Darshan UNIVERSITY

(https://www.darshan.ac.in/)

Python Programming - 2101CS405

Lab - 7

Functions

01) WAP to count simple interest using function.

In [1]:
```
def simpalInterest(p,r,n):
    return (p*r*n)/100;



p = int(input("Enter P:"))
r = int(input("Enter R:"))
n = int(input("Enter N:"))
interest = simpalInterest(p,r,n)
print(f"simple interest:{interest}")
```

```
Enter P:5
Enter R:2
Enter N:2
simple interest:0.2
```

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

In [2]:
```

def add_n_no(n):
    sum = 0;
    for i in range(1,n+1):
        sum += i;
    return sum

n = int(input("Enter N:"))
ans = add_n_no(n)
print(f"sum of first {n} Number::{ans}")
```

```
Enter N:2
sum of first 2 Number::3
```

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

In [3]:
```
def max_of_two(a,b):
    return a if a>b else b
a = int(input("Enter A:"))
b = int(input("Enter B:"))
maximum = max_of_two(a,b)
print(f"Maximum of {a} and {b} :: {maximum}")
```

```
Enter A:2
Enter B:2
Maximum of 2 and 2 :: 2
```

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

In [4]:
```
def isPrime(n):
    for i in range(2,n//2):
        if(n%i==0):
            return 0;
            break;
    else:
        return 1;
n = int(input("Enter N:"))
ans = isPrime(n)
ans
```

```
Enter N:2
```

Out[4]:
```
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 [5]:
```
def print_all_prime_upto_no(n):
    l1 = [];
    for i in range(2,n+1):
        for j in range(2,i):
            if(i%j==0):
                break;
        else:
            l1.append(i)
    return l1;

n = int(input("Enter N:"))
l2 = print_all_prime_upto_no(n)
l2
```

```
Enter N:2
```

Out[5]:
```
[2]
```

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):
    a = 0;
    b = 1;
    sum = 0;
    for i in range(1,n+1):
        print(sum)
        a = b;
        b = sum;
        sum = a+b
n = int(input("Enter N:"))
fibbo(n)
```

```
Enter N:2
0
1
```

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

In [7]:
```
def fact_using_recursion(n):
    if n==1 or n == 0:
        return 1;
    else:
        return n*fact_using_recursion(n-1)
n = int(input("Enter N:"))
print(fact_using_recursion(n))
```

```
Enter N:2
2
```

08) WAP to implement simple calculator using lamda function.

In [8]:
```
a = int(input("Enter A:"))
b = int(input("Enter B:"))
op = input("Enter Opration(+,-,*,/):")

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

```
Enter A:2
Enter B:2
Enter Opration(+,-,*,/):2
```

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 Stirng using '-(hyphen separated)' value")
l1 = str.split("-")
l1.sort()
print("-".join(l1))
```

```
Enter Stirng using '-(hyphen separated)' value2
2
```

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 [10]:
```
# postional_args
def positional_args(a):
    if a>0:
        print("+ve")
    else:
        print("-ve")
n = int(input("Enter N:"))
positional_args(n)

print("=========================================")
# default args

def default_args(a,b=0):
    print(a+b)
default_args(1,4)
default_args(3)



print("===========================================")


# keywords args

def keyword_args(a,b):
    print(a+b)
keyword_args(a=4,b=6)
keyword_args(b=3,a=2)

print("=================================================")
```

```
Enter N:2
+ve
=========================================
5
3
===========================================
10
5
=================================================
```

In [2]:
```
def arbitary_args(*l1):
    sum = 0
    for i in l1:
        sum += i
    return sum;
print(arbitary_args(1,2,3,4))
```

```
10
```

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

In [3]:
```
def power_using_recursion(b,p):
    if p == 1:
        return b;
    else:
#         p-=1
        return b*power_using_recursion(b,p-1)
base = int(input("Enter base:"))
power =int(input("Enter power:"))
ans = power_using_recursion(base,power)
print(f"Answer:{ans}")
```

```
Enter base:2
Enter power:42
Answer:4398046511104
```

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

In [25]:
```
def counting_number(n):
    if n//10 == 0:
        return 1;
    else:
        return 1 + counting_number(n//10);
print(f"Length of Number is::{counting_number(int(input('Enter Number:')))}")
```

```
Enter Number:123456
Length of Number is::6
```

03) WAP to reverse an integer number using recursion.

In [29]:
```
def rev_number(n):
    if n == 0:
        return 0;
    else:
        return ((n%10)*pow(10,counting_number(num))) + rev_number(n/10)
num = int(input('Enter Number:'))
print(f"Reversed Number is::{rev_number(num)}")
```

```
Enter Number:123
Reversed Number is::6666.666666666668
```

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

In [ ]: