Name :-> Dhruv Gandhi
Enrollment Number :-> 17SE02CE017
Department :-> Computer
Semester :-> 6th
Subject Name with code :-> SEIT3050 => Application Development Using open source technology.

Q1 What is the different type of data type in python?

=> in python we can store variable of different type.
=> python having built in datatype in default.

★ Type of Datatype

i> Text	v> Set
ii> Numeric	vi> Boolean.
iii> Sequence	vii> Binary
iv> mapping	

★ Text Datatype
we can store variable by using "str"

code x = "Hi"
print (type (x))

output -> class 'str'

★ Numeric Data type.
We can store ~~data~~ variable using "int, float, complex"

code = x = 200

y = 20.0

z = 2j

print (type (x))
print (type (y))
print (type (z))

output => < class 'int' >
< class 'float' >
< class 'complex' >

* Sequence data type.

~~Sequence~~

Sequence means we can store mutiple ~~variable~~ data in on variable. using "list, tuple, range"

code. x = ["a", "b", "c"]

y = ("a", "b", "c")

z = range(6).

print (type (x))
print (type (y))
print (type (z))

output < class 'list' >
< class 'tuple' >
< class 'range' >

PAGE NO.:
DATE: / /

* mapping Datatype.

mapping means we can map variable with value using 'dict'.

code x = {"name": "js", "age": "67"}
print (type (x))
output => <class 'dict'>

* set Datatype

set is number of variable using set, Frozenset.

code x = {"a", "b", "c"}
Y = Frozenset ({"a", "b", "c"})
print (type (x))
print (type (Y))
output => <class 'set'>
<class 'Frozenset'>

* Boolean Datatype

Boolean means it return Yes or No means true or fake using bool.

code x : True
print (type (x))
output => <class 'true'>

Binary Datatype.

Q1 :- a. Binary datatype can be store using bytes, bytearray, memoryview.

```
Code    x = b"a"
        y = bytearray(5)
        z = memoryview(bytes(5))
print (x)
print (y)
print (z)
print (type(x))
print (type(y))
print (type(z))
```

```
Code => b'a'
        bytearray(b'\x00\x00\x00\x00\x00')
        <memory at 0x00B587A0>
        <class 'bytes'>
        <class 'bytearray'>
        <class 'memoryview'>
```

Q2 Function and operator overloading can be performed in python and method overloading cannot.

Q3 False array and lists are not same.

Q4 Assume that i loop from 1 to 5 what will be the output of print (i, end = "s")?

Code => for i in range (1, 5):
print (i)

output => 1
2
3
4
5

Q7 How do you perform type conversion in python?

=> python defines type conversion functions to directly convert one data to another which is useful in day to day and competitive programming.

i> int (a, b) => this function convert any data type to integer. 'B' specifies the base in which string is. if data type is string.

ii> float ( ) :-> function is used to convert any data type to a floating point number.

iii> ord ( ) :-> to convert a character to integer.

iv> hex ( ) => to convert integer to hexadecimal string.

v> oct ( ) :-> to convert integer to octal string.

vi> tuple ( ) :-> to convert to a tuple.

vii> set ( ) :-> returns the type after converting to set.

viii> list ( ) :-> to convert any data type to a list type.

ix> dict ( ) :-> to convert a tuple of order (key, value) into a dictionary.

x> str ( ) :-> to convert integer into a string.

xi> complex (real, img) :-> convert real number to complex (real, img) number.

xii> chr (number) :-> converts number its corresponding ASCII character.

DATE: / /

D = "11111" H = "4" R = "DHR"

code

```
a = int (D, e)
b = float (D)
c = ord (H)
d = hex (56)
e = oct (56)
f = tuple (R)
g = set (R)
h = list (R)
i = complex (D, H)
j = str (D)
typ = (('a', 1), ('f', 2), ('g', 3))
k = dict (typ)

L = chr (76)
```

	output
print (a)	-> 18
print (b)	-> 11111.0
print (c)	-> 52
print (D)	-> 0x38
print (e)	-> 0o70
print (f)	-> ('D', 'H', 'R')
print (g)	-> {'D', 'H', 'R'}
print (h)	-> ['D', 'H', 'R']
print (i)	-> (11111+4j)
print (j)	-> 11111
print (k)	-> {'a': 1, 'f': 2, 'g': 3}
print (L)	-> L

Q11 = Write a program to show the use of Shorthand if-else

Code => a = 756
b = 765

print ("a") if a>b else print ("=") if a==b else print ("B").

Output => B.

Q12 Write a program to find the maximum of three numbers using nested if else.

Code =>
```
a = int (input ("Enter 1st Num: "))
b = int (input ("Enter 2nd Num: "))
c = int (input ("Enter 3rd Num: "))
if a>b:
    if a>c:
        d = a
    else:
        d = c
else:
    if b>c:
        d = b
    else:
        d = c
print ("maximum number is = ", c).
```

Output :-

Enter 1st Number : 75
Enter 2nd Number : 67
Enter 3rd Number : 51
Maximum Number is = 75

Q13 write program to show of else with loops.

★ code :> #with For loop

```
For n in range(10,20):
    for i in range(2,num):
        if n%i ==0:
            j = n/i
            print '%d Equals %d * %d' %(n,i,j)
            break
    else:
        print n, 'is a prime number'.
```

output

→ 10 Equals 2*5
11 is a prime number
12 Equals 2*6
13 is a prime number
14 Equals 2*7
15 Equals 3*5
16 Equals 2*8
17 is a prime nuber
18 Equals 2*9
19 is a prime nubers.

★ code => # with while loop.

```
a=0
while a<5:
    a = a+1
    print ("iteration no {} in while loop".format(a))
else:
    print ("else block in loop")
print ("out of loop").
```

Output =>
```
iteration no 1 in for loop.
iteration no 2 in for loop.
iteration no 3 in for loop.
iteration no 4 in for loop.
iteration no 5 in for loop.
else block in loop.
out of loop.
```

Q14 Write a users defined function which returns a lambda function. then call the lambda function.

code
```
a = int(input("a:"))
b = int(input("b:"))
def sumF(x,y):
    add = lambda x,y: x+y
    return add.
def subF(x,y):
    sub = lambda x,y: x-y
    return sub
def mulF(x,y):
    mul = lambda x,y: x*y
    return mul.
def divF(x,y):
    div = lambda x,y: x/y
    return div.
w = sumF(a,b)
print(w(a,b))
x = subF(a,b)
print(x(a,b))
y = mulF(a,b)
print(y(a,b))
z = divF(a,b)
print(z(a,b))
```

Output
```
a: 17
b: 3
20
14
51
5.66666667
```

DATE: / /

Q 15

What is the difference between iterative and Recursive function?

	Recursion	iterative
Definition	Recursion refers to a recursive function in which it calls itself again to repeat the code.	iterative is archived by an iterative function which loops to repeat some section of the code.
important point	A base case need to be determined.	A termination condition needs to be determined.
performance	Slow	Fast.
memory Usage	more	less
Code Size	Smaller	longer
Structure	Selection	Repetition
local variable	Not required	required.
infinite repetition	infinite recursion is capable of crashing the System.	infinite looping consume CPU cycles repeatedly.

code of programme	def fun(a,b):
	~~while (b):~~
	~~a,b = (b, a%b)~~
	~~return~~
	if (b == 0):
	return a
	else:
	return fun(b, a%b)
		def fun(a,b):
		while (b):
		a,b = (b, a%b)
		return a

PAGE NO.:
DATE: / /

Q 18 what is a use of __init__ Fuction in class

code => # __init__ Function in heritance

```
class User(object):
    def __init__(self, a):
        print("User init called.")
        Self.a = a.
    def method(self)
        return Self.a
class Student(User):
    def __init__(self, a):
        User.__init__(self, a)
        print("student initcalled").
        Self.a = a
    def method(self)
        return Self.a
my_object = Student('Dhruv')
```

Output => User Init called.
Student Init called.

Q 19 Create a module which has recursive function. import that module in another file and access the recursive function.

code => # create one file add.py as recursive function

```
def recur_sum(n):
    if n <= 1:
        return n
    else:
        return n + recur_sum(n-1).
```

in new file

```
import add
a = int(input("Enter number"))
if a < 0:
    print("Enter a positive number")
else:
    print("the sum is", add.recur_Sum(a))
```

Q20 Explain at least five Exceptions and handle them with a program.

code =>

```
a = 10
b = 0

try:
    c = a/b
    v = int(x)
    import recu
    f = open("d.txt", rt)
    it = iter([1,2,3])
    next(it)
except:
    if (ZeroDivisionError):
        print("divide by zero")
    if (TypeError):
        print("type Error")
    if (moduleNotFoundError):
        print("module not found")
    if (NameError):
        print("File not found")
    if (stopIteration):
        print("Array out of bound")
```

```
else:
    print ("error not found")

finally:
    print ("Finally Excepted")
```

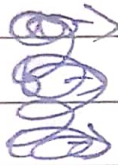
output =>
divide by zero
type Error
module not found
file not found
Array out of bound
Finally Excepted.

Q23 what do you mean by \A, \G, \B, \d, \D, \w, \W, \s, \S, \z in regular Expression.

Code

```
import re
x = re.findall("\ATHE", txt)
y = re.findall("\d", text)
z = re.findall("\D", txt)
a = re.findall("\w", txt)
b = re.findall("\W", txt)
c = re.findall("\s", txt)
d = re.findall("\S", txt)
e = re.findall("Spain\z", txt)
f = re.findall("\bin", txt)
g = re.findall("\Bin", txt)
txt = "The rain in Spain"
print(x)  # check string start with the
print(y)  # Check digit is there
print(z)  # return no digit is there
print(a)  # check every word character
print(b)  # return no word character
print(c)  # check every white space character.
print(d)  # Return every non white space character.
print(e)  # check end with Spain
print(f)  # Find in starting of Ending
print(g)  # Return NOT in starting or Ending
```

Output



x -> ['ThE']

y -> []

z -> ['T','H','E',' ','r','a','i','n',' ','i','n',' ','S','p','a','i','n']

a -> ['T','H','E',' ','ra','i','n',' ','in',' ','S','p','a','i','n']

b -> [' ',' ',' ',' ',' ']

c -> [' ',' ',' ',' ',' ']

d -> ['T','H','E',' ','r','a','i','n',' ','i','n',' ','S','p','a','i','n']

e -> ['Spain']

f -> []

g -> ['in','in']