Darshan
UNIVERSITY
योग: कर्मसु कौशलम्

Python Programming - 2101CS405

Lab - 7

Name :Vyas Bhagyesh Y.

Enrollment No : 23010101662

Roll N0 : 23010101662

Functions

01) WAP to count simple interest using function.

In [1]:

```
p=float(input("Enter Principle : "));
r=float(input("Enter Rate of Interest : "));
n=float(input("Enter Time : "));

def simpleI(p,r,n):
    return p*r*n/100
print("Simple Interest :: ",simpleI(p,r,n))
```

```
Simple Interest ::  102.0
```

02) WAP that defines a function to add first n numbers.

In [2]:

```
def add(n):
    sum=0
    for i in range(1,n+1):
        sum+=i
    return sum
n=int(input("Enter n :"))
print("Sum = ",add(n))
```

```
Sum =  5050
```

03) WAP to find maximum number from given two numbers using function.

```
In [3]: def maxn(a,b):
            return a if a>b else b
        a=int(input("Enter number-1 : "))
        b=int(input("Enter number-2 : "))
        print("Max = ",maxn(a,b))
```

```
Max =  57
```

04) WAP that defines a function which returns 1 if the number is prime otherwise return 0.

```
In [5]: def isprime(n):
            for i in range(2,int(n**0.5)):
                if n%i==0:
                    return 0
            return 1
        n=int(input("Enter number : "))
        isprime(n)
```

```
Out[5]: 1
```

05) Write a function called primes that takes an integer value as an argument and returns a list of all prime numbers up to that number.

```
In [ ]: def primeList(n):
            l2=[]
            c=0
            for i in range(2,n+1):
                flag=True
                for j in range(2,i):
                    # c+=1
                    if i%j==0:
                        flag=False
                if flag==True:
                    l2.append(i)
            return l2
        l=primeList(1000)
        print(l)
```

06) WAP to generate Fibonacci series of N given number using function name fibbo. (e.g. 0 1 1 2 3 5 8...)

```
In [6]: def fibbo(n):
            li=[0,1]
            n1=0
            n2=1
            n3=0
            for i in range(2,n):
                n3=n1+n2
                li.append(n3)
                n1=n2
                n2=n3
            return li

        n=int(input("Enter Size: "))
        print(fibbo(n))
```

```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 676
5]
```

07) WAP to find the factorial of a given number using recursion.

In [7]:

```
n=int(input("Enter number : "))

def fact(n):
    if n==1:
        return 1
    return n*fact(n-1)

print("Factorial of ",n,"= ",fact(n))
```

```
Factorial of  10 =  3628800
```

08) WAP to implement simple calculator using lamda function.

In [8]:

```
a=int(input("Enter number-1 : "))
b=int(input("Enter number-2 : "))
op=input("Enter Operator : ")

if op=='+':
    print((lambda a,b:a+b)(a,b))
elif op=='-':
    print((lambda a,b:a-b)(a,b))
elif op=='*':
    print((lambda a,b:a*b)(a,b))
elif op=='/':
    print((lambda a,b:a/b)(a,b))
else:
    print((lambda a,b:a//b)(a,b))
```

```
6
```

09)Write a Python program that accepts a hyphen-separated sequence of words as input and prints the words in a hyphen-separated sequence after sorting them alphabetically

Sample Items : green-red-yellow-black-white

Expected Result : black-green-red-white-yellow

In [9]:

```
str=input("Enter hyphen-separated sequence of words : ")
print(str)
l1=str.split('-')
l1=sorted(l1)
print('-'.join(l1))
```

```
green-red-yellow-black-white
black-green-red-white-yellow
```

10) Write a python program to implement all function arguments type

Positional arguments

Default argument

Keyword arguments (named arguments)

Arbitrary arguments (variable-length arguments args and kwargs)

In [18]:

```
# Positional argument
```

```
def positionalArgument (a,b) :
    print(a+b)

defaultArgument(10,30)
```

```
40
```

In [ ]:

```
# Default argument
def defaultArgument (a,b=10) :
    print(a+b)

defaultArgument(10)
```

In [15]:

```
# Keyword Argument
def keywordArgument (a,b) :
    print(a+b)

defaultArgument(a=10,b=20)
```

```
30
```

In [16]:

```
# Arbitary argument
def add_number(n1,*n):
    sum=n1
    for i in n:
        sum=sum+i
    print("Sum = ",sum)

add_number(10,10,10,10)
```

```
Sum =  40
```

In [17]:

```
# Arbitary keyword argument
def add_number(**a):
    sum=0
    sum=a['a']+a['b']
    print("Sum = ",sum)
    for i in a.items():
        print(i)
    print(a)
add_number(a=4,b=3,c=4)
```

```
Sum =  7
('a', 4)
('b', 3)
('c', 4)
{'a': 4, 'b': 3, 'c': 4}
```

01) WAP to calculate power of a number using recursion.

In [10]:

```
def power(n,p):
    if p>0:
        return n*power(n,p-1)
    else:
        return 1

n=int(input("Enter Number: "))
p=int(input("Enter power: "))
print("Answer = ",power(n,p))
```

```
Answer =  8
```

02) WAP to count digits of a number using recursion.

In [11]:

```
def digit(n):
    if n//10==0:
        return 1
    else:
        return 1+digit(n//10)
n1=int(input("Enter Number: "))
print("Digits = ",digit(n1))
```

```
Digits =  7
```

03) WAP to reverse an integer number using recursion.

In [12]:

```
n=int(input("Enter Number: "))

def r_n(n,rev_n=0):
    if n==0:
        return rev_n
    else:
        lsdigit=n%10
        rev_n=rev_n*10+lsdigit
        return r_n(n//10,rev_n)
r_n(n)
```

Out[12]:

```
4321
```

04) WAP to convert decimal number into binary using recursion.

In [13]:

```
def d_b(dec):
    if dec == 0:
        return 0
    else:
        return(dec % 2 + 10 * d_b(int(dec // 2)))

dec=int(input("Enter Decimal Number: "))
d_b(dec)
```

Out[13]:

```
10110
```