*#program -1*x=8  
y=9  
z=x+y  
print(z)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#2-program for using range*a = range(1, 10)  
for i in a:  
 print(i)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#3\_program- Range*b=range(3,30,2)  
for j in b:  
 print(j)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program for list*thislist=["Apple","Mango","orange"]  
for n in thislist:  
 print(n)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program for using range in list*thislist = ["one" , "Two" , "Three" , "four"]  
for i in range(len(thislist)):  
 print(thislist[i])  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program for using while loop*thislist = ["ganga","Narmada","krishna"]  
i = 0  
while i < len(thislist):  
 print(thislist[i])  
 i = i + 1  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program for using loop comprehension*fruits = ["Banana","Mango","Apple","Avakoda"]  
newlist=[]  
for x in fruits:  
 if "a" in x:  
 newlist.append(x)  
 print(newlist)  
 print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program to sum all items in the list*x = [10,11,12,13]  
print("The item to be added", x)  
result = sum(x)  
print("The sum of all items are" , result)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#Program to multiply all items in a list  
#x = int[3, 4, 5, 6]*y=1  
for i in x:  
 y = y\*i  
print(y)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program to get the largest number in a list*items =[10,56,12,45,20]  
print("Entered elements are",items)  
print("The biggest element is")  
print(max(items))  
*#program to get the smallest number in a list*items =[10,56,12,45,20]  
print("Entered elements are",items)  
print("The smallest element is ", min(items))  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#7.program to find the frequency of a character in a string*str=input("Enter the string: ")  
d = dict()  
for i in str:  
 if i in d:  
 d[i]=d[i]+1  
 else:  
 d[i]=1  
print(d)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#8. program to count the number of strings where the string length is 2or more and first and last char is same.*y= ['abc','xyz','aba','1221']  
print("The given strings are : ", y)  
for i in y:  
 length=len(i)  
 print("The length of the strings are: ", length)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#Program for slicing in Strings*b= "Hello, World!"  
print("The given string is: " , b)  
print("Slicing from 2nd position: " ,b[2:])  
print("Slicing from start position: " ,b[:5])  
print("Slicing from backward position: " ,b[-5:-2])  
print("Slicing from backward position: " ,b[:-2])  
print("Slicing from backward position: " ,b[-3:-1])  
print("The slicing result is: ",b[4:2:-1])  
c="Welcome to scaler"  
print(" slicing from backward ", c[-16:-4])  
print(" slicing from backward ", c[-16:-4:2])  
print(" slicing from backward ", c[3:-7])  
print(" slicing from backward ", c[-11:-1:2])  
print(" slicing from backward ", c[-13:-1:2])  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program for slicing:*b= [40, 50, 20, 30, 90]  
print("The given list: ", b)  
c=b[1:3]  
print(c)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#next program*h=list(range(30,100,10))  
print("given Range ", h)  
j=list(range(10))  
print("Arrived range = ", j)  
print("After slicing-1", j[2:6:2])  
a=list(range(10))  
print("Arrived range = ", a)  
print("After slicing-2 ", a[0:8:3])  
a=list(range(10))  
print("After slicing-3 ", a[:-2])  
a=list(range(10))  
print("The Element are ", a)  
print("After slicing-4 ", a[:-2:2])  
a=list(range(10))  
print("The Element are ", a)  
print("After slicing-5 ", a[::4])  
a=list(range(10))  
print("The Element are ", a)  
print("After slicing-6 ", a[2:-2])  
a="Stuttgart"  
print("Given String: ",a)  
print("After slicing -7 " , a[2:-2])  
a="Stuttgart"  
print("Given String: ",a)  
print("After slicing-8 " , a[-2:])  
a=list(range(10))  
print("The Element are ", a)  
print("After slicing-9 ", a[2:3])  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program for set - -(19 july)*f={"red","blue","green","yellow"}  
print(f)  
f={"red","blue","green","yellow"}  
for c in f:  
 print(c)  
f={"red","blue","green","yellow"}  
print("red" in f)  
print("pink" in f)  
*#program using add method*j= {"apple","banana","cherry"}  
print("Given elements are: " ,j )  
j.add("orange")  
print("Result is ", j)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#Program to add the items from another set*set1 = {1,2,3,4,5}  
set2 = {6,7,8,9,9}  
print("set1 Elements: ",set1)  
print("set2 Elements: ",set2)  
set1.update(set2)  
print("combined list: " ,set1)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program to add list in set using update method*set1 = {"Red","blue","Green","yellow"}  
list1 = ["apple","orange","mango","banana"]  
print("Set items ",set1)  
print("list items ",list1)  
set1.update(list1)  
print("Result is " ,set1)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program to remove items*set1 = {"Red","blue","Green","yellow"}  
print("Given list of items: ",set1)  
set1.remove("Red")  
print("After removing,remaining items are: ",set1)  
print("Given list of items: ",set1)  
set1.discard("Red")  
print("After removing,remaining items are: ",set1)  
set3= {"apple","orange","mango","banana"}  
print("given elements are: ",set3)  
*#set3.remove("cherry")*print("Elements after using remove method", set3)  
set3.discard("cherry")  
print("Elements after using discard method", set3)  
print("```````````````````````````````````````````")  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program using pop method*set1 = {"Red","blue","Green","yellow"}  
x = set1.pop()  
print("pop element is: ",x)  
print("Elements after pop method",set1)  
print("```````````````````````````````````````````")  
  
*#program using del and clear method*r={2,4,6,8,10}  
print("Given Elements are: ",r)  
r.clear()  
print("Result after clear process: ",r)  
print("```````````````````````````````````````````")  
r={1,3,5,7,9}  
print("Given Elements are: ",r)  
del r  
print("Result after del process")  
print("```````````````````````````````````````````")  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
*#program for using for loop*t= ("chennai","mumbai","kolkatta","Goa")  
print("Given elements are: ",t)  
print("Result elements are")  
for g in t:  
 print(g)  
print("```````````````````````````````````````````")  
  
*#program for loop*t={4,5,6,7,8}  
print("Given Elements are: ",t)  
for i in t:  
 print("Result: ",i)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program using union and intersection method*t={2,4,6,8,10}  
r={1,3,5,7,8,6,9}  
print("elements in t: ",t)  
print("elements in r: ",r)  
f=t.union(r)  
print("Elements after union method : ",f)  
s = t.intersection(r)  
print("Elements after intersection method:",s)  
t.update(r)  
print("Elements after update method : ",t)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#Program for using dictionary*dict1={"Name":"Kumar","age":"24","education":"B.Sc"}  
print(dict1)  
print(dict1["Name"])  
print(dict1["age"])  
dict1["place"]="kadalur"  
print(dict1)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program -Assignment*str1="""  
 Twinkle, twinkle, little star,  
 How I wonder what you are!  
 Up above the world so high,  
 Like a diamond in the sky.  
 Twinkle, twinkle, little star,  
 How I wonder what you are """  
print("The result string", str1)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program to add new item in dictionary*print("Dictionary")  
car={ "brand":"ford",  
 "model":"Mustang",  
 "year":1964}  
x = car.values()  
print("Elements in the Dictionary: ",x)  
car["color"]="red"  
print("Updated elements: ",x)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program to reflect changes in the dictionary*print("Dictionary")  
car={ "brand":"ford",  
 "model":"Mustang",  
 "year":1964}  
x = car.items()  
print("Elements in the Dictionary: ",x)  
car["color"]="red"  
car["year"] = 2020  
print("Updated elements: ",x)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#Program to check the value*thisdict={ "brand":"ford",  
 "model":"Mustang",  
 "year":1964}  
if "model" in thisdict:  
 print("yes,'model' is one of the keys in thisdict dictionary")  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program to check the value*thisdict={ "brand":"ford",  
 "model":"Mustang",  
 "year":1964}  
if "brand" in thisdict:  
 print("yes,'brand' is one of the keys in thisdict dictionary")  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*# program*print("Dictionary")  
car={ "brand":"ford",  
 "model":"Mustang",  
 "year":1964}  
x = car.items()  
print("Elements in the Dictionary: ",x)  
car["color"]="red"  
car.update({"year":2022})  
print("Updated elements: ",x)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program to pop items*student={"Name":"Ram",  
 "std":"fifth",  
 "Admission-No":2013}  
x=student.items()  
print("Given elements are: ",x)  
print("Pop process start")  
student.pop("Admission-No")  
print("Result:",x)  
student["Admission-No"]=2013  
print(student["Admission-No"])  
print("Before popping process: ",student)  
student.popitem()  
print("After popping process: ",student)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program using del keyword*student={"Name":"Ram",  
 "std":"fifth",  
 "Admission-No":2013}  
x=student.items()  
print("Given elements are: ",x)  
print("Del process start")  
del student["std"]  
print("Result: ",student)  
print("clear process start")  
student.clear()  
print("After clear process: ", student)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program using for loop*student={"Name":"Ram",  
 "std":"fifth",  
 "Admission-No":2013}  
print("Elements in the list: ")  
for x in student:  
 print(student[x])  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program using values in dictionary*student={"Name":"Ram",  
 "std":"fifth",  
 "Admission-No":2013}  
print("Elements in the list: ")  
for x in student.values() :  
 print(x)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program using keys in dictionary*student={"Name":"Ram",  
 "std":"fifth",  
 "Admission-No":2013}  
print("Elements in the list: ")  
for x in student.keys() :  
 print(x)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program using both keys and values in dictionary*student={"Name-":"Ram",  
 "std-":"fifth",  
 "Admission-No-":2013}  
print("Elements in the list: ")  
for x,y in student.items() :  
 print(x,y)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program using copy method*student={"Name-":"Ram",  
 "std-":"fifth",  
 "Admission-No-":2013}  
print("Elements in the list: ")  
student1=student.copy()  
print(student1)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program using dict method*student={"Name-":"Ram",  
 "std-":"fifth",  
 "Admission-No-":2013}  
print("Elements in the list: ")  
student1=dict(student)  
print(student)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program using nested dictionary*myfamily = {  
 "child1":{  
 "name":"uma",  
 "age":8  
 },  
 "child2":{  
 "name":"latha",  
 "age": 4  
 }  
 }  
print(myfamily)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program using if statement*a = 33  
b = 200  
  
if b > a:  
 print("b is greater than a" ,b)  
  
 *#program for while loop* i - 1  
 while i < 1:  
 print(i)  
 print("Executing loop")  
 if i == 3:  
 break  
 i += 1  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
  
  
*#program - functions*def demo(name,age):  
 print(name,age)  
  
demo("Mary",20)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program*def funct1(v1,v2,v3):  
 print("\n",v1,"\n",v2,"\n",v3)  
  
funct1(20,40,60)  
def funct1(v1,v2):  
 print("\n", v1, "\n", v2, "\n")  
funct1(30,20)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#Program-*def calculation(a,b):  
 add=a+b  
 sub=a-b  
 return add,sub  
print("Additon and Subtraction values are")  
res=calculation(50,20)  
print(res)  
  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
*#program*def calculate(a,b):  
 return a+b,a-b  
  
add,sub= calculate(50,30)  
print("addition -",add, "Subtraction -",sub)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program*def show\_emp(a,b=9000):  
 *# b=9000* print("Name: ",a,"," "Salary: ",b)  
show\_emp("Ben", 12000)  
show\_emp("mary")  
  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#Program*def my\_function(food):  
 for x in food:  
 print("List of items - ",x)  
  
fruits=["apple","banana","mango","cherry"]  
my\_function(fruits)  
  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
*#Program*def check(s):  
 return s\*5  
print("First result" , check(4))  
print("Second result" ,check(5))  
print("Third result" , check(9))  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program*def even(list1):  
 list2=[]  
 for i in list1:  
 if i % 2 ==0:  
 list2.append(i)  
 return list2  
list2= even([1,2,3,4,5,6,7])  
print("Even numbers: ",list2)  
  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
*#program*def arithmetic(a,b):  
 add = a+b  
 sub = a-b  
 multiply= a\*b  
 division=a/b  
 return add,sub,multiply,division  
a,b,c,d = arithmetic(21,18)  
print("Addition ",a)  
print("Subtraction ",b)  
print("Multiply ",c)  
print("Division ",d)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program*def function1():  
 localvariable="use only inside the function"  
 print("local variable- ",localvariable)  
function1()  
globalvariable = "we can use it anywhere"  
print("globalvariable-",globalvariable)  
  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program*globalvar= 777  
def function1():  
 print("Value is ",globalvar)  
def function2():  
 print("value is ",globalvar)  
  
function1()  
function2()  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program-global variable*x=5  
def function1():  
 print("value of x is ",x)  
def function2():  
 global x  
 x =7  
 print("Value of x,after changing it's value- ",x)  
def function3():  
 print("value of x in function3- ",x)  
  
function1()  
function2()  
function3()  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program- Non local variable* def outer\_func():  
 x=77  
  
 def inner\_func():  
 nonlocal x  
 x = 90  
 print("value in inner\_function ",x)  
  
 inner\_func()  
 print("Value of x ", x)  
  
outer\_func()  
  
*#program-Keyword arguments*def message(fname,sname):  
 print("Hello ",fname,sname)  
  
message(fname="mary",sname='Moses')  
message(fname="lara", sname="Moses")  
  
*#program-Variable length arguments*def addition(\*number):  
 total = 0  
 for i in number:  
 total=total+i  
 print("sum is ",total)  
  
addition()  
addition(20,30,40)  
addition(50,60,80,90)  
  
  
  
 *#program*def even(list1):  
 list2=[]  
 for i in list1:  
 if i % 2 == 0:  
 list2.append(i)  
 return list2  
  
list2 = even([1,5,7,4,6,8,10,12])  
print("Even numbers are" ,list2)  
  
*#program*def add(\*numbers):  
 total=0  
 for n in numbers:  
 total=total+n  
 print("Sum is ",total)  
  
add()  
add(4,5,6,7)  
add(20,30,40,50)  
  
*# #Program*def fact(n):  
 if n ==0:  
 return 1  
 else:  
 return n\*fact(n-1)  
print("factorial of a number ",fact(7))  
  
 *#Program -lambda function*l=[2,5,6,10,12,14,18]  
even\_no= list(filter(lambda x: x % 2 ==0,l))  
print("Even numbers are ",even\_no)  
  
*#program - ATM program*amount1=100000  
while True:  
 print("1 Deposit")  
 print("2 Withdrawal")  
 print("3 Balance")  
 a = input("Enter the option")  
 if a =="1":  
 s=int(input("Enter the Deposit amount"))  
 amount1 = amount1+s  
 print("total balance",amount1)  
 elif a =="2":  
 f = int(input("Enter the Withdrawal amount"))  
 amount1 = amount1 - f  
 print("Total balance ",amount1)  
 elif a =="3":  
 print("Existing balance", amount1)  
  
 b=input("Do you want to continue ?(yes/no)")  
 if b == "no":  
 print("Thank you")  
 break  
 elif b =="yes":  
 continue  
  
  
*#Program - SWap*a=int(input("Enter the first value "))  
b=int(input("Enter the second value "))  
print("Original value of a and b is :",a,"",b)  
a = a+b  
b = a-b  
a = a-b  
print(" After swapping a is :",a,"b is :" ,b)  
  
*#Program - occurrences of each character*str1="Hello world"  
print("occurrences of H :", str1.count("H"))  
print("occurrences of e :", str1.count("e"))  
print("occurrences of l :", str1.count("l"))  
print("occurrences of o :", str1.count("o"))  
print("occurrences of w :", str1.count("w"))  
print("occurrences of r :", str1.count("r"))  
print("occurrences of d :", str1.count("d"))  
  
*#program- Anagram strings*b=("LisTen AcT")  
b1=b.lower()  
a=("SiLeNt CAT")  
a1 = a.lower()  
a1 = [a1[i] for i in range(0,len(a1))]  
a1.sort()  
b1 = [b1[i] for i in range(0,len(b1))]  
b1.sort()  
if a1 == b1:  
 print("Given strings are Anagram")  
else:  
 print("Given strings are not Anagram")  
  
  
*#program -4. Write a python program to reverse each word of a given string*def sentence(s1):  
 words=s1.split(" ")  
 newwords=[word[::-1] for word in words]  
 newsentence=" ".join(newwords)  
 return newsentence  
  
s1="Python Concept Of The Day"  
print(sentence(s1))  
  
*#program-substring*def substr(str1,n):  
 for i in range(n):  
 temp= " "  
 for j in range(i,n):  
 temp += str1[j]  
 print(temp)  
str1="test"  
substr(str1,len(str1))  
print(str1)  
  
*#rpogram -print pattern*for i in range(1,6):  
 for j in range(5,0,-1):  
 if(i ==j):  
 print("\*",end="")  
 else:  
 print(j,end="")  
 print()  
  
*#Program for diamond pattern*rows = 5  
k = 2 \* rows - 2  
for i in range(0, rows):  
 for j in range(0, k):  
 print(end=" ")  
 k = k - 1  
 for j in range(0, i + 1):  
 print("\* ", end="")  
 print("")  
k = rows - 2  
  
for i in range(rows, -1, -1):  
 for j in range(k, 0, -1):  
 print(end=" ")  
 k = k + 1  
 for j in range(0, i + 1):  
 print("\* ", end="")  
 print("")  
*#Program - print pattern E*for row in range(7):  
 for col in range(5):  
 if col ==0 or ((row==0 or row ==3 or row ==6 and col>0)):  
 print("\*", end="")  
 else:  
 print(end=" ")  
 print()  
print("\n")  
print("\n")  
*#program - print pattern O*for row in range(7):  
 for col in range(5):  
 if ((col ==0 or col ==4) and (row!=0 and row!=6))or ((row ==0 or row==6) and (col>0 and col<4)):  
 print("\*",end="")  
 else:  
 print(end=" ")  
 print()  
print("\n")  
print("\n")  
*#program -print pattern F*for row in range(7):  
 for col in range(5):  
 if (col ==0) or ((row ==0 or row ==3) and col>0):  
 print("\*",end="")  
 else:  
 print(end = " ")  
 print()  
print("\n")  
print("\n")  
  
*#program - pattern triangle*rows=int(input("Enter number of rows: "))  
k=0  
count=0  
count1=0  
for i in range(1,rows+1):  
 for j in range(1,(rows-i)+1):  
 print(" ",end="")  
 count+= 1  
 while k!=((2\*i)-1):  
 if count<=rows-1:  
 print(i+k,end=" ")  
 count+=1  
 else:  
 count1+=1  
 print(i+k-(2\*count1),end=" ")  
 k+=1  
  
 count1=count=k= 0  
 print()  
print("\n")  
print("\n")  
  
*#program*a=int((input("Enter the first number")))  
b=(int(input("Enter the second number")))  
def add(a,b):  
 print("addition value - ", a+b)  
def sub(a,b):  
 print("subtraction value - ", a-b)  
def mul(a,b):  
 print("Multiplied value - ",a\*b)  
def div(a,b):  
 print(" Divided value - " , a/b)  
def pow(a,b):  
 print("power of a is - " , a\*\*b)  
  
add(a,b)  
sub(a,b)  
mul(a,b)  
div(a,b)  
*# pow(a,b)  
  
#program- self*class check:  
 def \_\_init\_\_(self):  
 print("Address of self = ", id(self))  
  
  
obj = check()  
print("Address of class object = ", id(obj))  
  
*# #PROGRAM*class student:  
 def check\_pass\_fail(self):  
 if self.marks >=40:  
 return True  
 else:  
 return False  
  
 def \_\_init\_\_(self,name,marks):  
 self.name = name  
 self.marks=marks  
  
student1 = student("harry" , 85)  
student2 = student("janet" , 30)  
did\_pass=student1.check\_pass\_fail()  
print(did\_pass)  
  
did\_pass=student2.check\_pass\_fail()  
print(did\_pass)  
  
*#program*class student:  
 def \_\_init\_\_(self,name,rollno,dob,city):  
 self.name = name  
 self.dob = dob  
 self.rollno = rollno  
 self.city = city  
  
 def address(self):  
 addr = f"Name: {self.name}\nDOB : {self.dob}\nRollno : {self.rollno}\nCity: {self.city} "  
 return addr  
  
stu1= student("Anandh",100,1998,"chennai")  
stu2 = student("Ram",200,1999,"Thanjavur")  
print(stu1.address())  
print(stu2.address())  
  
*#program -Array*import array  
balance=array.array('i', [300,200,100])  
print(balance[1])  
  
*#program*import array as myarray  
abc = myarray.array('d', [2.5,4.5,6.7])  
print("Array first element is :" , abc[0])  
print("Array last element is :", abc[-1])  
  
*#program - Array slicing*import array as myarray  
abc = myarray.array('q',[3,9,6,5,20,13,19,22,30,25])  
print(abc[1:4])  
print(abc[7:10])  
  
*#Program - class*class person:  
 def \_\_init\_\_(self,name,age):  
 self.name=name  
 self.age=age  
  
 def printfun(self):  
 print("My name is " ,self.name, ",age is ", self.age)  
  
p1= person ("John",37)  
p1.printfun()  
  
*#Program-delete*class person:  
 def \_\_init\_\_( self,name,age):  
 self.name=name  
 self.age=age  
 def printfun(self):  
 print("My name is ",self.name,",age is ",self.age)  
  
p2=person("Mary",35)  
p2.printfun()  
*#del p2.age*print(p2.age)  
p2.name ="Ram"  
p2.printfun()  
  
*#program - Create a class named Person, with firstname and lastname properties, and a printname method:*class person:  
 def \_\_init\_\_(self,fname,lname):  
 self.firstname=fname  
 self.lastname=lname  
 def printname(self):  
 print(self.firstname,self.lastname)  
  
p2=person("Amenda","johnson")  
p2.printname()  
  
*#Program- create base and derived class*class person: *#baseclass* def \_\_init\_\_(self,fname,lname):  
 self.firstname=fname  
 self.lastname=lname  
 def printname(self):  
 print(self.firstname,self.lastname)  
  
class student(person): *#Derived class* pass  
  
p3=student("john","Robinson")  
p3.printname()  
  
*#Program*class person: *#baseclass* def \_\_init\_\_(self,fname,lname):  
 self.firstname=fname  
 self.lastname=lname  
 def printname(self):  
 print(self.firstname,self.lastname)  
  
class student(person): *#Derived class* def \_\_init\_\_(self,fname,lname):  
 person.\_\_init\_\_(self,fname,lname)  
  
x=student("Mike", "Olsen")  
x.printname()  
  
*#Program - using super function and add properties.*class person:  
 def \_\_init\_\_(self,fname,lname):  
 self.firstname=fname  
 self.lastname=lname  
 def printname(self):  
 print(self.firstname,self.lastname)  
  
class student(person):  
 def \_\_init\_\_(self,fname,lname):  
 super().\_\_init\_\_(fname,lname)  
 self.graduationyear = 2019  
  
x=student("mike","Olsen")  
print(x.graduationyear)  
  
*#program-using constructor*class person:  
 def \_\_init\_\_(self,name):  
 print("Inside constructor")  
 self.name=name  
 print("All variables are initialized")  
 def show(self):  
 print("My name is ",self.name)  
  
x=person("Emili")  
x.show()  
  
*#program-constructor*class Employee:  
 def show(self):  
 print("Inside show method")  
  
d=Employee()  
d.show()  
  
*#Program-Non-Parametrized Constructor*class Employee:  
 def \_\_init\_\_(self):  
 self.name="guna"  
 self.address="Krishna nagar"  
 def display(self):  
 print("My name is ",self.name,",I am reside at ",self.address)  
  
emp=Employee()  
  
emp.display()  
  
*#Program-Parameterized constructor*class industry:  
 def \_\_init\_\_(self,name,age,address):  
 self.name=name  
 self.age=age  
 self.address=address  
  
 def showmethod(self):  
 print("Name- ",self.name,"Age- ", self.age,"Address- ",self.address)  
  
i = industry("Shyam ",34,"Mangalapuram")  
i.showmethod()  
i1 = industry("Siva", 23," Natarajapuram")  
i1.showmethod()  
  
*#Program*class student:  
 def \_\_init\_\_(self,name,age=12, std=12):  
 self.name=name  
 self.age=age  
 self.std=std  
  
 def disp(self):  
 print(f"Name:{self.name}\nAge: {self.age}\nStandard: {self.std}")  
  
s1=student("Ram")  
s1.disp()  
  
s2=student("Anjana" , 13,8)  
s2.disp()  
  
  
*#program -print class name*class car:  
 def parts(self):  
 pass  
class bus:  
 def route(self):  
 pass  
b=bus()  
print(b.\_\_class\_\_)  
b1=b.\_\_class\_\_  
print(b1.\_\_name\_\_)  
c=car()  
print(c.\_\_class\_\_)  
classes=c.\_\_class\_\_  
print(classes.\_\_name\_\_)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
*#Program - Encapsulation*class student:  
 def \_\_init\_\_(self,name,salary,project):  
 self.name=name  
 self.salary=salary  
 self.project=project  
  
 def show(self):  
 print("Name is ",self.name, "salary = ",self.salary)  
  
 def work(self):  
 print("Name is ", self.name,",I am working at ",self.project)  
  
s1=student("Mary",12000,"NLP")  
s1.show()  
s1.work()  
s2 = student("Anderson" ,14000,"HIG")  
s2.show()  
s2.work()  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#Program - public members*class student:  
 def \_\_init\_\_(self,name,salary):  
 self.name=name  
 self.salary=salary  
  
 def printmethod(self):  
 print("Name is ",self.name,"and my salary is ",self.salary)  
  
g=student("Leo ", 10000)  
g.printmethod()  
  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
*#Program-private members*class detail:  
 def \_\_init\_\_(self,name,age,salary):  
 self.name=name  
 self.\_\_age=age  
 self.salary=salary  
  
d=detail("Anamica ",34,23000)  
print("Name is ",d.name,"salary ",d.salary)  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#Program- access private member*class employee:  
 def \_\_init\_\_(self,name,salary):  
 self.name=name  
 self.\_\_salary=salary  
  
 def show(self):  
 print("Name is ",self.name,"and my salary is ",self.\_\_salary)  
  
emp = employee("lara",30000)  
emp.show()  
print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~")  
  
*#program- Name Mangling*class Employee:  
 def \_\_init\_\_(self, name, salary):  
 self.name = name  
 self.\_\_salary = salary  
  
emp = Employee('Jessa', 10000)  
  
print('Name:', emp.name)  
print('Salary:', emp.\_Employee\_\_salary)  
  
  
*#program*print(4+3%5)  
  
*#program*class vehicle:  
 def \_\_init\_\_(self,price):  
 self.price=price  
 def display(self):  
 print("price = ", self.price)  
  
class category(vehicle):  
 def \_\_init\_\_(self,price,name):  
 vehicle.\_\_init\_\_(self,price)  
 self.name=name  
 def disp\_name(self):  
 print("vehicle=",self.name)  
  
obj= category(20000,'BMW')  
obj.disp\_name()  
obj.display()  
  
*#program*print(issubclass(category,vehicle))  
  
*#Program*print(isinstance(obj,vehicle))  
  
*#program*class A:  
 def display(self):  
 print("This is base class")  
  
class B(A):  
 def display(self):  
 print("This is derived class")  
  
obj = B()  
obj.display()  
  
*#Program -*class vehicle:  
 def vehicle\_info(self):  
 print("Inside vehicle class")  
class car(vehicle):  
 def car\_info(self):  
 print("Inside car class")  
  
c = car()  
c.vehicle\_info()  
c.car\_info()  
  
  
*#program- accessing private members from public method*class employee:  
 def \_\_init\_\_(self,name,salary):  
 self.name=name  
 self.\_\_salary=salary  
  
 def show(self):  
 print("Name- ",self.name,",Salary- ",self.\_\_salary)  
  
emp=employee("Mary",20000)  
emp.show()  
  
*#program- Name Mangling*class employee:  
 def \_\_init\_\_(self,name,salary):  
 self.name=name  
 self.\_\_salary=salary  
  
emp=employee("Ram",30000)  
print("Name- ",emp.name)  
print("Salary- ",emp.\_employee\_\_salary)  
  
*#Program-Protected Member*class company():  
 def \_\_init\_\_(self):  
 self.\_project="NLC"  
class employee(company):  
 def \_\_init\_\_(self,name):  
 self.name=name  
 company.\_\_init\_\_(self)  
  
 def show(self):  
 print("Employee name : ",self.name)  
 print("Working on project : ",self.\_project)  
  
emp=employee("Amenda")  
emp.show()  
  
Program-getters and setters  
class student:  
 def \_\_init\_\_(self,name,age):  
 self.name=name  
 self.\_\_age=age  
 def get\_age(self):  
 return self.\_\_age  
 def set\_age(self,age):  
 self.\_\_age=age  
  
stud = student("Gourav",12)  
print("Name- ",stud.name,",Age- ",stud.get\_age())  
print("After setting age")  
stud.set\_age(15)  
print("Name- ",stud.name,",Age- ",stud.get\_age())  
  
*#Program-Information Hiding and conditional logic for setting an object attributes*class student:  
 def \_\_init\_\_(self,name,rollno,age):  
 self.name=name  
 self.\_\_rollno=rollno  
 self.\_\_age=age  
 def show(self):  
 print("Name- ",self.name,",Roll.no- ",self.\_\_rollno,",Age- ",self.\_\_age)  
 def get\_rollno(self):  
 return self.\_\_rollno  
 def set\_rollno(self,number):  
 if number >50:  
 print("Please enter correct roll number")  
 else:  
 self.\_\_rollno=number  
  
s1=student("Mary",123,15)  
s1.show()  
print("After modify")  
s1.set\_rollno(23)  
s1.show()  
  
*#Program- classes and objects*class vehicle():  
 def \_\_init\_\_(self,maxspeed,mileage):  
 self.maxspeed=maxspeed  
 self.mileage=mileage  
  
v=vehicle(140,120)  
print("maxspeed ",v.maxspeed,"mileage ",v.mileage)  
  
*#Program- Bus object that will inherit all of the variables and methods of the parent Vehicle class and display it.*class Vehicle:  
  
 def \_\_init\_\_(self, name, max\_speed, mileage):  
 self.name = name  
 self.max\_speed = max\_speed  
 self.mileage = mileage  
  
class bus(Vehicle):  
 pass  
  
volvo=bus("School volvo",130,150)  
print("Bus name-",volvo.name,",Speed-",volvo.max\_speed,",Mileage-",volvo.mileage)  
  
  
*#Program*x=int(input("Enter the value to be checked as odd or even - "))  
def check():  
 if x & 1:  
 return 'odd'  
 else:  
 return 'even'  
  
print("Result is ",check())  
  
*#Program*class Mystore:  
 \_\_prod\_code=[]  
 \_\_prod\_name=[]  
 \_\_prod\_price=[]  
 \_\_prod\_quant=[]  
 def getdata(self):  
 self.p=int(input("Enter no.of products you need to store: "))  
 e=[]  
 for x in range(self.p):  
 e = self.\_\_prod\_code.append(input("Enter product code: "))  
 self.\_\_prod\_name.append(input("Enter product name: "))  
 self.\_\_prod\_price.append(int(input("Enter cost price: ")))  
  
 def display(self):  
 print(" Stock in stores")  
 print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
 print("Product code\t Product name\tCost price")  
 print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
 for x in range(self.p):  
 if self.e == self.\_\_prod\_code[x]:  
 print(" Caution!!! Code number is repeated")  
 break  
 for x in range(self.p):  
 print(self.\_\_prod\_code[x], "\t\t\t",self.\_\_prod\_name[x],"\t\t\t",self.\_\_prod\_price[x])  
 print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
 def print\_bill(self):  
 total\_price=0  
 for x in range(self.p):  
 print("Enter the Quantity of product code")  
 q=int(input(self.\_\_prod\_code[x] ))  
 self.\_\_prod\_quant.append(q)  
 total\_price=total\_price+self.\_\_prod\_price[x]\*self.\_\_prod\_quant[x]  
 print(" INVOICE RECEIPT ")  
 print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
 print("Product code\t\tProduct name\t\tCost price\t\tQuantity\t\tTotal Amount")  
 for x in range(self.p):  
 print(self.\_\_prod\_code[x],"\t\t\t\t",self.\_\_prod\_name[x],"\t\t\t\t",self.\_\_prod\_price[x],"\t\t\t\t", self.\_\_prod\_quant[x],"\t\t\t\t",self.\_\_prod\_quant[x]\*self.\_\_prod\_price[x])  
 print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
 print(" Total Amount=", total\_price)  
 Amount=int(input("Enter the amount given by consumer "))  
 Total= Amount -total\_price  
 print("Balance amount= ",Total)  
  
s=Mystore()  
s.getdata()  
s.display()  
s.print\_bill()  
  
  
*#Program -Create a Bus class that inherits from the Vehicle class. Give the capacity argument of Bus.seating\_capacity() a default value of 50.*class Vehicle:  
 def \_\_init\_\_(self, name, max\_speed, mileage):  
 self.name = name  
 self.max\_speed = max\_speed  
 self.mileage = mileage  
  
 def seating\_capacity(self, capacity):  
 return f" The Seating Capacity of a {self.name} is {capacity}"  
  
class bus(Vehicle):  
 def seating\_capacity(self,capacity=50):  
 return super().seating\_capacity(capacity=50)  
  
b = bus("Omni",150,250)  
print(b.seating\_capacity())  
  
*#program-Define a class attribute”color” with a default value white. I.e., Every Vehicle should be white.*class Vehicle:  
  
 def \_\_init\_\_(self, name, max\_speed, mileage):  
 self.name = name  
 self.max\_speed = max\_speed  
 self.mileage = mileage  
  
class Bus(Vehicle):  
 color = "white"  
  
class Car(Vehicle):  
 color = "white"  
  
c= Car("SWift ",170,280)  
print("Name- ",c.name,",Max-speed- ",c.max\_speed,",Mileage- ",c.mileage)  
c1= Bus("Volvo" , 180,230)  
print("Name- ",c1.name,",Max-speed- ",c1.max\_speed,",Mileage- ",c1.mileage)  
  
*#Program - Create a Bus child class that inherits from the Vehicle class. The default fare charge of any vehicle is seating capacity \* 100.  
# If Vehicle is Bus instance, we need to add an extra 10% on full fare as a maintenance charge.  
# So total fare for bus instance will become the final amount = total fare + 10% of the total fare.*class vehicle():  
 def \_\_init\_\_(self,name,maxspeed,mileage,capacity):  
 self.name=name  
 self.maxspeed=maxspeed  
 self.mileage=mileage  
 self.capacity=capacity  
 def fare(self):  
 print("Name- ",self.name,"Maxspeed- ",self.maxspeed,"Mileage- ",self.mileage,"Seating capacity- ",self.capacity)  
 return self.capacity\*100  
class bus(vehicle):  
 pass  
schoolbus= bus("Omni",120,260,50)  
print("Amount= ",schoolbus.fare())  
  
*#program- Write a program to determine which class a given Bus object belongs to.*class vehicle():  
 def \_\_init\_\_(self,name,maxspeed,mileage,capacity):  
 self.name=name  
 self.maxspeed=maxspeed  
 self.mileage=mileage  
 self.capacity=capacity  
class bus(vehicle):  
 pass  
  
schoolbus = bus("Maruthi",150,250,30)  
print("Name- ",schoolbus.name,",Maxspeed- ",schoolbus.maxspeed,",Mileage- ",schoolbus.mileage,",Seating capacity- ",schoolbus.capacity)  
print(type(schoolbus))