PROGRAMMING WITH PYTHON
LAB MANUAL
B.Tech, 4th Semester

Experiment: 1
Objective: *To Familiarization with programming environment.*

I. Write a "Python" program to calculate area and perimeter of rectangle.

Program Code:

```
# Python Program to find Area of a Rectangle

width = float(input('Please Enter the Width of a Rectangle: '))
height = float(input('Please Enter the Height of a Rectangle: '))

# calculate the area
Area = width * height

# calculate the Perimeter
Perimeter = 2 * (width + height)

print("\n Area of a Rectangle is: %.2f" %Area)
print(" Perimeter of Rectangle is: %.2f" %Perimeter)
```

II. Write a „Python" program to convert length in feet to Centimetre

Program Code:

```
print("Input your height: ")

h_ft = int(input("Feet: "))

h_inch = int(input("Inches: "))


h_inch += h_ft * 12

h_cm = round(h_inch * 2.54, 1)


print("Your height is : %d cm." % h_cm)
```

Experiment: 2

Objective: *To implement Simple computational problems*

I. Write a "Python" program to swap two numbers using third Variable

```
# Python program to swap two variables

# To take input from the user
# x = input('Enter value of x: ')
# y = input('Enter value of y: ')

x =  int (input('Enter value of x: '))
y =  int (input('Enter value of y: '))

# create a temporary variable and swap the values
temp = x
x = y
y = temp

print('The value of x after swapping: {}'.format(x))
print('The value of y after swapping: {}'.format(y))
```

II. Write a "Python" program to print ASCII Value of a character (using "ord()" function)

Program Code:

```
# Program to find the ASCII value of the given character

# Change this value for a different result

c = input("Enter a character: ")


# Uncomment to take character from user

#c = input("Enter a character: ")


print("The ASCII value of '" + c + "' is",ord(c))
```

Experiment: 3

Objective: *To implement Problems involving if-then-else structures*

I. Write a "Python" program to check whether a number is even or odd.

Program Code:

```
# Python program to check if the input number is odd or even.
# A number is even if division by 2 give a remainder of 0.
# If remainder is 1, it is odd number.
num = int(input("Enter a number: "))
if (num % 2) == 0:
print("{0} is Even".format(num))
else:
print("{0} is Odd".format(num))
```

II. Write a "Python" program to determine largest among three numbers.

Program Code:

```
# Python program to find the largest number among the three input numbers
# take three numbers from user
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
num3 = float(input("Enter third number: "))

if (num1 > num2) and (num1 > num3):
```

```
    largest = num1
elif (num2 > num1) and (num2 > num3):
    largest = num2
else:
    largest = num3


print("The largest number is",largest)
```

III.Write a "Python" program for GCD of more than two (or array) numbers

Program Code:

```
# GCD of more than two (or array) numbers
# This function implements the Euclidian
# algorithm to find H.C.F. of two number

a=int(input("enter 1st no:"))
b=int(input("enter 2nd no:"))
c=int(input("enter 3rd no:"))
def gcd(a,b):
    if b!=0:
        return gcd(b,a%b)
    else:
        return a
d=gcd(a,b)
if gcd==1:
print("gcd = 1")
else:
    for i in range(0,c):
        m=gcd(d,c)
print("gcd is ",m)
```

OR,

```
def findgcd(x, y):
    while(y):
        x, y = y, x % y
    return x
l = [int(i) for i in input().split()]
num1=l[0]
num2=l[1]
gcd=findgcd(num1,num2)
for i in range(2,len(l)):
gcd=findgcd(gcd,l[i])
print("gcd is: ",gcd)
```

Experiment: 4

Objective: *To implement Problems involving if-then-else structures & logical operator*

I. Write a "Python" program to determine whether an year is Leap Year or not

Program Code:

```
#Python 3.7.1 (default, Dec 10 2018, 22:54:23) [MSC v.1915 64 bit (AMD64)] on win32
#Type "help", "copyright", "credits" or "license()" for more information.
year = int(input("Enter a year: "))
if (year % 4) == 0:
   if (year % 100) == 0:
       if (year % 400) == 0:
print("{0} is a leap year".format(year))
       else:
print("{0} is not a leap year".format(year))
   else:
print("{0} is a leap year".format(year))
else:
print("{0} is not a leap year".format(year))
```

II. Write a "Python" program to check whether a character is uppercase or lowercase alphabet.

Program Code:

```
# Python Program to check character is Lowercase or Uppercase
ch = input("Please Enter Your Own Character : ")
```

```
if(ch.isupper()):
print("The Given Character ", ch, "is an Uppercase Alphabet")
elif(ch.islower()):
print("The Given Character ", ch, "is a Lowercase Alphabet")
else:
print("The Given Character ", ch, "is Not a Lower or Uppercase Alphabet")
```

III. Write a "Python" program to determine largest among three numbers using ternary operator.

```
# Python3 program to find largest
# among three numbers using
# ternary operator

# As python doesn't have ternary
# operator, we use conditional
# expression, in form of
# conditional expression:
# Here, mx is a variable
n1 = 5
n2 = 10
n3 = 15

mx = (n1 if (n1 > n2 and n1 > n3) else
        (n2 if (n2 > n1 and n2 > n3) else n3))

# We also can use
# mx=(n1>n2) and n1 or n2

# str() function is used to
```

```
# convert integer to string,
# it is required as we are
# concatenating integers with string
print("Largest number among " + str(n1) +
                                    " , " + str(n2) +
                                " and " + str(n3) +
                                " is " + str(mx))
```

Experiment: 5

Objective: *To implement Iterative problems using loop*

I. Write a "Python" program to find the sum of n natural numbers using while, do while and for loop (n>0).

```
n=int(input("Enter a number: "))
sum1 = 0
while(n > 0):
   sum1=sum1+n
   n=n-1
   #print(n)
print("The sum of first n natural numbers is",sum1)
```

OR

```
n = input("Enter Number to calculate sum")
n = int (n)
average = 0
sum = 0
for num in range(0,n+1,1):
   sum = sum+num
print("SUM of first ", n, "numbers is: ", sum )
```

II. Write a "Python" program to determine factorial of a given Number

```
# This Python Program finds the factorial of a number


def factorial(num):
    """This is a recursive function that calls
   itself to find the factorial of given number"""
    if num == 1:
        return num
    else:
        return num * factorial(num - 1)


# We will find the factorial of this number
num = int(input("Enter a Number: "))


# if input number is negative then return an error message
# elif the input number is 0 then display 1 as output
# else calculate the factorial by calling the user defined function
if num< 0:
print("Factorial cannot be found for negative numbers")
elifnum == 0:
print("Factorial of 0 is 1")
else:
print("Factorial of", num, "is: ", factorial(num))
```

III. Write a "Python" program for Sum of squares of first n natural numbers.

```
def squaresum(n) :
    # Iterate i from 1
    # and n finding
    # square of i and
    # add to sum.
```

```
sm = 0
   for i in range(1, n+1) :
sm = sm + (i * i)
   return sm
# Driven Program
n = int(input("Enter a Number: "))
print(squaresum(n))
```

Experiment: 6

Objective: *To implement Iterative problems using loop*

I. Write a "Python" program to determine whether a number is prime or not.

```
num = int(input("enter a number: "))


for i in range(2, num):
        if num % i  == 0:
                print("not prime number")
                break
else:
        print("prime number")
```

II. Write a "Python" program to determine whether a number is Armstrong number or not.

```
input_num = (input("Enter any number: "))
digit_len = len(str(input_num))
try:
arm_num = 0
val = int(input_num)
   while val> 0:
     reminder = val % 10
arm_num += reminder ** digit_len
```

```
val //= 10
   if (int(input_num) == arm_num):
print(input_num, 'is an ARMSTRONG number')
   else:
print(input_num,  'is NOT an armstrong number')
except ValueError:
print("That's not a valid number, Try Again !")
```

Experiment: 7

Objective: *To implement 1D Array manipulation*

I. Write a "Python" program to find sum of n element of an array.

```
def _sum(arr,n):

   # return sum using sum
   # inbuilt sum() function
   return(sum(arr))

# driver function
#arr=[]
# input values to list
arr = [int(i) for i in input().split()]

# calculating length of array
n = len(arr)

ans = _sum(arr,n)

# display sum
print ("Sum of the array is ",ans)
```

II. Write a "Python" program to find the largest number of an array.

```
# Python3 program to find maximum
# in arr[] of size n

# python function to find maximum
# in arr[] of size n
def largest(arr,n):
```

```
    # Initialize maximum element
    max = arr[0]

    # Traverse array elements from second
    # and compare every element with
    # current max
    for i in range(1, n):
        if arr[i] > max:
            max = arr[i]
    return max

# Driver Code
arr = [int(i) for i in input().split()]
n = len(arr)
Ans = largest(arr,n)
print ("Largest in given array is",Ans)

# This code is contributed by Smitha Dinesh Semwal
```

III. Write a "Python" program to implement linear search in an Array

```
def linear_search(alist, key):
    """Return index of key in alist. Return -1 if key not present."""
    for i in range(len(alist)):
        if alist[i] == key:
            return i
    return -1


alist = input('Enter the list of numbers: ')
alist = alist.split()
alist = [int(x) for x in alist]
key = int(input('The number to search for: '))

index = linear_search(alist, key)
if index < 0:
print('{} was not found.'.format(key))
else:
print('{} was found at index {}.'.format(key, index))
```

Experiment: 8.

Objective: *To implement searching and sorting techniques using 1D Array manipulation*

I. Write a 'Python' program to implement binary search in sort array.

```
# Python code to implement iterative Binary search
def binarySearch(arr, l, r, x):

    while l <= r:

            mid = int(l + (r - l)/2);

            if arr[mid] == x:
                    return mid

            elifarr[mid] < x:
                    l = mid + 1

            else:
                    r = mid - 1

    return -1


arr = [int(i) for i in input().split()]
x =  int (input('Enter value of x: '))

result = binarySearch(arr, 0, len(arr)-1, x)

if result != -1:
    print ("Element is present at index ",  result)
else:
```

```
print ("Element is not present in array")
```

II. Write a "Python" program to sort an array using bubble sort algorithm.

```
# Python program for implementation of Bubble Sort

def bubbleSort(arr):
    n = len(arr)

    for i in range(n):

        for j in range(0, n-i-1):

            # traverse the array from 0 to n-i-1
            # Swap if the element found is greater
            # than the next element
            if arr[j] >arr[j+1] :
                arr[j], arr[j+1] = arr[j+1], arr[j]

# Driver code to test above
arr = [int(i) for i in input().split()]

bubbleSort(arr)

print ("Sorted array is:")
for i in range(len(arr)):
    print ("%d" %arr[i]),
```

Experiment: 9

Objective: *To implement Simple functions programs*

I. Write a "Python" program to calculate addition of two numbers using function.

```
def sum(x,y):
ans = x+y
   return ans

num1 = int(input("Enter first number:"))
a = num1
num2 = int(input("Enter second number:"))
b = num2

sum(a,b)
print(sum(a,b))
```

II. Write a "Python" program to swap two number using function call.

```
a=int(input("enter a no :"))
b=int(input("enter another no :"))
print("Before swapping:")
print("Value of a==",a,"and b==",b)
def swap(a,b):
   temp=a
   a=b
   b=temp
```

```
print("after swapping \n value of a==",a,"and b==",b)
swap(a,b)
```

Experiment: *10*

Objective: *To implement String operations Programs*

I. Write a "Python" program to find length of a string using string function and also without using string function.

Using string function

```
str = input("Enter a string: ")
print(len(str))
```

With outUsing string function

```
str = input("Enter a string: ")

# counter variable to count the character in a string
counter = 0
for s in str:
    counter = counter+1
print("Length of the input string is:", counter)
```

II. Write a "Python" program to remove all duplicates from a given String

```
from collections import Counter
def remov_duplicates(str):
```

```
    # split input string separated by space
    str = str.split(" ")

    # joins two adjacent elements in iterable way
    for i in range(0, len(str)):
            str[i] = "".join(str[i])

    # now create dictionary using counter method
    # which will have strings as key and their
    # frequencies as value
    UniqW = Counter(str)

    # joins two adjacent elements in iterable way
    s = " ".join(UniqW.keys())
    print (s)

# Driver program
if __name__ == "__main__":
    str = input("Enter a string: ")
    remov_duplicates(str)
```

Experiment: 11

Objective: *To implement String operations Programs*

I. Write a "Python" program to check if a string is palindrome or not using string function and also without using string function.

Using String

```
# function which return reverse of a string
def reverse(s):
    return s[::-1]

def isPalindrome(s):
    # Calling reverse function
    rev = reverse(s)

    # Checking if both string are equal or not
    if (s == rev):
            return True
    return False


# Driver code
s = input("Enter a string: ")
ans = isPalindrome(s)
```

```
if ans == 1:
    print("Yes")
else:
    print("No")
```

Without using string

```
# Python Program to Check a Given String is Palindrome or Not

string = input("Please enter your own String : ")
str1 = ""

for i in string:
   str1 = i + str1
print("String in reverse Order :  ", str1)

if(string == str1):
print("This is a Palindrome String")
else:
print("This is Not a Palindrome String")
```

II. Write a "Python" program to Remove all duplicates from a given string in Python

```
from collections import Counter

def remov_duplicates(str):

    # split input string separated by space
    str = str.split(" ")

    # joins two adjacent elements in iterable way
    for i in range(0, len(str)):
            str[i] = "".join(str[i])

    # now create dictionary using counter method
    # which will have strings as key and their
    # frequencies as value
    UniqW = Counter(str)

    # joins two adjacent elements in iterable way
    s = " ".join(UniqW.keys())
    print (s)

# Driver program
if __name__ == "__main__":
    str = input("Enter a string: ")
    remov_duplicates(str)
```

Experiment: *12*
Objective: *To implement lists programs*

I. Write a "Python" program Count occurrences of an element in a list

```
# Program to count the number of times an element
# Present in the list
# Count function is used
def Count(tup, en):
    return tup.count(en)

# Driver Code
tup = arr = [(i) for i in input().split()]
enq = input('Enter your input')
#enq1 = int(input('Enter your input'))
#enq2 = int(input('Enter your input'))
print(Count(tup, enq), "times")
#print(Count(tup, enq1), "times")
#print(Count(tup, enq2), "times")
```

III. Write a "Python" program to Cloning or Copying a list

```
# Python program to copy or clone a list
# Using the Slice Operator
```

```
def Cloning(li1):
    li_copy = li1[:]
    return li_copy

# Driver Code
li1 = arr = [(i) for i in input().split()]
li2 = Cloning(li1)
print("Original List:", li1)
print("After Cloning:", li2)
```

Experiment: *13*

Objective: *To implement Tuple Program*

I. Write a "Python" program to find number of lists in a tuple

```
# Python code to find number of list in a tuple

# Initial list
Input1 = ([1, 2, 3, 4], [5, 6, 7, 8])
Input2 = ([1, 2], [3, 4], [5, 6])
Input3 = ([9, 8, 7, 6, 5, 4, 3, 2, 1], [1, 2, 3])

# Using len to find no of list in tuple
Output1 = len(Input1)
Output2 = len(Input2)
Output3 = len(Input3)

# Printing
print("Initial list :")
print(Input1)
print("No of list in tuples are :")
print(Output1)
print("\n")

print("Initial list :")
```

```
print(Input2)
print("No of list in tuples are :")
print(Output2)
print("\n")


print("Initial list :")
print(Input3)
print("No of list in tuples are :")
print(Output3)
print("\n")
```

OR,

```
values=input("input the list element seperated with space and each list seperated with ',':")
list=values.split(",")
tuple=tuple(list)
print("tuple :",tuple)
l=len(tuple)
print("The num of list present in the tuple is:",l)
```

II. Write a "Python" program to remove matching tuples

```
# Python3 code to demonstrate
# filter repeated tuple
# using list comprehension

# initializing lists
test_list1 = [('Geeks', 'for'), ('Geeks', 'is'), ('Computer', 'Science')]
test_list2 = [('Geeks', 'for'), ('Geeks', 'is')]

# printing original lists
print("The original list 1 : " + str(test_list1))
print("The original list 1 : " + str(test_list2))

# using list comprehension
# filter repeated tuple
res = [sub for sub in test_list1 if sub not in test_list2]

# print result
print("The filtered list of tuples : " + str(res))
```

Experiment: 14

Objective: *To implement Dictionaries program*

I. Write a "Python" program Check if binary representations of two numbers are anagram

```
# function to Check if binary representations
# of two numbers are anagram
from collections import Counter

def checkAnagram(num1,num2):

    # convert numbers into in binary
    # and remove first two characters of
    # output string because bin function
    # '0b' as prefix in output string
    bin1 = bin(num1)[2:]
    bin2 = bin(num2)[2:]

    # append zeros in shorter string
    zeros = abs(len(bin1)-len(bin2))
    if (len(bin1)>len(bin2)):
            bin2 = zeros * '0' + bin2
    else:
```

```
            bin1 = zeros * '0' + bin1

    # convert binary representations
    # into dictionary
    dict1 = Counter(bin1)
    dict2 = Counter(bin2)

    # compare both dictionaries
    if dict1 == dict2:
            print('Yes')
    else:
            print('No')

# Driver program
if __name__ == "__main__":
    num1 = int(input("Enter first number: "))
    num2 = int(input("Enter first number: "))
    checkAnagram(num1,num2)
```

III. Write a program to create grade calculator in Python

```
# Python Program - Calculate Grade of Student

print("Enter 'x' for exit.");
print("Enter marks obtained in 5 subjects: ");
mark1 = input();
if mark1 == 'x':
exit();
else:
   mark1 = int(mark1);
   mark2 = int(input());
   mark3 = int(input());
   mark4 = int(input());
   mark5 = int(input());
   sum = mark1 + mark2 + mark3 + mark4 + mark5;
   average = sum/5;
   if(average>=91 and average<=100):
     print("Your Grade is A+");
elif(average>=81 and average<=90):
     print("Your Grade is A");
elif(average>=71 and average<=80):
     print("Your Grade is B+");
elif(average>=61 and average<=70):
     print("Your Grade is B");
elif(average>=51 and average<=60):
     print("Your Grade is C+");
elif(average>=41 and average<=50):
     print("Your Grade is C");
```

```
elif(average>=0 and average<=40):
    print("Your Grade is F");
  else:
    print("Strange Grade..!!");
```

Experiment: 15

Objective: *To implement different type of Modules concept*

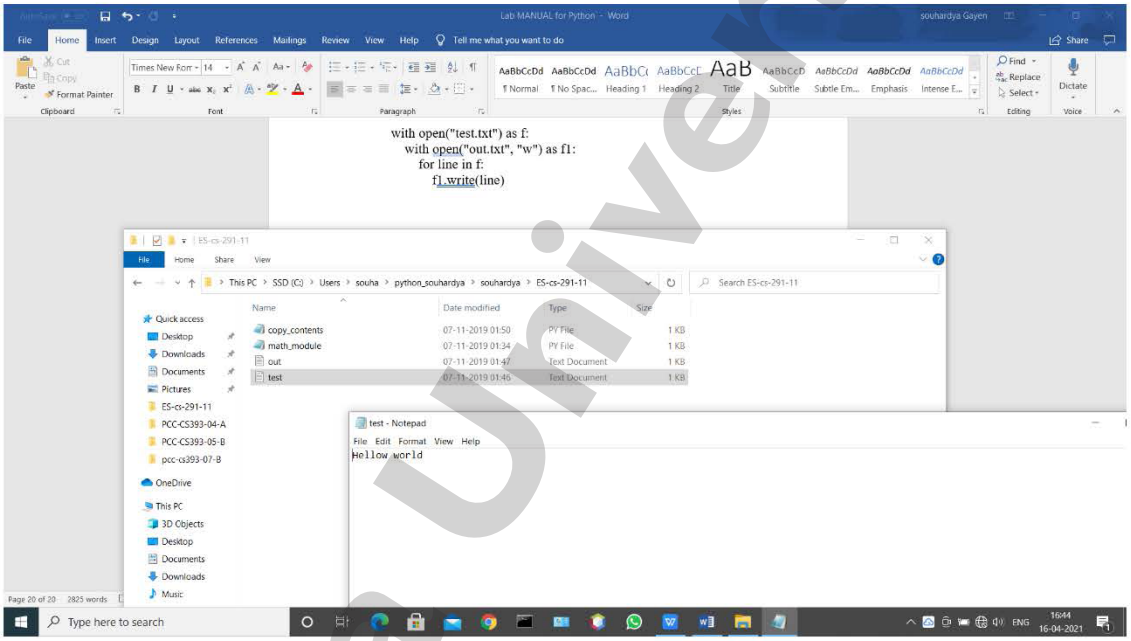
I. Write a program in "Python" by importing math module to check the use of
i) ceil() ii) floor() iii) fabs() iv) factorial()
v) copysign() vi) gcd()

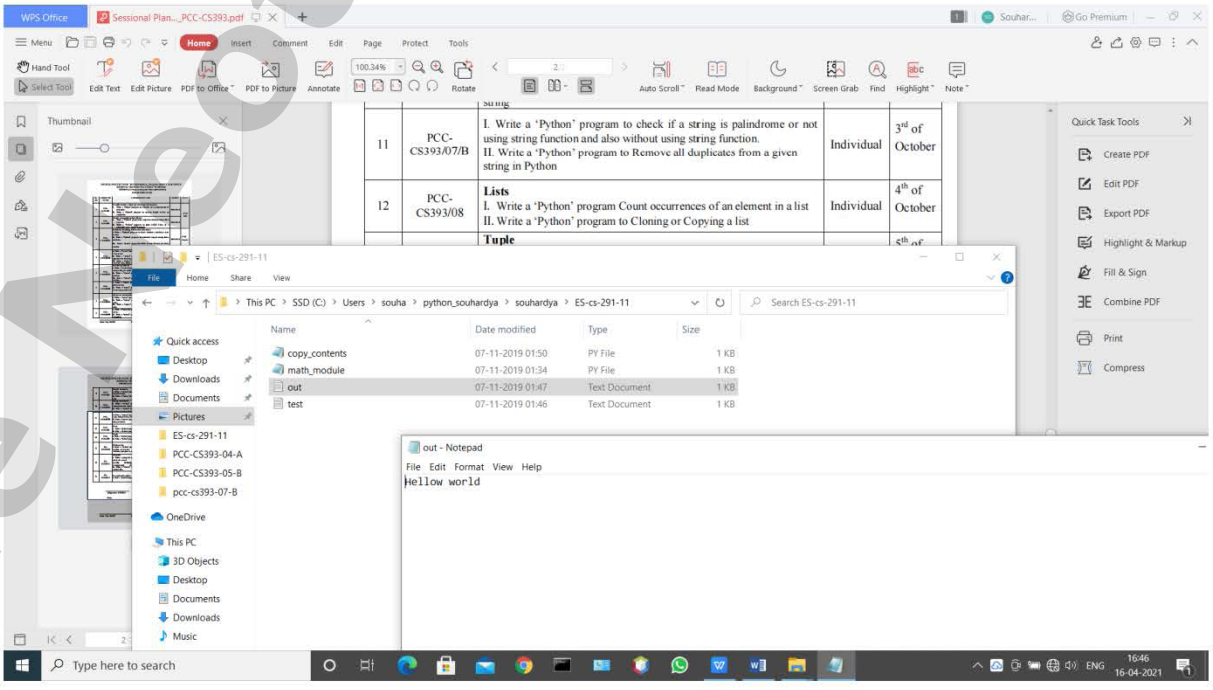
```
from math import *
n=eval(input("Enter a number:"))
p=eval(input("Enter a number:"))
print(ceil(n))
print(floor(n))
print(fabs(n))
print(factorial(n))
print(copysign(n,p))
print(gcd(n,p))
```

II. Write a "Python" program to copy the contents of a file to

another file.

```
with open("test.txt") as f:
    with open("out.txt", "w") as f1:
        for line in f:
            f1.write(line)
```





Experiment: 16

Objective: *To implement Exception Handling Programs*

I. Write a "Python" program to check Multiple Exception Handling

```
# import module sys to get the type of exception
import sys

randomList = ['a', 0, 2]

for entry in randomList:
   try:
print("The entry is", entry)
      r = 1/int(entry)
      break
   except:
      print("Oops!",sys.exc_info()[0],"occured.")
```

```
print("Next entry.")
print()
print("The reciprocal of",entry,"is",r)
```

OR,

```
import math

def square(x):
    if int(x) is 0:
        raise ValueError('Input argument cannot be zero')
    if int(x) < 0:
        raise TypeError('Input argument must be positive integer')
    return math.pow(int(x), 2)
while True:

    try:
        y = square(input('Please enter a number\n'))
        print(y)
    except ValueError as ve:
        print(type(ve), '::', ve)
    except TypeError as te:
        print(type(te), '::', te)
while True:

    try:
        y = square(input('Please enter a number\n'))
        print(y)
    except (ValueError, TypeError) as e:
        print(type(e), '::', e)
```