|  |  |
| --- | --- |
| **Ex. No. 5**  **Date:26.05.2021** | **FUNCTIONS – LEVEL 1** |

**AIM:**

To write Python programs using functions concept.

**PROGRAMMING BASE:**

**I. Defining a Function**

Functions are self contained programs that perform some particular task.

**Syntax:**

deffunctionname( parameters ):

"function\_docstring"

function\_suite

return [expression]

2 parts

1. Function definition

2. Function call

Function Definition

===============

a) Function with no parameter and no return value

-------------------------------------------------------------

Syntax:

---------

deffunction\_name():

# Statements

b) Function with parameter and no return value

-------------------------------------------------------------

Syntax:

---------

deffunction\_name(parameters):

# Statements

c) Function with no parameter and return value

-------------------------------------------------------------

Syntax:

---------

deffunction\_name():

# Statements

return variable

d) Function with parameter and return value

-------------------------------------------------------------

Syntax:

---------

deffunction\_name(parameter):

# Statements

return variable

Function Call

==========

a) Function with no parameter and no return value

-------------------------------------------------------------

Syntax:

---------

function\_name()

b) Function with parameter and no return value

-------------------------------------------------------------

Syntax:

---------

function\_name(arguments)

c) Function with no parameter and return value

-------------------------------------------------------------

Syntax:

---------

variable = function\_name()

d) Function with parameter and return value

-------------------------------------------------------------

Syntax:

---------

variable = function\_name(arguments)

**PROGRAMS:**

**a) Number Conversion**

**Description:**

To convert the given decimal number into binary, octal and hexadecimal numbers using user defined functions

Get the option for number conversion:

Option 1 - Binary to Decimal

Option 2 - Decimal to Binary

Option 3 - Decimal to Octal

Option 4 - Decimal to Hexadecimal

Use Type 1 function for Option 1

Use Type 2 function for Option 2

Use Type 3 function for Option 3

Use Type 4 function for Option 4

The first input is to specify the value for the option

The second input is to specify the number

Sample Input

1

1101

Sample Output

13

Sample Input

2

-7

Sample Output

Give a positive number

Sample Input

2

10

Sample Output

1010

Sample Input

3

33

Sample Output

41

**Program:**

'''NameR.sridevi

Roll.No:20UIT021

Program name: To convert the given number into binary, octal and hexadecimal numbers using user defined functions.'''

def bin\_to\_Deci():

pow1=sum1=0

number=int(input())

while(number>0):

rem=number%10

sum1+=(rem\*(2\*\*pow1))

number=number//10

pow1+=1

print(sum1)

def Deci\_to\_bin(number):

binary=''

reverse=i=0

while(number>0):

rem=number%2

binary+=str(rem)

number=number//2

reverse=binary[::-1]

print(int(reverse))

def Deci\_to\_octal():

number=int(input())

binary=reverse=0

if(number>0):

while(number>0):

rem=number%8

binary=binary\*10+rem

number=number//8

while(binary>0):

rem=binary%10

reverse=reverse\*10+rem

binary=binary//10

return reverse

def Deci\_to\_Hexa(number):

number=int(input())

binary=reverse=0

if(number>0):

while(number>0):

rem=number%16

binary=binary\*10+rem

number=number//16

while(binary>0):

rem=binary%10

reverse=reverse\*10+rem

binary=binary//10

return reverse

option=int(input())

if(option==1):

bin\_to\_Deci()

elif(option==2):

number=int(input())

if(number>0):

Deci\_to\_bin(number)

else:

print("Give a positive number")

elif(option==3):

octal=Deci\_to\_octal()

print(octal)

elif(option==4):

number=int(input())

if(number>0):

Hexa=Deci\_to\_Hexa(number)

print(Hexa)

else:

print("Give a positive number")

Deci\_to\_Hexa()

**Test Cases:**

|  |  |  |
| --- | --- | --- |
| **Test Case No.** | **Input** | **Expected Output** |
| 1 | 1  1101 | 13 |
| 2 | 2  -7 | Give a positive number |
| 3 | 2  10 | 1010 |
| 4 | 3  33 | 41 |
| **Total Test Cases** | | **4** |
| **Number of Test Cases Passed** | | **4** |

**b) To print all pronic numbers between a range**

**Description:**

Pronic number is a number which is the product of two consecutive integers, that is, a number n is a product of x and (x+1).

Example:

If the input is 6 then it is the product of 2\*3 which has a consecutive integers.

Get the positive starting and ending limit.

Sample Input

5

15

Sample Output

6

12

Sample Input

10

5

Sample Output

Starting limit is greater than ending limit

Sample Input

10

-20

Sample Output

Limit cannot be negative

**Program:**

'''Name: R.sridevi

Roll Number:20UIT021

Program Name:To print all pronic numbers between a range'''

num1=int(input())

num2=int(input())

if(num2<0 or num1<0):

print("Limit cannot be negative")

elif(num1>num2):

print("Starting limit is greater than ending limit")

else:

for i in range(num1,num2+1):

flag = 0

for j in range(0, i + 1):

if j \* (j + 1) == i:

flag = 1

break

if flag == 1:

print(i,'\n')

**Test Cases:**

|  |  |  |
| --- | --- | --- |
| **Test Case No.** | **Input** | **Expected Output** |
| 1 | 5  15 | 6  12 |
| 2 | 10  -20 | Limit cannot be negative |
| 3 | 10  5 | Starting limit is greater than ending limit |
| 4 | 0  15 | 0  2  6  12 |
| 5 | 100  150 | 110  132 |
| 6 | -9  18 | Limit cannot be negative |
| 7 | -6  -10 | Limit cannot be negative |
| **Total Test Cases** | | **7** |
| **Number of Test Cases Passed** | | **7** |

**RESULT:**

Thus, the Python programs are executed successfully.