40-program-python

January 18, 2024

1 *1.calculator progarm*

[23]:
```
def calc(n): #use def(define)function name calc(n)
    s=0 #sum(s)
    p=1 #product(p)
#take tow number
    a=int(input("first number:"))
    b=int(input("second number:"))
    if n==1: #add
        s=a+b
        print("sum",s)
    elif n==2: #multiblication
        p=a*b
        print("prod",p)
    elif n==3:  #subtract
        s=a-b
        print("diff",s)
    else: #devision
        p=a/b
        print("divide",p)
i=int(input("enter for calculation 1. add 2. product 3.subtract 4.
 ↪divide\ncolculation:"))
calc(i)
```

```
enter for calculation 1. add 2. product 3.subtract 4.divide
colculation:3
first number:78
second number:88
diff -10
```

2 *2.Insertion Sort Program*

[24]:
```
def insort(a): #Define(def) function named insort(a)
    for i in range(1,len(a)): #list a
        b = a[i] # b variable store each element
        j = i - 1
        while j >= 0 and b < a[j]:
```

```
            a[j], a[j + 1] = a[j + 1], a[j] #swap two element
            j -= 1
a = [88,39,66,44,20,12,43]
insort(a)
for i in a:
    print(i)
```

```
12
20
39
43
44
66
88
```

3 *3.Linear Search program*

[25]:
```
n=int(input("Enter the number to be searched (1-10):"))
a=[1, 2, 4, 3,5,7,9,8,6,10 ] #array a is instalized
for i in range(1,(len(a))): # linear search
    if n==a[i]: # found number
        print("number found at",i+1)
```

```
Enter the number to be searched (1-10):7
number found at 6
```

4 *4.GCD of two numbers*

[26]:
```
def hcf(a, b):    # a,b two numbers
    if(b == 0):
        return a
    else:
        return hcf(b, a % b)

a=int(input("Enter the first number: "))
b=int(input("Enter second number: "))

# prints 12
print("The gcd of",a,"and",b,"is : ",end="")
print(hcf(a,b))
```

```
Enter the first number: 12
Enter second number: 14
The gcd of 12 and 14 is : 2
```

5 5.Find square root

[27]:
```
def sqrt(n):
    x = n ** 0.5 #  used exponential operator (**)
    print(x)
n = int(input("Enter the number whose square root you need to find: "))
sqrt(n)
```

```
Enter the number whose square root you need to find: 6
2.449489742783178
```

6 *6.decimal to binary*

[28]:
```
def dectobin(n):
    if n>1:
        dectobin(n//2)
    print(n%2,end='')
n=int(input("Enter the decimal number: "))
dectobin(n)
```

```
Enter the decimal number: 67
1000011
```

6.1 *7.find Compound interest*

[31]:
```
#formula for A = P * (1 + r/n)^(n*t)
n = int(input("Enter the principle amount:"))# n = principal
rate = int(input("Enter the rate:"))
years = int(input("Enter the number of years:"))
principal=n
time=years
n + ((n * rate) / 100)
Amount = principal * (pow((1 + rate / 100), time))
CI = Amount - principal
print(CI)
```

```
Enter the principle amount:4000
Enter the rate:2
Enter the number of years:13
1174.4265218151859
```

7 *8.diamond pattern program*

[32]:
```
h = eval(input("Enter diamond's height: "))
for x in range(h):
    print(" " * (h - x), "*" * (2*x + 1))
for x in range(h - 2, -1, -1):
    print(" " * (h - x), "*" * (2*x + 1))
```

```
Enter diamond's height: 6
       *
      ***
     *****
    *******
   *********
  ***********
   *********
    *******
     *****
      ***
       *
```

8 *9.hellow dimond*

[33]:
```
# Reading number of row
row = int(input('Enter number of row: '))
# Upper part of hollow diamond
for i in range(1, row+1):
    for j in range(1,row-i+1):
        print(" ", end="")
    for j in range(1, 2*i):
        if j==1 or j==2*i-1:
            print("*", end="")
        else:
            print(" ", end="")
    print()
# Lower part of hollow diamond
for i in range(row-1,0,-1):
    for j in range(1,row-i+1):
        print(" ", end="")
    for j in range(1, 2*i):
        if j==1 or j==2*i-1:
            print("*", end="")
        else:
            print(" ", end="")
    print()
```

```
Enter number of row: 4
```

```
   *
  * *
 *   *
*     *
 *   *
  * *
   *
```

9 *10.biggest number*

[35]:
```
l=[]
k=int(input("num:"))
for i in range(k):
    n=int(input("Enter the no.:"))
    l.append(n)
big=0
for i in l:
    if i>big:
        big=i;
print("The Biggest no. of all",k,"is:",big)
```

```
num:4
Enter the no.:87
Enter the no.:56
Enter the no.:446
Enter the no.:67
The Biggest no. of all 4 is: 446
```

10 *11.Reverse string program*

[36]:
```
string=input("Enter a string:")
revstring=string[::-1]  #//performs the reverse of the string
print(revstring)    # // print the reversed string
```

```
Enter a string:tamil
limat
```

[37]:
```
def reverse(s):
    str = ""
    for i in s:
        str = i + str
    return str
s =(input("Name :"))
print("The original string is : ", end="")
print(s)
```

```
print("The reversed string(using loops) is : ", end="")
print(reverse(s))
```

```
Name :tamill
The original string is : tamill
The reversed string(using loops) is : llimat
```

11 12.Anagram python program *

[38]:
```
a=input("Enter string 1:")
b=input("Enter string 2:")
count=0
for i in a:
    for j in b:
        if i==j:
            count=count+1
if count==len(a):
    print("Strings are anagram of each other.")
else:
    print("Strings are not anagram of each other.")
```

```
Enter string 1:tamil
Enter string 2:karthi
Strings are not anagram of each other.
```

12 *13.average of N numbers*

[39]:
```
n = int(input("Enter the total number you want to enter:"))
sum = 0
for i in range(n):
    x = int(input("Enter the number:"))
    sum = sum + x
avg = sum / n
print("Average=", avg)
```

```
Enter the total number you want to enter:2
Enter the number:345
Enter the number:456
Average= 400.5
```

13 *14.Armstrong Number*

[40]:
```
num=int(input("Enter number:")) # user input
sum=0
#n=len(str(num))
```

```
temp=num
while temp > 0 :
  digit = temp % 10
  sum+=digit**3 #n is any digt number aspcet
  temp //=  10
if num==sum :
  print(num,"it is amstrong number: ")
else:
  print(num,"it is not amstrong number:")
```

```
Enter number:78
78 it is not amstrong number:
```

14 *15.Factorial Program*

[41]:
```
n= int(input("Enter the number of factorial : "))
prod=1
for i in range(1,n+1):
    prod=prod*i
print(prod)
```

```
Enter the number of factorial : 78
113242811782062978314575211587320462287317495794882519900489628256688353252342007662450862131773440000000000000000000
```

15 *16.Leap year*

[42]:
```
n = int(input("Enter the year : "))
if n % 4 == 0:
  if n % 100 == 0:
    if n % 400 == 0:
      print("The year is a leap year")
    else:
      print("The year is not a leap year")
  else:
    print("The year is a leap year")
else:
  print("The year is not a leap year")
```

```
Enter the year : 2004
The year is a leap year
```

16 *17.Prime Number*

[43]:
```
num = int(input("Enter a number: "))
if num == 1:
    print(num, "is not a prime number")
elif num > 1:
   # check for factors
   for i in range(2,num):
       if (num % i) == 0:
           print(num,"is not a prime number")
           print(i,"times",num//i,"is",num)
           break
   else:
       print(num,"is a prime number")
# if input number is less than
# or equal to 1, it is not prime
else:
   print(num,"is not a prime number")
```

```
Enter a number: 2004
2004 is not a prime number
2 times 1002 is 2004
```

17 *18.Fibonacci series*

[44]:
```
n= int(input("enter the number :"))
c=[]
c.append(0)
c.append(1)
a=0
b=1
d=0
for i in range(1, n-1):
  d=a+b
  c.append(d)
  a=b
  b=d
for i in c:
  print(i)
```

```
enter the number :67
0
1
1
2
3
5
```

8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
6765
10946
17711
28657
46368
75025
121393
196418
317811
514229
832040
1346269
2178309
3524578
5702887
9227465
14930352
24157817
39088169
63245986
102334155
165580141
267914296
433494437
701408733
1134903170
1836311903
2971215073
4807526976
7778742049
12586269025
20365011074
32951280099
53316291173

```
86267571272
139583862445
225851433717
365435296162
591286729879
956722026041
1548008755920
2504730781961
4052739537881
6557470319842
10610209857723
17167680177565
27777890035288
```

18 *19.palindrome number*

[45]:
```
n= int(input("Enter the number:"))
temp=n
rev=0
while(n>0):
  digit=n%10
  rev=rev*10+digit
  n=n//10
if (temp==rev) :
  print("It is a palindrome")
else:
   print("It is not a palindrome")
```

```
Enter the number:89
It is not a palindrome
```

19 20.even or odd

[46]:
```
#program to determine a number is even or not
a= int(input("Enter number "))
if(a%2==0):
  print(a,"is even number")
else:
  print(a,"is odd number")
```

```
Enter number 67
67 is odd number
```

20 *21.reverse of number*

[47]:
```
num = int(input('Enter an integer number: '))
# reverse of number
reverse = 0
while(num > 0):
    last_digit = num % 10
    reverse = reverse * 10 + last_digit
    num = num // 10
# display result
print('The reverse number is = ', reverse)
```

```
Enter an integer number: 765
The reverse number is =  567
```

21 *22.Generate 4digt otp number*

[48]:
```
import math, random # import library
def generateOTP() :  # generate OTP
    digits = "0123456789"
    OTP = ""
    for i in range(4) :
        OTP += digits[math.floor(random.random() * 10)]

    return OTP

print("OTP of 4 digits:", generateOTP())
```

```
OTP of 4 digits: 7578
```

22 *23.possitive or negative*

[49]:
```
num = float(input("Enter a number: "))
if num > 0:
   print("Positive number")
elif num == 0:
   print("Zero")
else:
   print("Negative number")
```

```
Enter a number: 6
Positive number
```

23 *24.prime numbers within an interval*

[50]:
```
low =int(input("lower value:"))
up = int(input("upper value:"))

print("Prime numbers between", low, "and", up, "are:")

for num in range(low, up + 1):
   if num > 1: # all prime numbers are greater than 1
       for i in range(2, num):
           if (num % i) == 0:
               break
       else:
           print(num)
```

```
lower value:67
upper value:89
Prime numbers between 67 and 89 are:
67
71
73
79
83
89
```

24 *25.amstrong certain interval*

[51]:
```
lower =100
upper =int(input("enter upper value:"))

for num in range(lower, upper + 1):

   order = len(str(num))
   sum = 0

   temp = num
   while temp > 0:
       digit = temp % 10
       sum += digit ** order
       temp //= 10

   if num == sum:
       print(num)
```

```
enter upper value:78
```

25 *26. Sum of natural numbers*

[52]:
```
#n*(n+1)/2
num =int(input("number:"))

if num < 0:
   print("Enter a positive number")
else:
   sum = 0
   while(num > 0):
       sum += num
       num -= 1
   print("The sum is", sum)
```

```
number:89
The sum is 4005
```

26 *27.ASCII value*

[57]:
```
c =(input(" ASCII value :"))
print("The ASCII value of '" + c + "' is", ord(c))
```

```
 ASCII value :k
The ASCII value of 'k' is 107
```

27 *28.LCM*

[58]:
```
def compute_lcm(x, y):
   if x > y:
       greater = x
   else:
       greater = y

   while(True):
       if((greater % x == 0) and (greater % y == 0)):
           lcm = greater
           break
       greater += 1
   return lcm

num1 =int(input("enter num1:"))
num2 = int(input("enter num2:"))

print("The L.C.M. is", compute_lcm(num1, num2))
```

```
enter num1:34
enter num2:56
```

```
The L.C.M. is 952
```

28 *29.shuffle a deck of card*

[59]:
```
import itertools, random  # import module
deck = list(itertools.product(range(1,14),['Spade','Heart','Diamond','Club']))
random.shuffle(deck) # shuffle the cards
print("You got:")
for i in range(6):
   print(deck[i][0], "of", deck[i][1])
```

```
You got:
4 of Club
2 of Heart
1 of Club
13 of Diamond
11 of Club
7 of Diamond
```

29 *30.display calendar*

[60]:
```
import calendar
# To take month and year input from the user
yy = int(input("Enter year: "))
mm = int(input("Enter month: "))
print(calendar.month(yy, mm))
```

```
Enter year: 2024
Enter month: 6
     June 2024
Mo Tu We Th Fr Sa Su
                1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
```

30 *31.date and time in string format*

[61]:
```
from datetime import datetime

my_date_string ="jun 11 2011 11:20PM"

datetime_object = datetime.strptime(my_date_string, '%b %d %Y %I:%M%p')
```

```
print(type(datetime_object))
print(datetime_object)
```

```
<class 'datetime.datetime'>
2011-06-11 23:20:00
```

31 *32.Number swaping*

[62]:
```
a=int(input("Enter the a value:"))
b=int(input("Enter the b value:"))
print("Enter the before swapping:",a,b)
temp = a
a = b
b = temp
print("enter the after swapping:",a,b)
```

```
Enter the a value:55
Enter the b value:77
Enter the before swapping: 55 77
enter the after swapping: 77 55
```

32 *33.Mark sheet*

[63]:
```
name=(input("student Name:"))
sub1=int(input("Enter marks of the first subject: "))
sub2=int(input("Enter marks of the second subject: "))
sub3=int(input("Enter marks of the third subject: "))
sub4=int(input("Enter marks of the fourth subject: "))
sub5=int(input("Enter marks of the fifth subject: "))
avg=(sub1+sub2+sub3+sub4+sub4)/5
if(avg>=90):
    print("Grade: A")
elif(avg >=80 and avg<90):
    print("Grade: B")
elif(avg>=70 and avg<80):
    print("Grade: C")
elif(avg>=60 and avg<70):
    print("Grade: D")
else:
    print("Grade: F")
```

```
student Name:kkk
Enter marks of the first subject: 90
Enter marks of the second subject: 0
Enter marks of the third subject: 0
```

```
Enter marks of the fourth subject: 0
Enter marks of the fifth subject: 0
Grade: F
```

33 *34.creating a dictionary*

[64]:
```
country_capitals = {
  "Germany": "Berlin",
  "Canada": "Ottawa",
  "England": "London"
}

# printing the dictionary
print(country_capitals)
```

```
{'Germany': 'Berlin', 'Canada': 'Ottawa', 'England': 'London'}
```

34 *35.nested dictionary*

[65]:
```
people = {1: {'name': 'sibasri', 'age': '27', 'sex': 'Male'},
          2: {'name': 'naveen', 'age': '22', 'sex': 'Female'},
          3: {'name': 'tamil', 'age': '22', 'sex': 'Female'}}

print(people)
```

```
{1: {'name': 'sibasri', 'age': '27', 'sex': 'Male'}, 2: {'name': 'naveen',
'age': '22', 'sex': 'Female'}, 3: {'name': 'tamil', 'age': '22', 'sex':
'Female'}}
```

35 *36.inheritance*

[66]:
```
class Person(object):
# Constructor
  def __init__(self, name, id):
    self.name = name
    self.id = id
  def Display(self):
    print(self.name, self.id)
emp = Person("tamil", 1423) # An Object
emp.Display()
```

```
tamil 1423
```

36 *37.Encapsulation*

[67]:
```
class Base:
        def __init__(self):
                self._a = 6
class Derived(Base):
        def __init__(self):
                Base.__init__(self)
                print("Calling protected member of base class: ",self._a)
                self._a = 8
                print("Calling modified protected member outside class: ",self.
 ↪_a)

obj1 = Derived()
obj2 = Base()
print("Accessing protected member of obj1: ", obj1._a)
print("Accessing protected member of obj2: ", obj2._a)
```

```
Calling protected member of base class:  6
Calling modified protected member outside class:  8
Accessing protected member of obj1:  8
Accessing protected member of obj2:  6
```

37 *38.Polymorphism*

[68]:
```
def add(x, y, z = 0):
        return x + y+z
print(add(2, 3))
print(add(2, 3, 4))
print(add(2, 3, 9))
```

```
5
9
14
```

38 *39.reverse arr[]*

[70]:
```
# Function to reverse arr[]
def rverseArray(arr,d):
        c=(arr[d:])+(arr[:d])
        return c
arr = [1, 2, 3, 4, 5, 6, 7]
d=5
print(rverseArray(arr,d))
```

```
[6, 7, 1, 2, 3, 4, 5]
```

39 *40.own font using Python*

[71]:
```
name = input("Name: ")
length = len(name)
l = ""

for x in range(0, length):
        c = name[x]
        c = c.upper()

        if (c == "A"):
                print("..######..\n..#....#..\n..######..", end = " ")
                print("\n..#....#..\n..#....#..\n\n")

        elif (c == "B"):
                print("..######..\n..#....#..\n..#####...", end = " ")
                print("\n..#....#..\n..######..\n\n")

        elif (c == "C"):
                print("..######..\n..#.......\n..#.......", end = " ")
                print("\n..#.......\n..######..\n\n")

        elif (c == "D"):
                print("..#####...\n..#....#..\n..#....#..", end = " ")
                print("\n..#....#..\n..#####...\n\n")

        elif (c == "E"):
                print("..######..\n..#.......\n..#####...", end = " ")
                print("\n..#.......\n..######..\n\n")

        elif (c == "F"):
                print("..######..\n..#.......\n..#####...", end = " ")
                print("\n..#.......\n..#.......\n\n")

        elif (c == "G"):
                print("..######..\n..#.......\n..#.####..", end = " ")
                print("\n..#....#..\n..#####...\n\n")

        elif (c == "H"):
                print("..#....#..\n..#....#..\n..######..", end = " ")
                print("\n..#....#..\n..#....#..\n\n")

        elif (c == "I"):
                print("..######..\n....##....\n....##....", end = " ")
                print("\n....##....\n..######..\n\n")

        elif (c == "J"):
```

```
            print("..######..\n....##....\n....##....", end = " ")
            print("\n..#.##....\n..####....\n\n")

        elif (c == "K"):
            print("..#...#...\n..#..#....\n..##......", end = " ")
            print("\n..#..#....\n..#...#...\n\n")

        elif (c == "L"):
            print("..#.......\n..#.......\n..#.......", end = " ")
            print("\n..#.......\n..######..\n\n")

        elif (c == "M"):
            print("..#....#..\n..##..##..\n..#.##.#..", end = " ")
            print("\n..#....#..\n..#....#..\n\n")

        elif (c == "N"):
            print("..#....#..\n..##...#..\n..#.#..#..", end = " ")
            print("\n..#..#.#..\n..#...##..\n\n")

        elif (c == "O"):
            print("..######..\n..#....#..\n..#....#..", end = " ")
            print("\n..#....#..\n..######..\n\n")

        elif (c == "P"):
            print("..######..\n..#....#..\n..######..", end = " ")
            print("\n..#.......\n..#.......\n\n")

        elif (c == "Q"):
            print("..######..\n..#....#..\n..#.#..#..", end = " ")
            print("\n..#..#.#..\n..######..\n\n")

        elif (c == "R"):
            print("..######..\n..#....#..\n..#.##...", end = " ")
            print("\n..#...#...\n..#....#..\n\n")

        elif (c == "S"):
            print("..######..\n..#.......\n..######..", end = " ")
            print("\n.......#..\n..######..\n\n")

        elif (c == "T"):
            print("..######..\n....##....\n....##....", end = " ")
            print("\n....##....\n....##....\n\n")

        elif (c == "U"):
            print("..#....#..\n..#....#..\n..#....#..", end = " ")
            print("\n..#....#..\n..######..\n\n")
```

```
        elif (c == "V"):
                print("..#....#..\n..#....#..\n..#....#..", end = " ")
                print("\n...#..#...\n....##....\n\n")

        elif (c == "W"):
                print("..#....#..\n..#....#..\n..#.##.#..", end = " ")
                print("\n..##..##..\n..#....#..\n\n")

        elif (c == "X"):
                print("..#....#..\n...#..#...\n....##....", end = " ")
                print("\n...#..#...\n..#....#..\n\n")

        elif (c == "Y"):
                print("..#....#..\n...#..#...\n....##....", end = " ")
                print("\n....##....\n....##....\n\n")

        elif (c == "Z"):
                print("..######..\n......#...\n.....#....", end = " ")
                print("\n....#.....\n..######..\n\n")

        elif (c == " "):
                print("..........\n..........\n..........", end = " ")
                print("\n..........\n\n")

        elif (c == "."):
                print("----..----\n\n")
```

```
Name: tamil
..######..
…##…
…##…
…##…
…##…


..######..
..#…#..
..######..
..#…#..
..#…#..


..#…#..
..##..##..
..#.##.#..
..#…#..
..#…#..
```

```
..######..
…##…
…##…
…##…
..######..


..#…
..#…
..#…
..#…
..######..
```

[ ]: