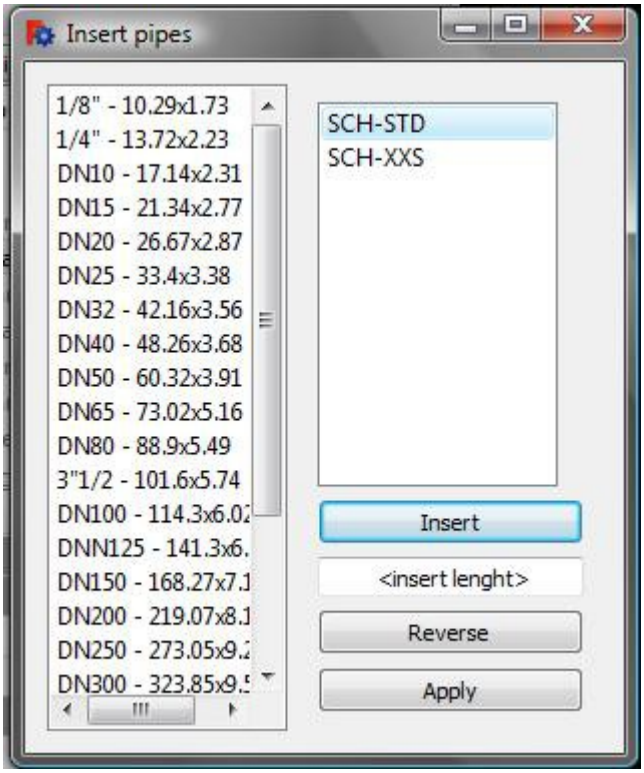


First we need to add some lines to the sketches of the ramp of stairs. Here to the right there's a picture of one of them with the relevant dimensions: so let's add to it the sketch of the handrail in order to get the outline as below.

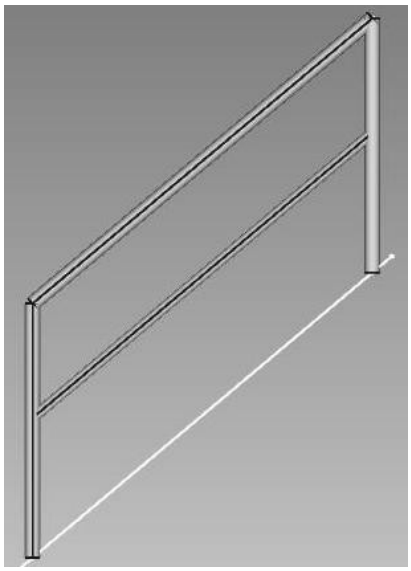




Obviously to lay down a tube over the axes we had just sketched it's necessary to use the command "Insert a tube", that you can execute pressing on the button of the toolbar or selecting the command from the menu.
Doing that, opens the dialog to the left.
This is designed in order to have on its left the list of available sizes for the tube and to the right the list of available "schedula" numbers. Changing the selection in the right list will update pipes Outside Diameters and Thicknesses in the left list.
Actually these list are taken from a .csv file inside the *./tables* folder of the module and you can create your own tables of pipes, just following some simple rules.



- The files must be comma-separated-values type, with semicolon as delimiter.
- The names of the file must start with the kind of piping object ("Pipe" in this case) followed by an underscore and the name of it's pressure-rating class, that will be shown in the right list of the dialog above.
- The file must contain the table of dimensions of tubes, being the first row of the table the names of properties as defined in the relevant class in *pipeFeatures.py*: read the automatic documentation of classes with the menu path *Help -> Automatic Python documentation...*



It's simpler to do rather than say: just copy one of the existing files in that folder, rename it and change values: next time you'll open the dialog you will find in it your brand new class of pipes.

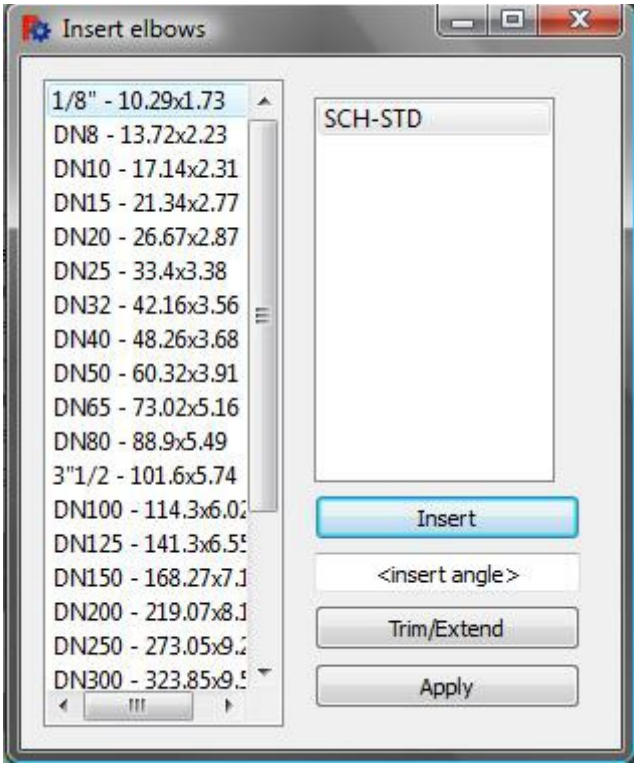
Also, it's clear, you can modify the dimensions of pipes directly from the properties tab after you created it, without need that its dimensions are present in one of the .csv files.

So select DN50 / SCH-STD and select the two vertical and the top diagonal lines of the sketch; press the "Insert" button. Repeat the operation for the lower diagonal line but with DN32 size. This is what you get.

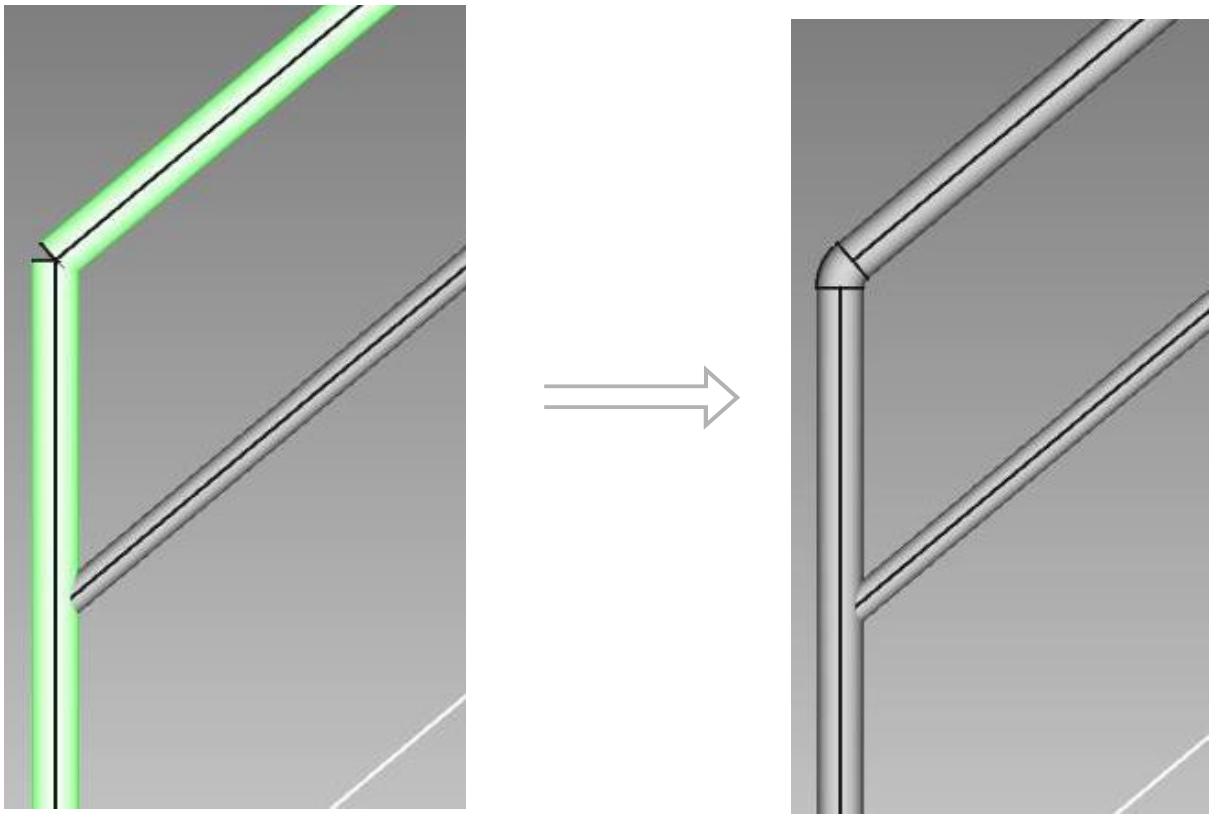
To complete briefly the description of this dialog:

- The "Apply" button will change the dimension of selected tubes in the viewport to that of the selected size in the left list.
- The "Reverse" button will change the orientation of the pipe by 180°, in case it's needed in some case: for example when you insert a tube over a circular edge (read automatic documentation for all options!).

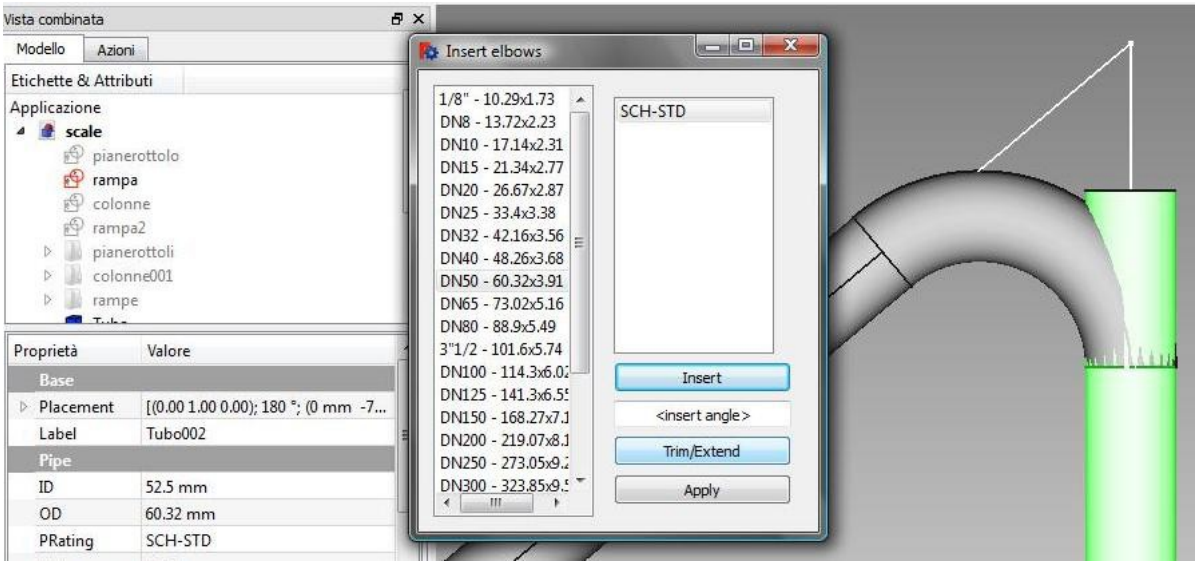
To insert the elbow elements where the pipes intersect with an angle, use the "Insert an elbow" command: that pop-ups the dialog to the right, which is very similar to that of "Insert a pipe".
In fact it works quite in the same way: there are two list for the sizes and classes of elbows, linked to the file in the *./tables* folder; there is the "Insert" and "Apply" button, but the "Reverse" button is replaced with an handy "Trim/Extend" that can be used to adjust pipe terminations when they are not trimmed automatically by the "Insert" command. For example this happens when you modify the bending radius after you created the curve.

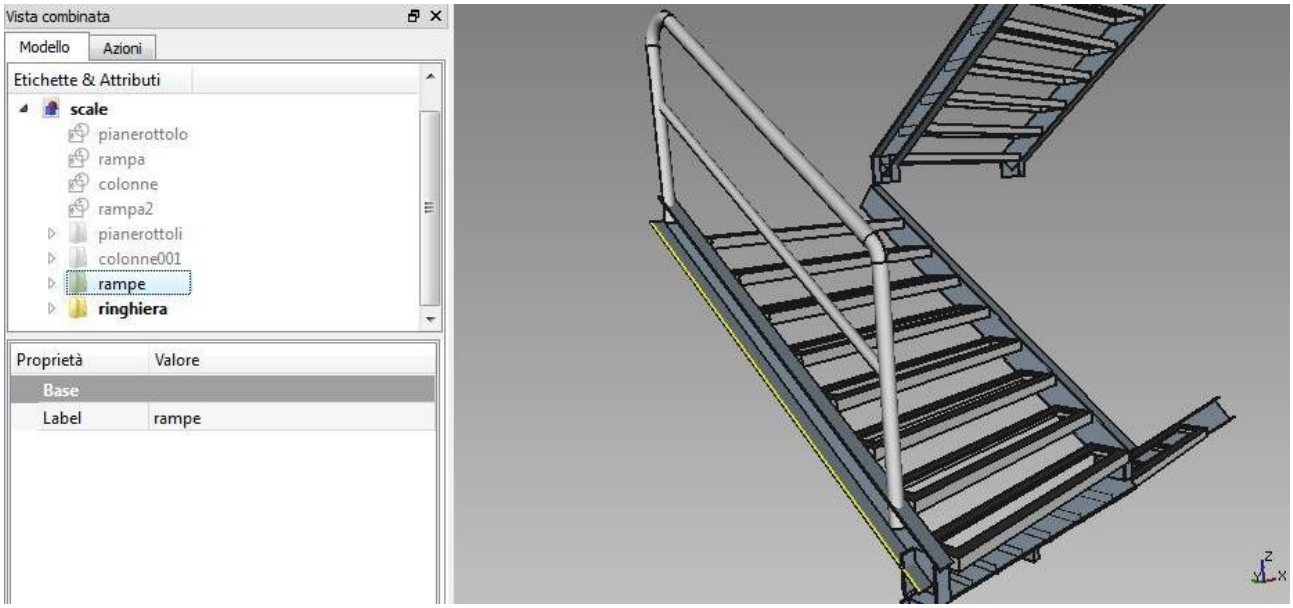


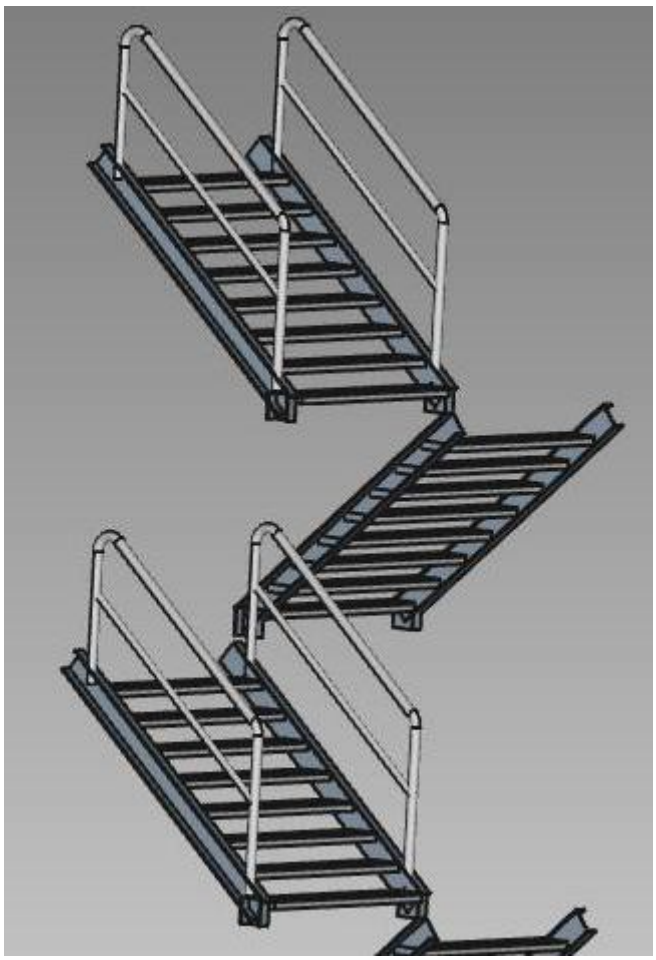
So, select first the two pipes that form the lower intersection like below and press insert. You see that an elbow is created at the intersection of tubes and these latter are trimmed to the edges of the elbow.
Notice also that you don't need to select one size from the list of the dialog because the command, if at least one pipe is selected, seeks the list to find the same size of the pipe.



We make the same for the upper intersection but let's consider that for estethic reason we want to increase the bending radius. As said above, we can make this from the properties tab and then trim the tubes to the new edges with the dedicated button of the dialog.







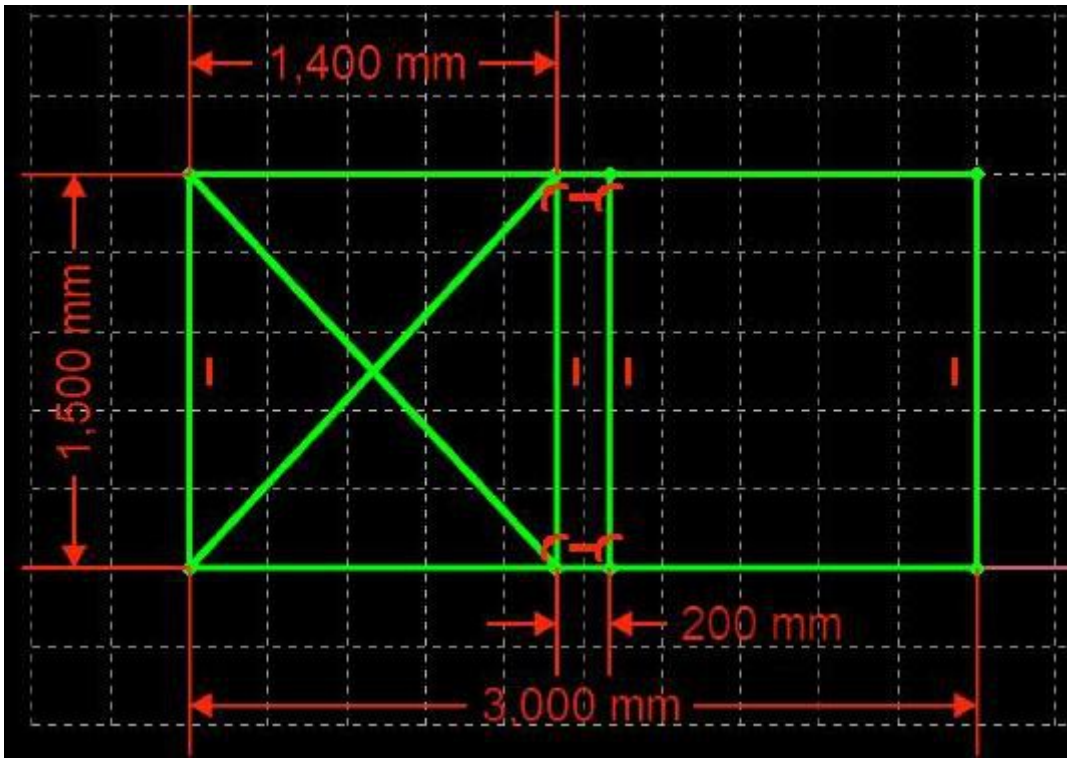
So, here we have our first handrail placed on the side of one of the ramps of stairs. At this point it's worth to create a group in the model tree and drag in there all the tubes we created up to now; as said in the other tutorial this is not mandatory but it helps to keep the file clean and readable.

After that we can use our old friend from FRAMETOOLS workbench: the *shiftTheBeam* command.

With that it's possible to move and copy the handrail to the other beams of ramps, using the dimensions of the model to measure the distance of translation. See "tutorialFrame.pdf" for details.

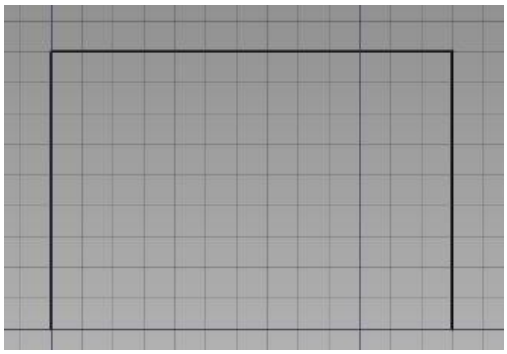
At the end the drawing will look like at side.

You can draw similarly the other handrails but I want to show an alternative method of doing that.
To draw the handrails on the intermediate floors first consider their dimensions from the previous tutorial:

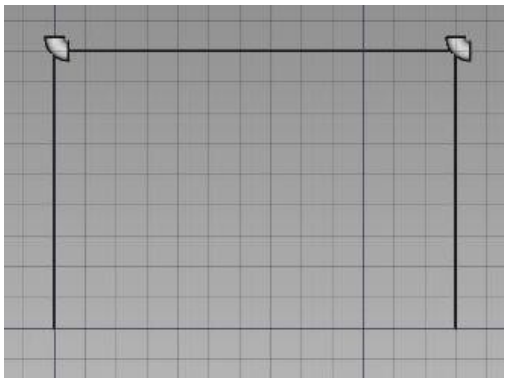


So I'm going to create on a separate file an handrail 1300 mm long and then import it in the drawing of the stair and copy/move it as necessary.

Open a new file and draft the shape of a portal on it, using DWire object for a change, 1300 mm long and 1000 mm high.



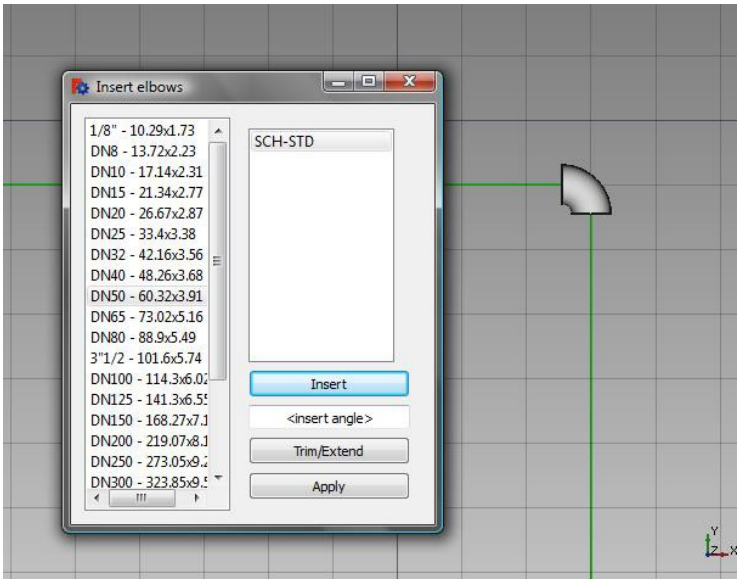
Then create two DN50 curves on the two top vertexes. This is an alternative way to insert elbows: select one vertex, select DN50 in the left list of the dialog and press insert. If you don't specify a different angle in the textbox, a 90° curve with a default orientation will be placed over the vertex. Repeat that for the other vertex.

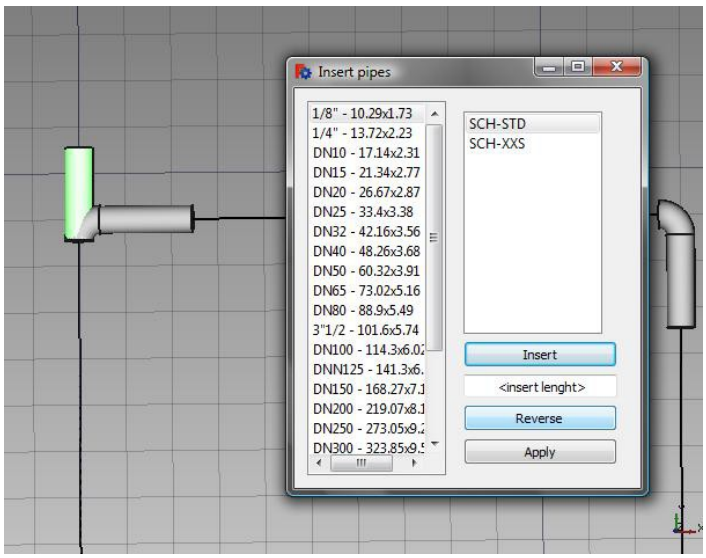


To orient the two elements according the lines of the portal you may use the spinTheBeam command from FRAMETOOLS.

Actually in the code the function spinTheBeam() has been replaced with the more flexible rotateTheTubeAx(), whose full features are exploited in the relevant command and dialog.

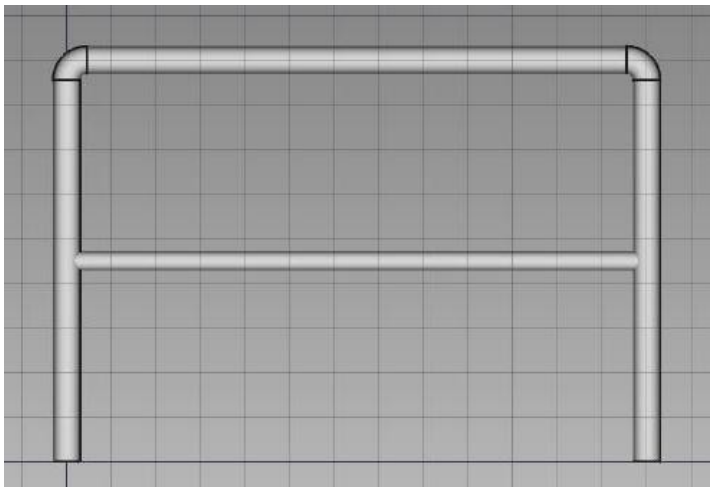
Otherwise you may have selected two edges before pushing "Insert": in that case the curve would has been inserted like shown at side.

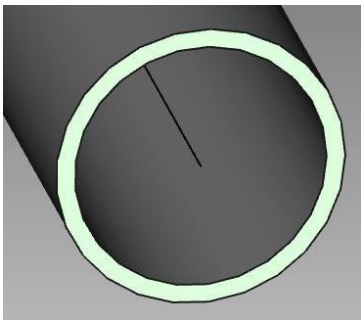




Whatever method you use, now the task is to insert pipes on the circular edge of the curves just created. This is as simple as select at once each edge and press "Insert" on the dialog for "Insert a tube". As for curves, the size of the pipe is automatically choosen according to the size of the curves selected in the viewport. Default Height is 200 mm but you can also insert a different value.
According to the edge orientation, the tube may not be positioned as you would expect; but in that case you can select the tube and click on "Reverse" button.

Extend the horizontal tube to the edge of the opposite elbow and the vertical tubes to the bottom vertex of one of vertical edges.
Draw also the second orizontal line and create the smaller DN32 horizontal tube. In this way you have quickly completed the handrail structure.

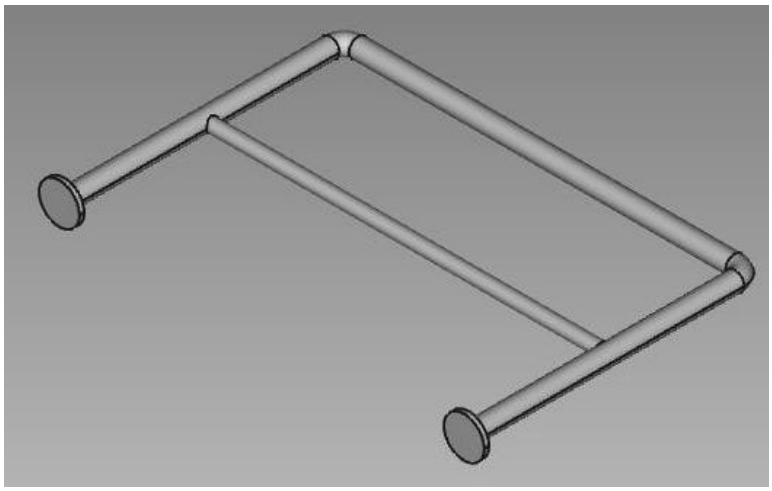




Since we have some spare time we can also make the feet of the handrail: first select all tubes we created and fuse them with the appropriate command of *Part* workbench.
Then select one of the visible crowns of the tube to create a sketch on it.

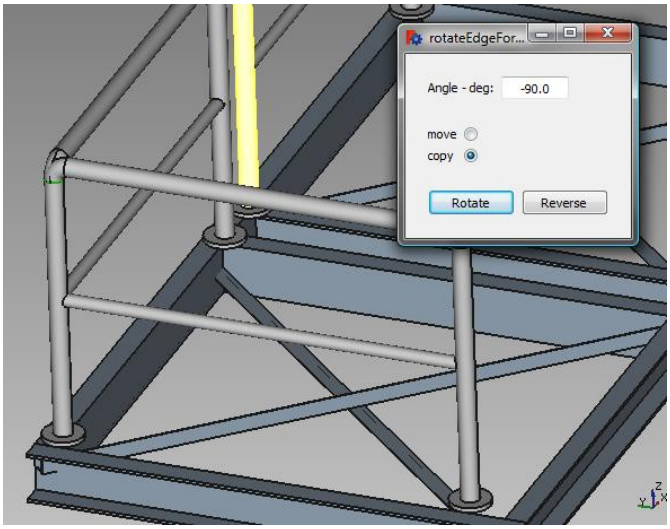
The sketch sholuld look as below. Purple circle are the edges of the pipe projected on the sketch.

Extrude the sketch and the handrail is ready to be imported in the main project file.

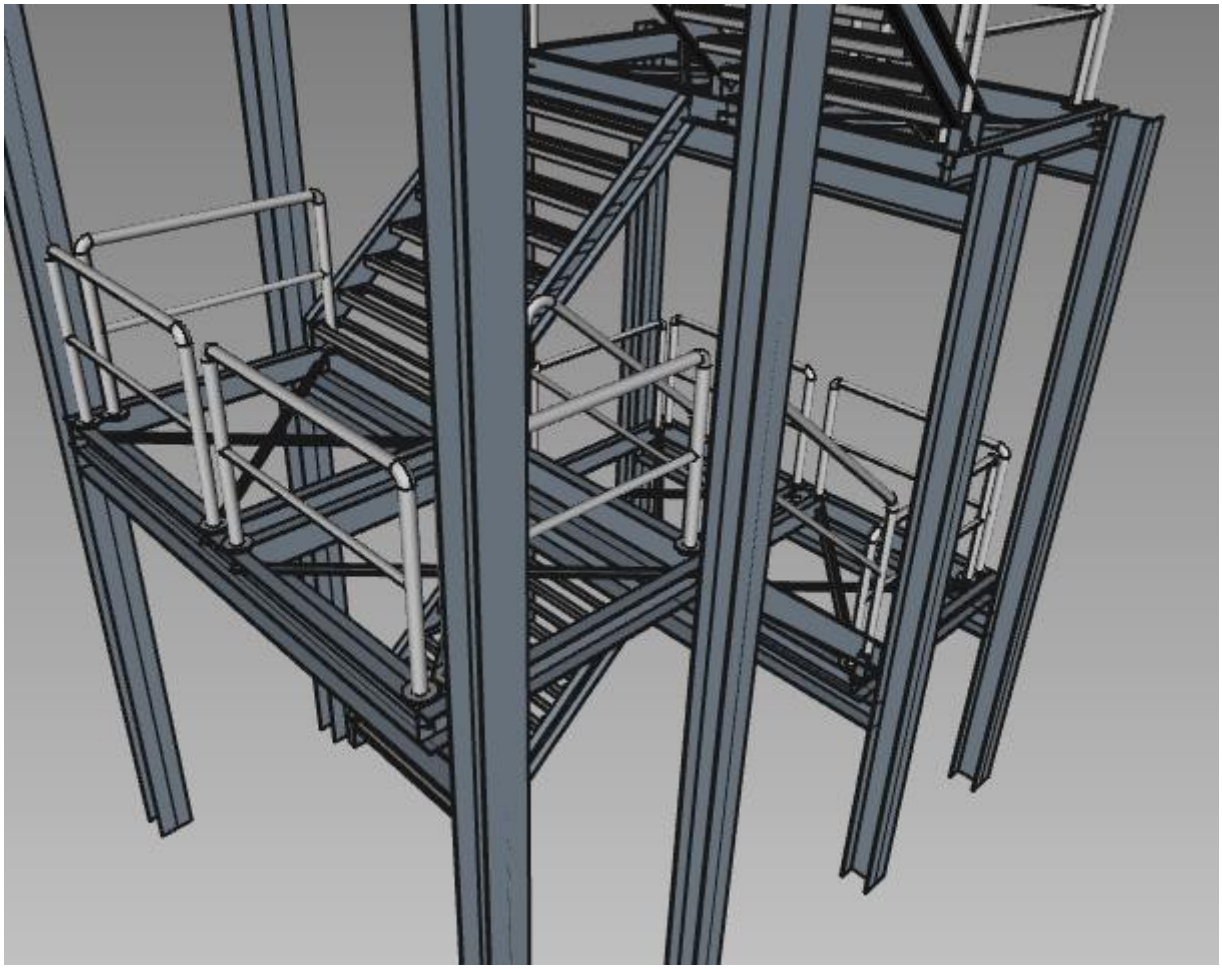


I think I can skip all the operations of orienting the handrails according to a face, translating and copying because they are already described in another tutorial.
It's worth to mention only the "Rotate through edge" command, used to create a copy of the handrail at 90°.

Intuitively, with this tool you are able to rotate one object around the axis of one circular edge of it. So, as shown in the picture, select one of the circular edges on the vertical tube of the handrail, insert 90 for the angle of rotation and select the "copy" radio button. If the direction of rotation is not the one that you expected you can use the "reverse" button.



So we've done! Except that there is no grating on the floors, now our fire-fighting stair looks a bit more safe.



The files of this tutorial are in the *./example* folder and in next page is a brief description of other commands available in PYPETOOLS workbench (up to now).

Other commands:

Inserts a flange. The dialog is designed similar as for the insertion of pipes and elbows. Table of flanges are customizable in *./tables* folder as well.

Mates two circular edges of two separate objects.

This command is used to place a floating curve on the plane defined by two edges or beams or pipes. It actually make the Z axis of the Shape of the elbow parallel to the normal of the plane and translate its Placement in the intersection point.

Extend two pipes or beams to their intersection point, if exists.

Notes:

- PYPETOOLS workbench is under heavy development, so the list of commands will be updated time by time.
- My target is to save backward compatibility for all commands and objects but at this time I can not assure that it will be possible in all cases.
- I made an effort to provide good documentation within the code. Thus, for command descriptions you can browse the automatic Python documentation at the CommandsPipe.py and pipeForms.py pages.