*Course Project*

By- Dhruv Tarwadi TY D 69 Yash Sali TY D 48 Jay Maniyar TY D 07

**Box Stacking Algorithm**

Brief Explanation

You are given a set of n types of rectangular 3-D boxes, where the ith box has height h(i), width w(i) and depth d(i) (all real numbers). You want to create a stack of boxes which is as tall as possible, but you can only stack a box on top of another box if the dimensions of the 2-D base of the lower box are each strictly larger than those of the 2-D base of the higher box. Of course, you can rotate a box so that any side functions as its base. It is also allowable to use multiple instances of the same type of box.

**Code (Python)-**

class Box:

def \_\_init\_\_(self, h, w, d):

self.h = h

self.w = w

self.d = d

def \_\_lt\_\_(self, other):

return self.d \* self.w < other.d \* other.w

def maxstackheight(arr, n):

rot = [Box(0, 0, 0) for \_ in range(3 \* n)]

index = 0

for i in range(n):

rot[index].h = arr[i].h

rot[index].d = max(arr[i].d, arr[i].w)

rot[index].w = min(arr[i].d, arr[i].w)

index += 1

rot[index].h = arr[i].w

rot[index].d = max(arr[i].h, arr[i].d)

rot[index].w = min(arr[i].h, arr[i].d)

index += 1

rot[index].h = arr[i].d

rot[index].d = max(arr[i].h, arr[i].w)

rot[index].w = min(arr[i].h, arr[i].w)

index += 1

n \*= 3

rot.sort(reverse=True)

msh = [0] \* n

for i in range(n):

msh[i] = rot[i].h

for i in range(1, n):

for j in range(0, i):

if rot[i].w < rot[j].w and rot[i].d < rot[j].d:

if msh[i] < msh[j] + rot[i].h:

msh[i] = msh[j] + rot[i].h

maxm = -1

for i in range(n):

maxm = max(maxm, msh[i])

return maxm

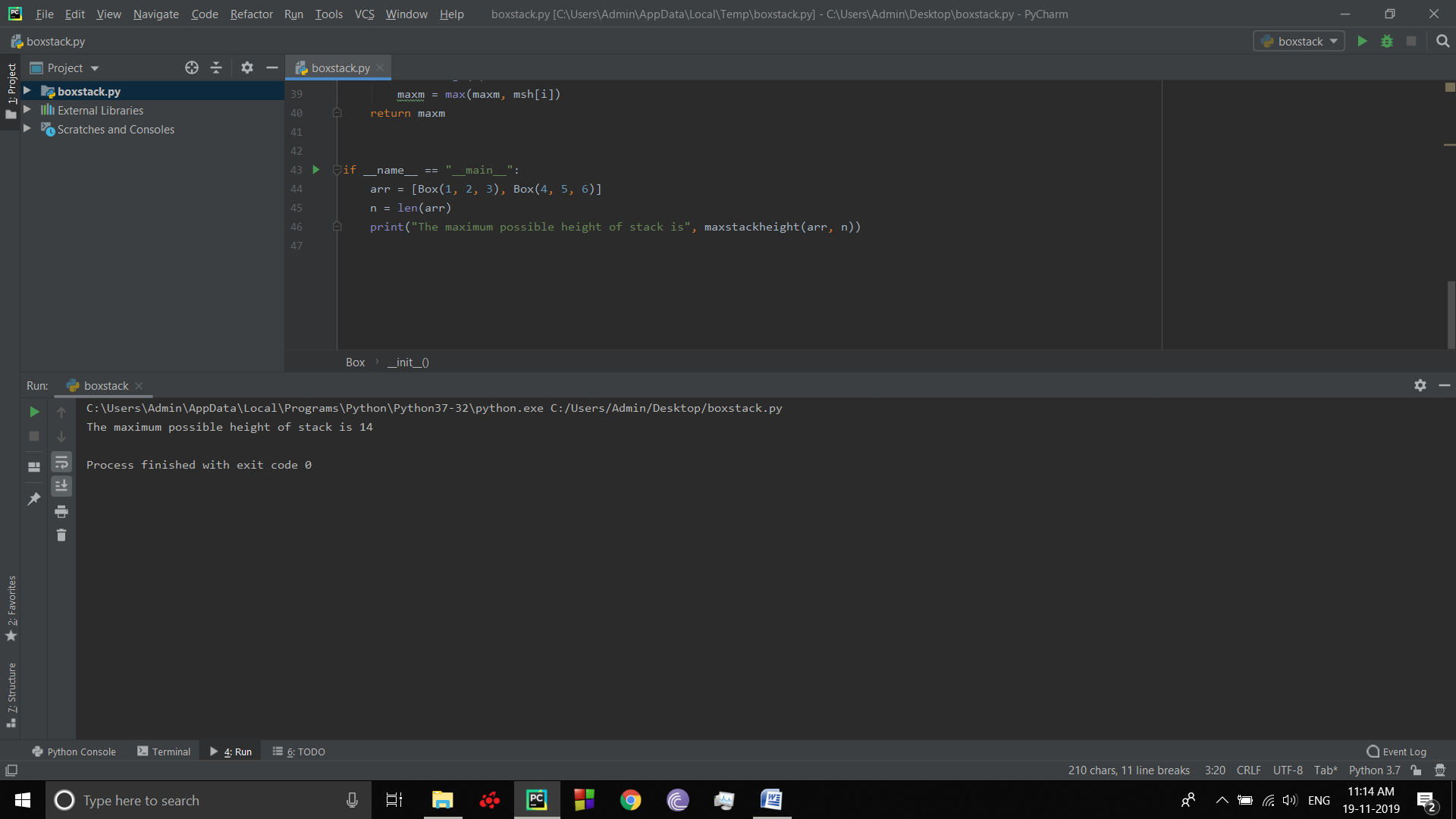
if \_\_name\_\_ == "\_\_main\_\_":

arr = [Box(1, 2, 3), Box(4, 5, 6)]

n = len(arr)

print("The maximum possible height of stack is", maxstackheight(arr, n))

Output-



Input and Output Respectively.