LAB 1

F190151

Ahmad Raza

19F-0151  [Company address]

**TASK NO 1:**

X="My Name is Ahmad"  
Y="My Roll no is 19F-0151"  
Z="I like AI Subject : A"  
print(X)  
print(Y)  
print(Z)

*Output:*

Graphical user interface, text, application

Description automatically generated

**TASK NO 2:**

def add(X,Y):  
 print("ADDITION : ")  
 print(X + Y)  
def subtract(X,Y):  
 print("Subtraction : ")  
 print(X - Y)  
def Multiply(X,Y):  
 print("Multiplication : ")  
 print(X \* Y)  
def Divide(X,Y):  
 print("Division : ")  
 print(X / Y)  
 print("Reminder is : ")  
 print(X % Y)  
def Square(X):  
 print("Square of number 1 : ")  
 print(X \* X)  
def cube(X):  
 print("Cube of number 1 : " )  
 print(X \* X \* X)  
def Square2(Y):  
 print("Square of number 2 : ")  
 print(Y \* Y)  
def cube2(Y):  
 print("Cube of number 2 : " )  
 print(Y \* Y \* Y)  
X= float(input("Enter the first number "))  
Y= float(input("Enter the second number "))  
add(X,Y)  
subtract(X,Y)  
Multiply(X, Y)  
Divide(X, Y)  
Square(X)  
cube(X)  
Square2(X)  
cube2(X)

*Output:*

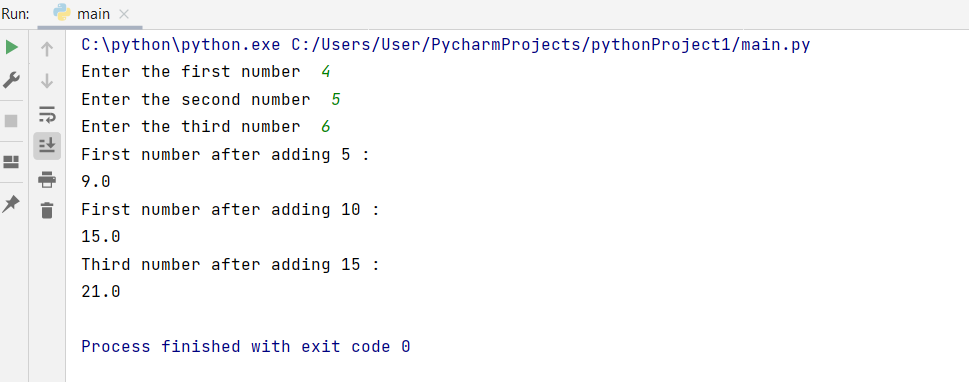
Graphical user interface, text, application, email

Description automatically generated

**TASK NO 3:**

def ADD(X,Y,Z):  
 print("First number after adding 5 : ")  
 print(X + 5)  
 print("First number after adding 10 : ")  
 print(Y + 10)  
 print("Third number after adding 15 : ")  
 print(Z + 15)  
  
  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 X = float(input("Enter the first number "))  
 Y = float(input("Enter the second number "))  
 Z = float(input("Enter the third number "))  
 ADD(X,Y,Z)

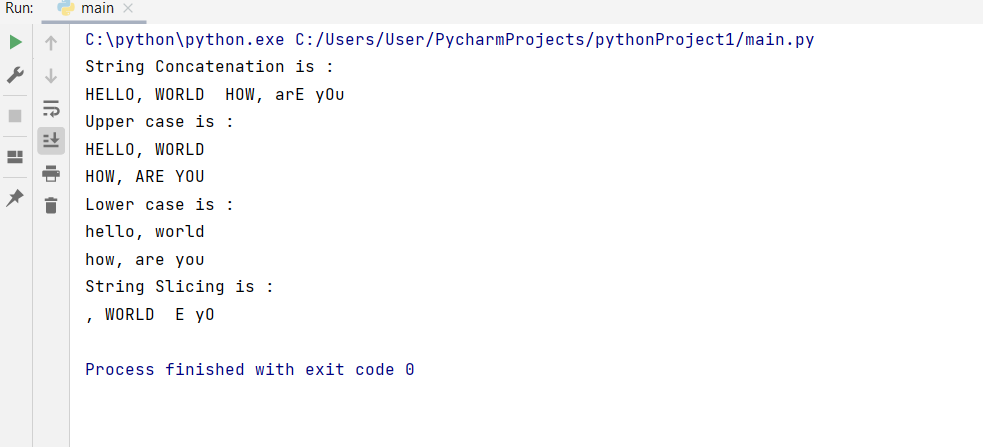
*Output:*



**TASK NO 4:**

def concatenate(str1, str2):  
 print("String Concatenation is : ")  
 print(str1 + " " + str2)  
  
  
def case(str1, str2):  
 print("Upper case is : ")  
 print(str1.upper())  
 print(str2.upper())  
 print("Lower case is : ")  
 print(str1.lower())  
 print(str2.lower())  
  
  
def slicing(str1, str2):  
 print("String Slicing is : ")  
 x=str1[5:12]  
 y=str2[-5:-1]  
 print(x + " " + y)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 str1 = "HELLO, WORLD"  
  
 str2 = "HOW, arE yOu"  
  
 concatenate(str1, str2)  
 case(str1, str2)  
 slicing(str1, str2)

*Output:*



**TASK NO 5:**

X = int(input("Enter starting number : "))  
Y = int(input("Enter ending number : "))  
  
def even(X, Y):  
 print("Even Numbers are ")  
 num = 0;  
 for x in range(X,Y):  
 while num <= Y:  
 if (num % 2 == 0):  
 print(num)  
 num = num + 1  
  
def odd(X,Y):  
 num=0;  
 print("Odd Numbers are ")  
 for x in range(X,Y):  
  
 while num <= Y:  
 if (num % 2 != 0):  
 print(num)  
 num = num + 1  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 even(X, Y)  
 odd(X, Y)

*Output:*

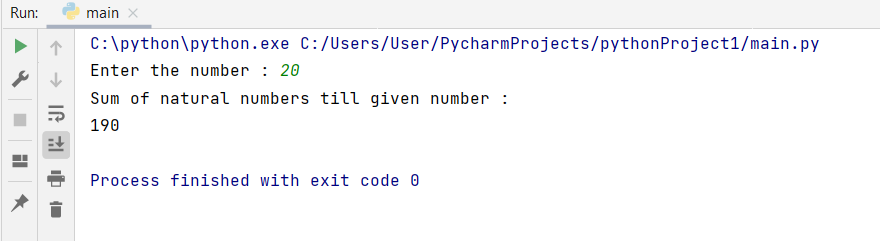
Graphical user interface, text, application, email

Description automatically generated

**TASK NO 6:**

X=int(input("Enter the number : "))  
def sum(X):  
 num=1  
 sum=0  
 print("Sum of natural numbers till given number : ")  
 while num != X:  
 sum += num  
 num=num+1  
 print(sum)  
if \_\_name\_\_ == "\_\_main\_\_":  
 sum(X)

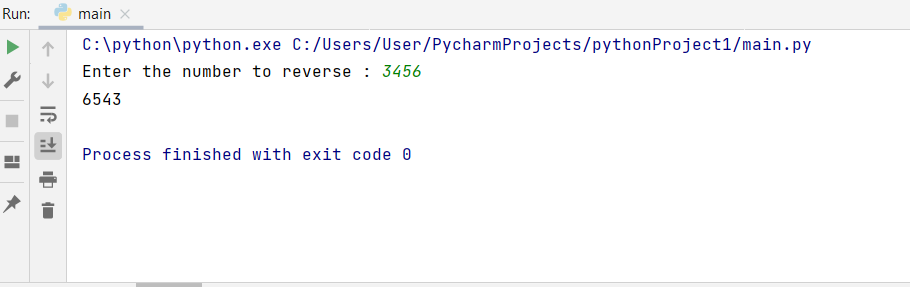
*Output:*



**TASK NO 7:**

def reverse(num1):  
 num=0  
 while num1 != 0:  
 x = num1 % 10  
 num = num \* 10 + x  
 num1 //= 10  
  
 print(num)  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 num1 = int(input("Enter the number to reverse : "))  
 reverse(num1)

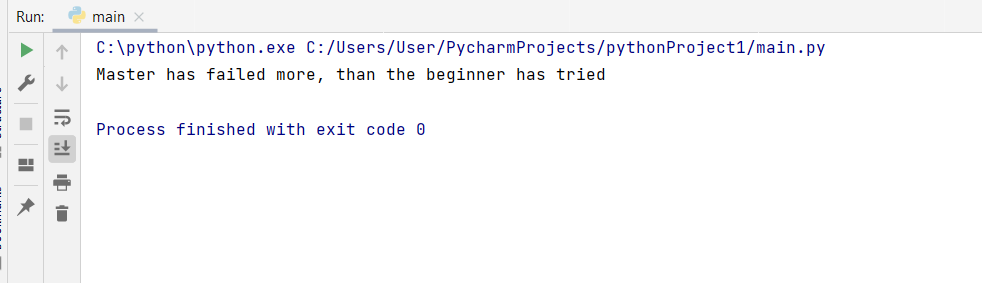
*Output:*



**TASK NO 8:**

string = "Master has failed more, than the beginner has tried"  
  
def str(string):  
 string1 = string[0:22]  
 string2 = string[22:51]  
 print(string1 + string2)  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 str(string);

*Output:*



**TASK NO 9:**

list = ["apple", "cherry", "orange", "kiwi", "melon", "mango"]  
  
  
def func(list):  
 list.pop(1)  
 list.pop(3)  
 list.insert(3,"banana")  
 print(list)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 func(list)

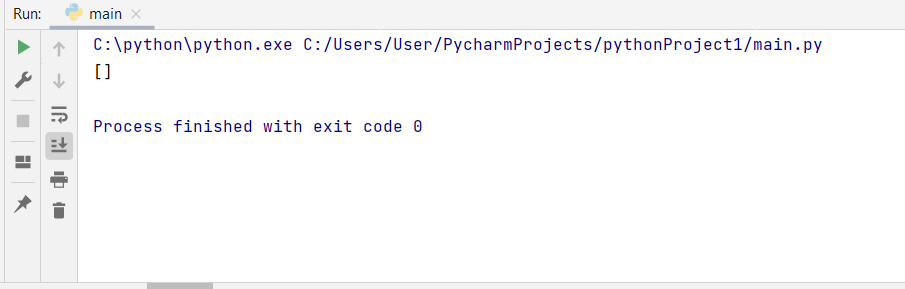
*Output:*



**TASK NO 10:**

list = [ 1, 4, 56, 2, 4 , 12, 6, 89 ,11, 0]  
  
  
def empty(list):  
 num=9;  
 while num >= 0:  
 list.pop(num);  
 num=num-1;  
 print(list);  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 empty(list)

*Output:*



**TASK NO 11:**

marks = {'AI': 74,'CN': 76,'DS': 42,'PS': 54}  
  
def sum(marks):  
 sum=0;  
 print("Sum is : ")  
 for var in marks.values():  
 sum=sum+var  
 print(sum)  
  
 print("Keys : ")  
 for x in marks:  
 print(x)  
  
 print("Values : ")  
  
 for x in marks:  
 print(marks[x])  
  
 print("Maximum value with key is : ")  
  
 maxkey = max(marks, key=marks.get)  
 print(maxkey)  
  
 maxvalue = max(marks.values())  
 print(maxvalue)  
  
  
 print("Minimum value with key is : ")  
  
 maxkey = min(marks, key=marks.get)  
 print(maxkey)  
  
 maxvalue = min(marks.values())  
 print(maxvalue)  
  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 sum(marks)

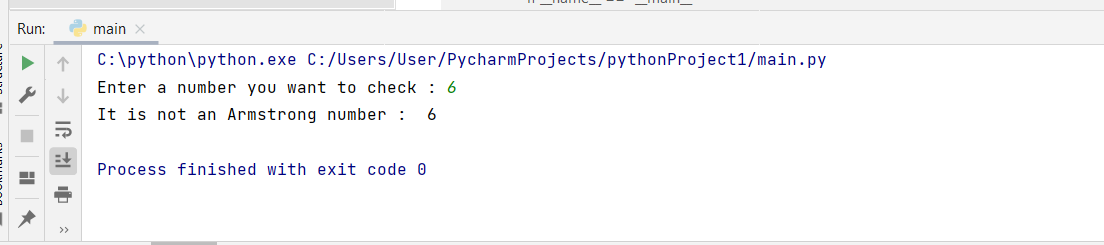
*Output:*



**TASK NO 12:**

def Armstrong():  
 var = int(input("Enter a number you want to check : "))  
 sum = 0  
 temp = var  
 while temp > 0:  
 value = temp % 10  
 sum =sum+ value \*\* 3  
 temp //= 10  
  
 if var == sum:  
 print("It is an Armstrong number : ",var)  
 else:  
 print("It is not an Armstrong number : ",var)  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 Armstrong();

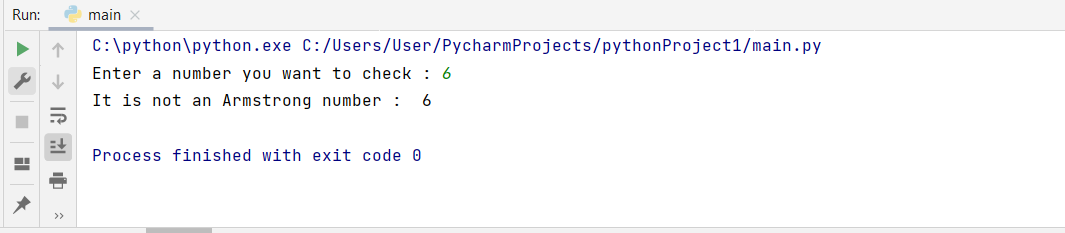
*Output:*



**TASK NO 13:**

num = int(input("Enter the number you want to check : "))  
  
def fun(var):  
 if var == 1:  
 return var  
 else:  
 return var\*fun(var-1)  
  
  
if num == 0:  
 print("The factorial of 0 is 1")  
else:  
 print("The factorial of is",fun(num))  
  
 if \_\_name\_\_ == "\_\_main\_\_":  
 fun(num)

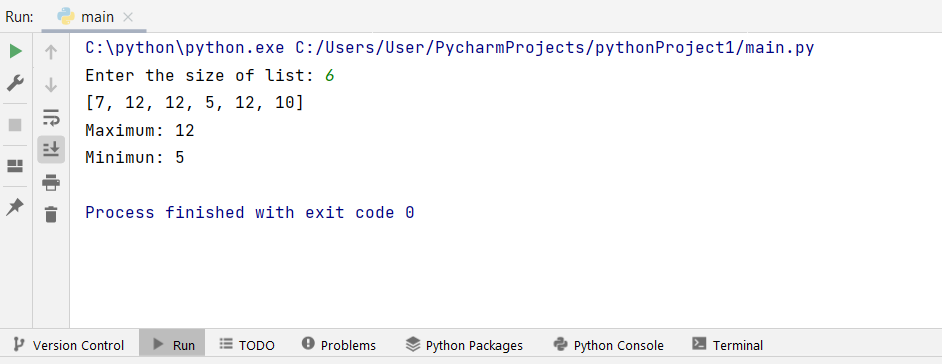
*Output:*



**TASK NO 14:**

import random  
var = int(input("Enter the size of list: "))  
list =[]\*var  
  
def rand(list):  
 for x in range(var):  
 list.append(random.randint(1,15))  
 print(list)  
 max=0  
 min=16  
 for x in range(var):  
  
 if(max<list[x]):  
 max=list[x]  
  
 if (min>list[x]):  
 min=list[x]  
 print("Maximum is :", max)  
 print("Minimun is :", min)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 rand(list)

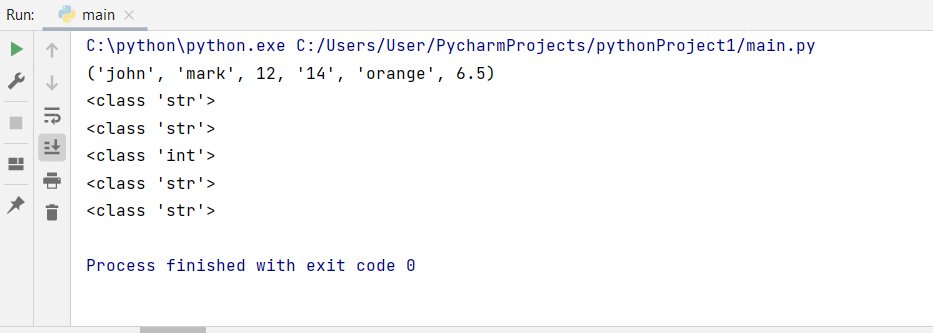
*Output:*



**TASK NO 15:**

tup = ("john", "mark", 12, "14" , "orange", 4.5)  
  
def edit(tup):  
 x=float(6.5)  
 y = list(tup)  
 y.remove(4.5)  
 y.append(x)  
 tup = tuple(y)  
 print(tup)  
  
def prnt(tup):  
 for x in range (5):  
 print(type(tup[x]))  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 edit(tup)  
 prnt(tup)

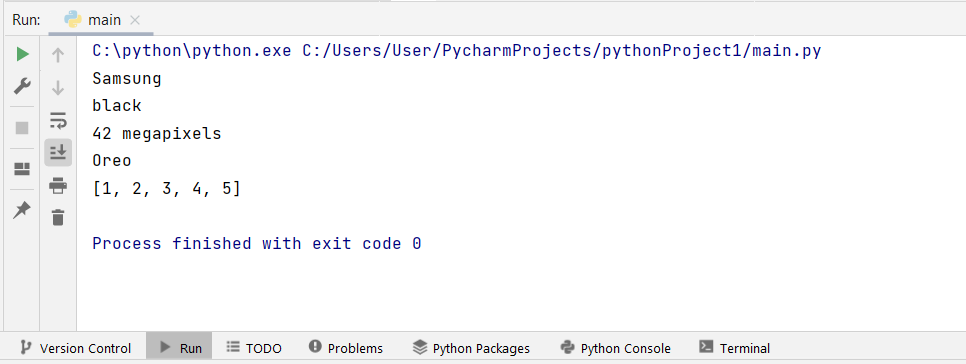
*Output:*



**TASK NO 16:**

Dictionary = {"brand":"Samsung","color":"black","camera":"42 megapixels","OS-type":"Oreo", "year":2012}  
  
  
def fun(Dictionary):  
  
 Dictionary["sizes"] = [1,2,3,4,5]  
 Dictionary.pop("year")  
 for x in Dictionary:  
 print(Dictionary[x])  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 fun(Dictionary)

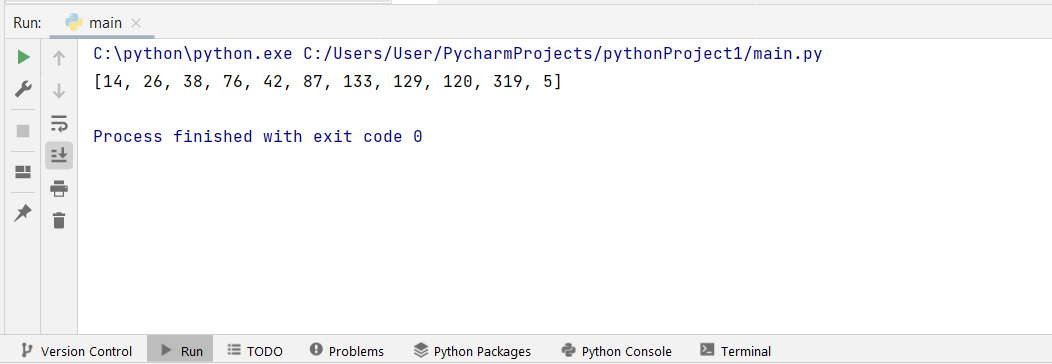
*Output:*



**TASK NO 17:**

list1 = [11,22,33,44,21,54,67,54,33,222,4]  
list2 = [3,4,5,32,21,33,66,75,87,97,1]  
  
def fun(list1, list2):  
 list3=[]\*11  
 list3 = [list1[i] + list2[i] for i in range(len(list1))]  
 print(list3)  
 return list3;  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 fun(list1, list2)

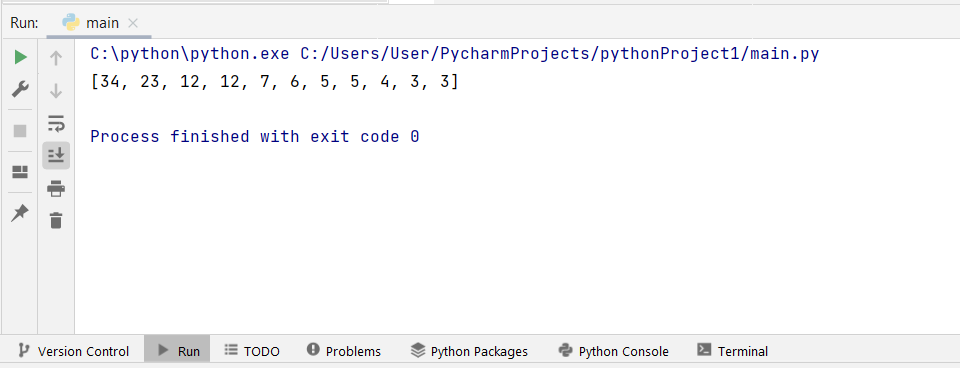
*Output:*



**TASK NO 18:**

list = [ 5, 6, 7, 23 ,12 ,3, 3, 4 ,5, 12, 34]  
  
  
def sorting(list):  
 list.sort(reverse=True)  
 print(list)  
 return list  
  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 sorting(list)

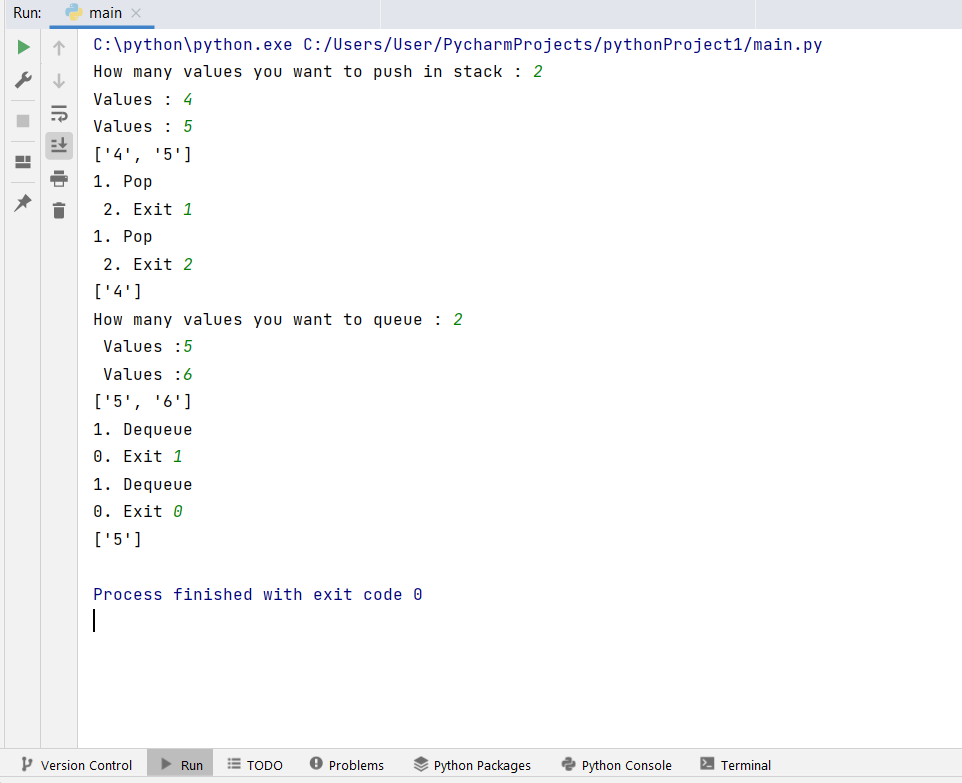
*Output:*



**TASK NO 19:**

stack=[]  
queue=[]  
def queuefun(queue):  
 var1=int(input("How many values you want to queue : "))  
  
 for x in range(var1):  
 queue.append(input(" Values :"))  
  
  
 print(queue);  
  
  
def dequeue(queue):  
 var2=1;  
 while var2 != 0:  
 x=int(input("1. Dequeue\n0. Exit "))  
 if(x==1):  
 queue.pop()  
 else :  
 break  
 print(queue)  
  
def push(stack):  
 var1=int(input("How many values you want to push in stack : "))  
  
 for x in range(var1):  
 stack.append(input("Values : "))  
  
 print(stack);  
  
def pop(stack):  
 num1=1;  
 while num1 != 0:  
 var2=int(input("1. Pop\n 2. Exit "))  
 if(var2==1):  
 stack.pop()  
 else :  
 break  
 print(stack)  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 push(stack);  
 pop(stack);  
 queuefun(queue);  
 dequeue(queue);

*Output:*



**TASK NO 20:**

class Student:  
 Age = 0  
 FirstName = ""  
 Lastname = ""  
 CNIC = 0  
 Courses = []  
 Gender = ""  
 CGPA = 0.0  
 SGPA = 0.0  
 CH = 0  
 def \_\_init\_\_(self,Firstname,Lastname,Age,CNIC,Courses,Gender,CGPA,SGPA,CH):  
 self.Age = Age  
 self.FirstName = Firstname  
 self.Lastname = Lastname  
 self.CNIC = CNIC  
 self.Courses = Courses  
 self.Gender = Gender  
 self.CGPA = CGPA  
 self.SGPA = SGPA  
 self.CH = CH  
  
  
res = Student("Ahmad","Raza",21,376454362,["AI" , "WEB"] , "Male" , 3.5,3.3,17)  
print(res.Courses)

*Output:*

Graphical user interface, text, application

Description automatically generated

**TASK NO 22:**

import math  
class shape():  
 def \_\_init\_\_(self, classname):  
 self.classname = classname  
  
 def display(self, classname):  
 if classname == 'c':  
 print("Object called by Circle", self.classname)  
 elif classname == 'r':  
 print("Object called by Rectangle", self.classname)  
  
class circle () :  
 def \_\_init\_\_(self, radius):  
 self.radius = radius  
  
 def area(self):  
 a = math.pi \* (self.radius \*\* 2)  
 print("Area of circle:", round(a, 2))  
  
 def perimeter(self):  
 p = 2 \* math.pi \* self.radius  
 print("Perimeter of circle:", round(p, 2))  
  
class rectangle() :  
 def \_\_init\_\_(self, l, w):  
 self.l = l  
 self.w = w  
  
 def area(self, l, w):  
 a = l \* w  
 print("Area of Rectangle:", round(a, 2))  
  
 def perimeter(self, l, w):  
 p = 2\*(l \* w)  
 print("Perimeter of Rectangle:", round(p, 2))  
  
  
def main():  
 obj = shape('')  
 rad = float(input("Enter radius of circle: "))  
 c = circle(rad)  
 c.area()  
 c.perimeter()  
 obj.display('c')  
 print("\n")  
 len = float(input("Enter Length of rectangle: "))  
 w = float(input("Enter Width of rectangle: "))  
 obj.display('r')  
 r = rectangle(len,w)  
 r.area(len,w)  
 r.perimeter(len,w)  
  
main()

*Output:*

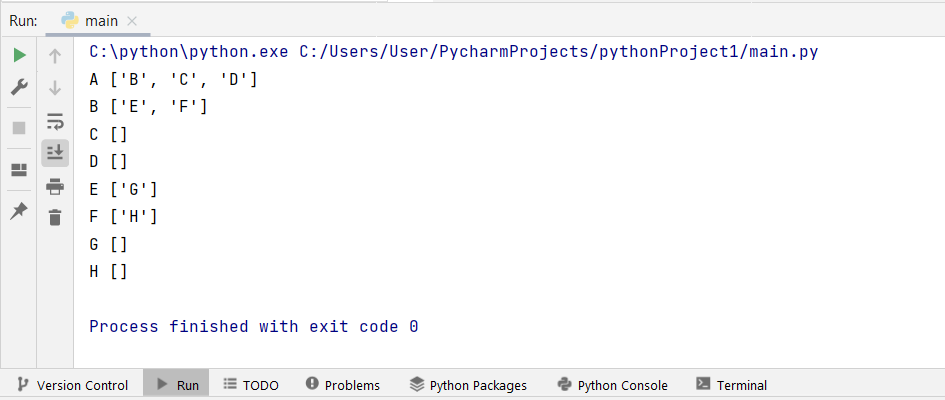
Graphical user interface, text, application

Description automatically generated

**TASK NO 24:**

class Vertex:  
 def \_\_init\_\_(self, n):  
 self.name = n  
 self.neighbors = set()  
  
 def add\_neighbor(self, val\_of\_vertex):  
 self.neighbors.add(val\_of\_vertex)  
  
  
class Graph:  
 vertices = {}  
  
 def add\_vertex(self, vertex):  
 if isinstance(vertex, Vertex) and vertex.name not in self.vertices:  
 self.vertices[vertex.name] = vertex  
 return True  
 else:  
 return False  
  
 def add\_edge(self, u, v):  
 if u in self.vertices and v in self.vertices:  
 self.vertices[u].add\_neighbor(v)  
 return True  
 else:  
 return False  
  
 def print\_graph(self):  
 for key in sorted(list(self.vertices.keys())):  
 print(key, sorted(list(self.vertices[key].neighbors)))  
  
  
g = Graph()  
for i in range(ord('A'), ord('I')):  
 g.add\_vertex(Vertex(chr(i)))  
edges = ['AB', 'AC', 'AD', 'BE', 'BF', 'EG', 'FH']  
for edge in edges:  
 g.add\_edge(edge[0], edge[1])  
g.print\_graph()

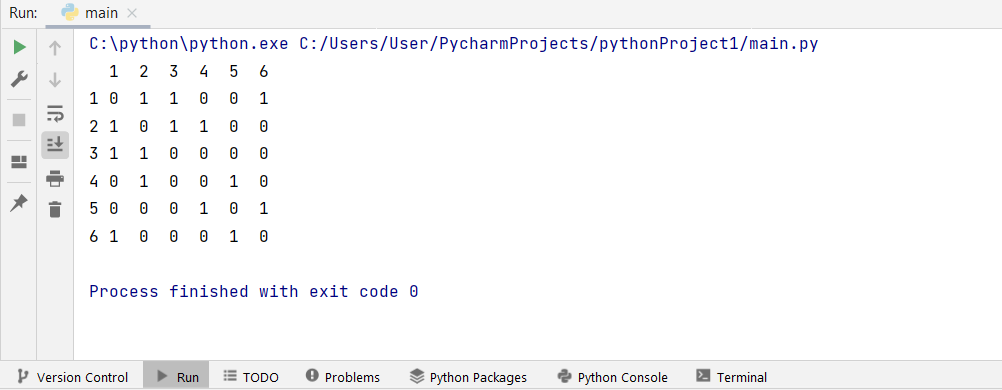
*Output:*



**TASK NO 25:**

class Vertex:  
 def \_\_init\_\_(self, name):  
 self.name = name  
  
  
class Graph:  
 vertices = {}  
 edges = []  
 edge\_indices = {}  
  
 def add\_vertex(self, vertex):  
 if isinstance(vertex, Vertex) and vertex not in self.vertices:  
 self.vertices[vertex.name] = vertex  
 for row in self.edges:  
 row.append(0)  
 self.edges.append([0] \* (len(self.edges) + 1))  
 self.edge\_indices[vertex.name] = len(self.edge\_indices)  
 return True  
 else:  
 return False  
  
 def add\_edges(self, u, v, weight=1):  
 if u in self.vertices and v in self.vertices:  
 self.edges[self.edge\_indices[u]][self.edge\_indices[v]] = weight  
 self.edges[self.edge\_indices[v]][self.edge\_indices[u]] = weight  
  
 return True  
 else:  
 return False  
  
 def print\_graph(self):  
 for x, y in sorted(self.edge\_indices.items()):  
 print(end=' ')  
 print(x, end='')  
 print('')  
 for v, i in sorted(self.edge\_indices.items()):  
 print(v + ' ', end='')  
 for j in range(len(self.edges)):  
 print(self.edges[i][j], end=' ')  
 print(' ')  
g = Graph()  
for i in range(ord('1'), ord('7')):  
 g.add\_vertex(Vertex(chr(i)))  
edges = ['12', '13', '16', '23', '24', '54', '56']  
for edge in edges:  
 g.add\_edges(edge[0], edge[1])  
g.print\_graph()

*Output:*



**TASK NO 26:**

|  |  |
| --- | --- |
| Double Shift | [Search Everywhere](https://www.jetbrains.com/help/pycharm/searching-everywhere.html)  Quickly find any file, action, class, symbol, tool window, or setting in PyCharm, in your project, and in the current Git repository. |
| Ctrl+Shift+A | [Find Action](https://www.jetbrains.com/help/pycharm/searching-everywhere.html#find_action)  Find a command and execute it, open a tool window, or search for a setting. |
| Alt+Enter | [Show Context Actions](https://www.jetbrains.com/help/pycharm/intention-actions.html)  Quick-fixes for highlighted errors and warnings, intention actions for improving and optimizing your code. |
| Ctrl+E | [View recent files](https://www.jetbrains.com/help/pycharm/navigating-through-the-source-code.html#recent_files)  Select a recently opened file from the list. |
| Ctrl+W  Ctrl+Shift+W | [Extend or shrink selection](https://www.jetbrains.com/help/pycharm/working-with-source-code.html)  Increase or decrease the scope of selection according to specific code constructs. |
| Ctrl+/  Ctrl+Shift+/ | [Add/remove line or block comment](https://www.jetbrains.com/help/pycharm/working-with-source-code.html#editor_lines_code_blocks)  Comment out a line or block of code. |

**TASK NO 27:**

**How to install library from terminal**

Text

Description automatically generated

**How to install library without using terminal**

* Files>Settings>Project>Python Interpreter

Graphical user interface, application

Description automatically generated

**TASK NO 28:**

**How to zoom in pycharm**

You can zoom through **CTRL+MOUSE WHEEL** option

**TASK NO 30:  
How to change theme in Pycharm**

Press control Alt S to open Appearance setting

Graphical user interface, text, application

Description automatically generated