Inheritance 2017

Nesting software

By

Hrishikesh Suslade

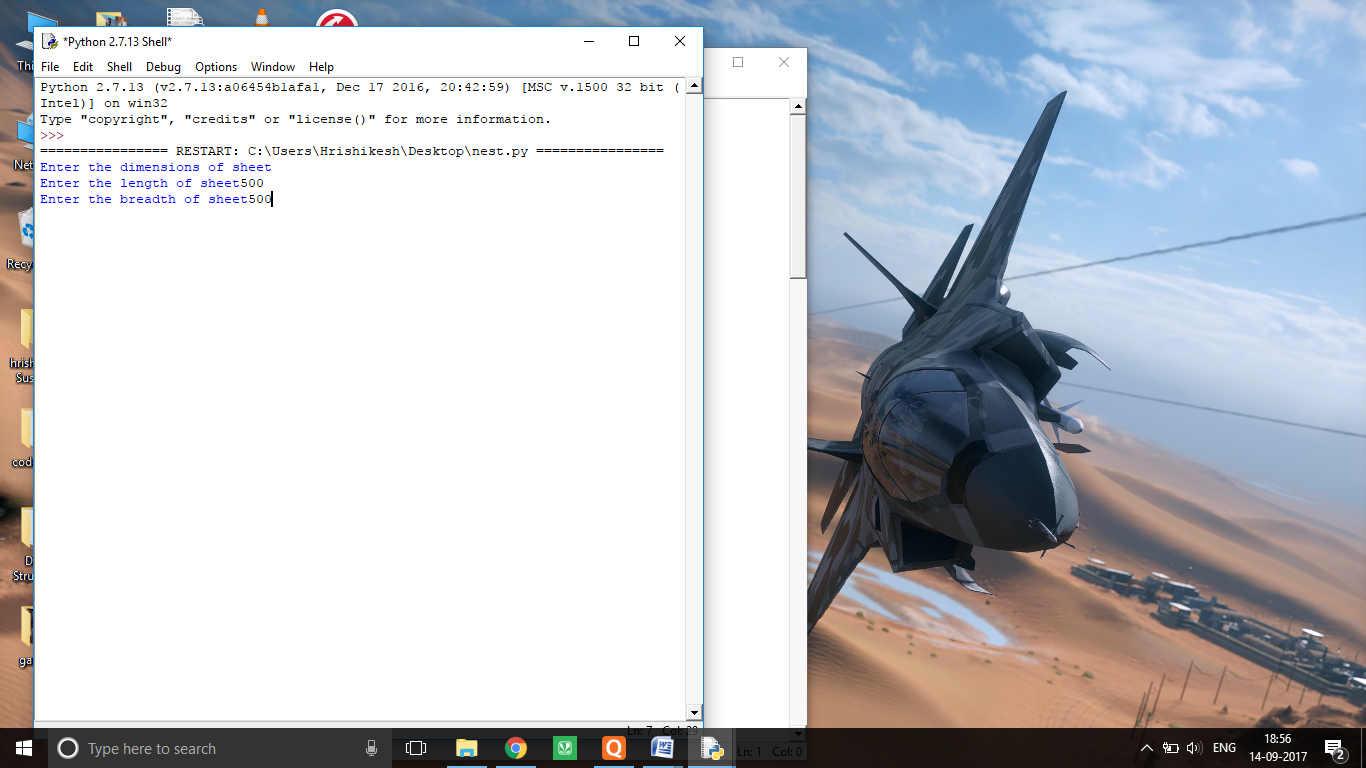
SY IT

**Aim :** To fit as maximum number of shapes in minimum area.

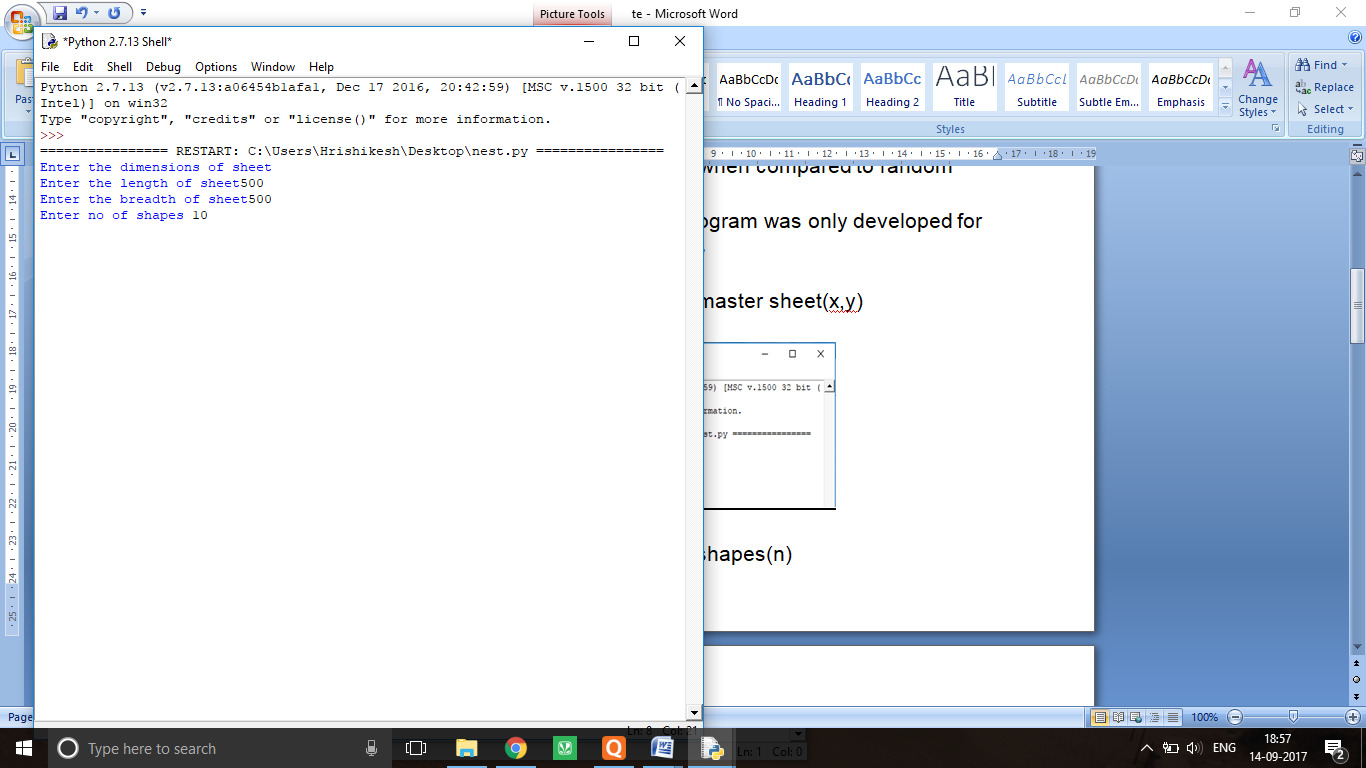
**Language used:** Python 2.7.3(num.py, graphics.py)

**Algorithms used:**

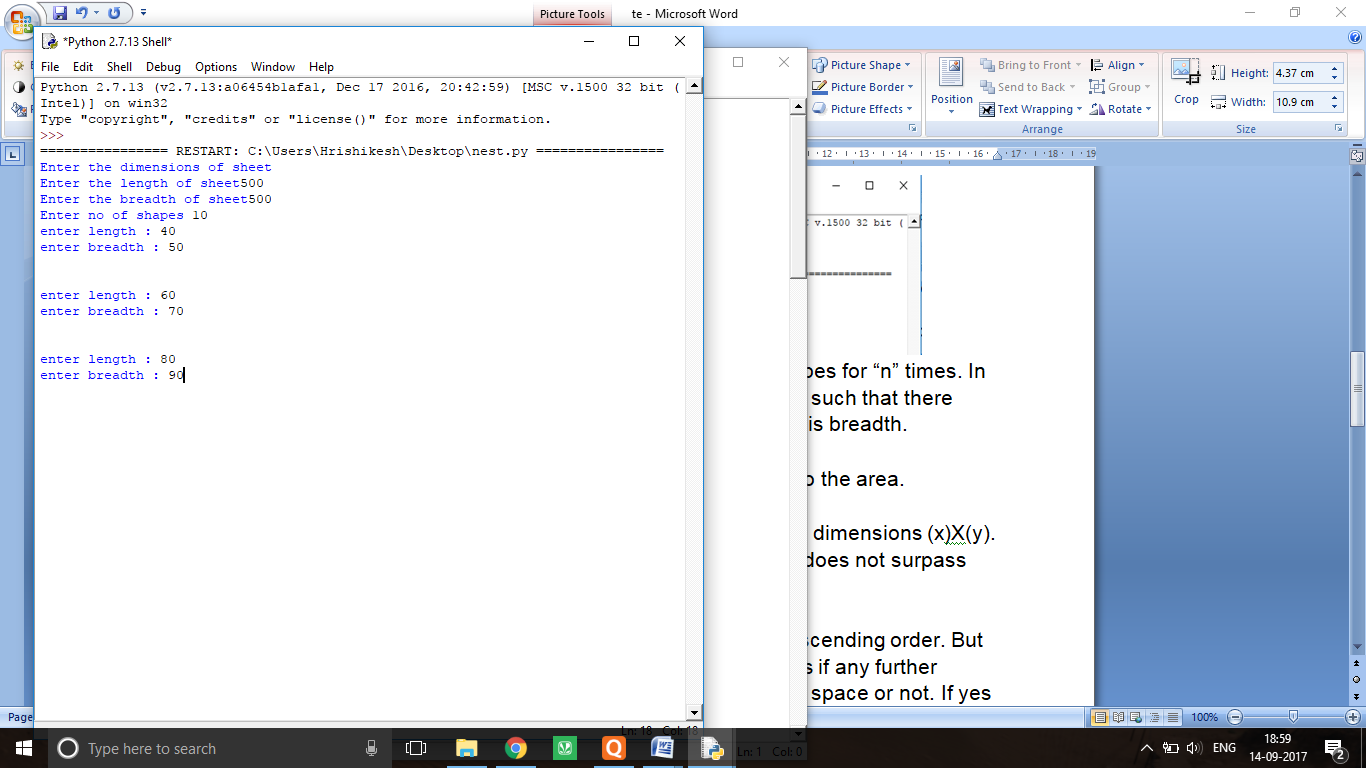
* There were a number of algorithms available for executing this task but implementing them was not possible due to high complexity
* Due this reason we developed a technique which is not highly efficient but is efficient when compared to random cutting or placing of shapes
* Due to high complexity the program was only developed for optimizing rectangular shapes
* **Step-1 :** Input the size of the master sheet(x,y)

****

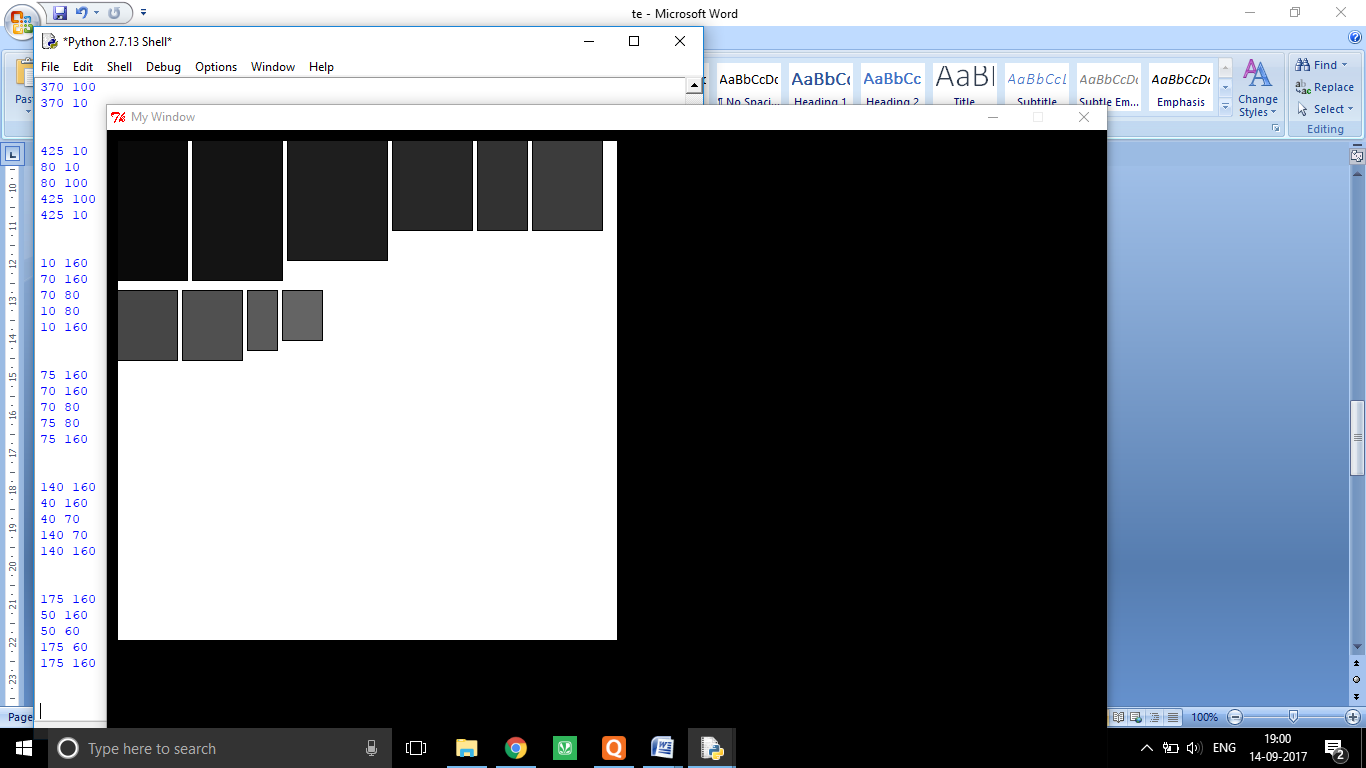
* **Step-2 :** Input the number of shapes(n)

****

* **Step-3 :** Input the dimensions of the shapes for “n” times. In this step only the rectangles are oriented such that there longer side is the height and shorter one is breadth.

****

* **Step-4 :** Sort the Rectangles according to the area.
* **Step-5:** Start placing them on a sheet on dimensions (x)X(y).With the condition that the sum of areas does not surpass the area of master sheet.
* **Step-6:** The rectangles are placed in descending order. But when it reaches the left extreme it checks if any further rectangle can be placed in the remaining space or not. If yes it will place the shape and remove it from list if not that area will be skiped.



Note: Small gaps are left between the shapes with the intention of making it easier to cut.

**Future Plans:**

* Currently the program can only optimize Rectangular shapes that also till a very low extent
* Future optimization includes modifying such that it can optimize complex shapes also with high efficiency

**Applications:**

* It can be used to cut shapes efficiently so that wastage of material is minimum
* We have also created a robot in which the co-ordinates of the optimized shapes can be sent by a programmer. It can efficiently and accurately draw those shapes on any material. Making the cutting job easier and efficient.