Solution for Internal Assessment Test 1 – Nov 2021
Course Instructors: Dr. P. N. Singh & Mrs. Sherly

Sub:	Application Development using Python					Sub Code:	18CS55	Branch:	CSE	
Date:	13-11-2021	Duration:	90 mins	Max Marks:	50	Sem / Sec:	A, B & C			OBE

Answer any FIVE FULL Questions

		MARKS	CO	RBT
1 (a)	Explain elif, for, while, break and continue statements in Python with one example program.	[10]	CO1	L2

elif

► elif is an else if that follows an if or another elif statement
► Provides another condition to be checked if previous condition evaluates to False.
► elif keyword
► Condition (boolean expression that evaluates to True or False)
► Indented block of code (elif clause)
► When there is a chain of elif clauses, only one of them will execute.
► Once one of the conditions if found to be True, rest of the elif clauses are automatically skipped.

```
 if name== “Alice”:
      print (“Alice”)
elif age < 5:
      print(“you’re just a baby”)
```

while

► Used to execute a block of code over and over again.
► while keyword
► A condition (expression that evaluates to True or False.)
► A colon
► Starting on the next line : an indented block of code – while clause
► In while loop, condition is always checked at the beginning of the while loop.
► The first time the condition evaluates to False the iteration is skipped.
► spam = 0 print("\nwhile statement")

```
while spam < 5:
      print("Hello World")
      spam =spam +1
```

Output: while statement Hello World Hello World Hello World Hello World Hello World

break

►**break** will break out of a loop when it is encountered.

```
name = ' '
    while True:
    name = input("Enter your name:")
    if name == 'Max':
        break
print('Thank you')
```

continue

► On encountering a continue statement, program execution immediately jumps back to the start of the loop.
► Reevaluates the condition.

► This is similar to what happens when execution reaches the end of the loop.

```
while True:
      print('Who are you?')
      name = input()
      if name != 'Max':
          continue
      print('Hello Max. What is your password?')
pswd = input()
if pswd == '123':
       break
print('Access granted')
```

2 (a) Write short notes on print(), input() and string replication. [03] CO1 L2

Print()

► print('Hello world!')
► print('What is your name?') # ask for their name
► print() is a function.
► The string to be displayed is passed as a value to the function
► Value passed to a function is called an argument.
► The quotes just begin and end the string and are not printed.
► It can be used to insert a blank line: print()

input()

► Waits for user to type something and hit "ENTER".
► myName=input()
► Assigns a string to a user.
► print("It is good to meet you, "+myName)
► A single string is passed to the print() function.

String Replication

► String Replication operator:
When used with one string and one integer values, it replicates the string.
► >>> "hello"*3

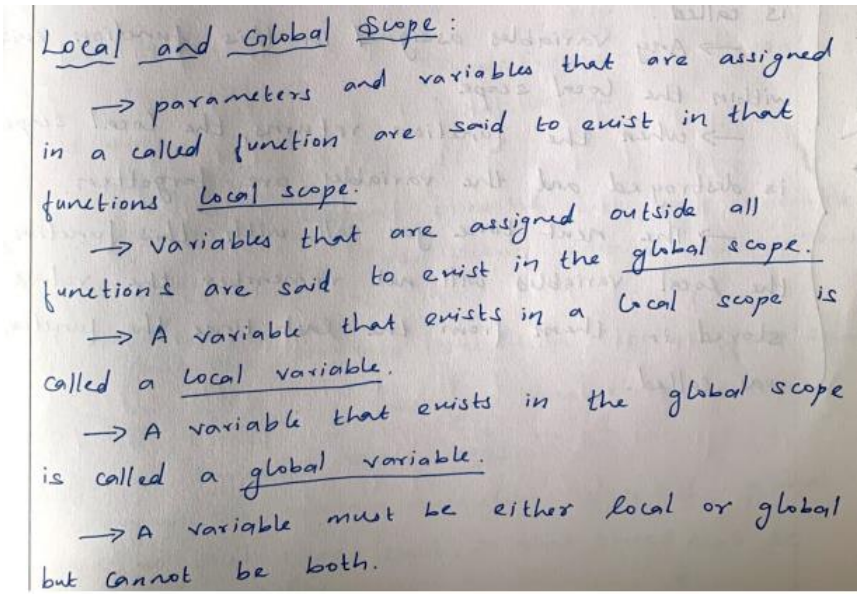
(b) Demonstrate the concept of exception with keywords try, except and finally. Implement a code which prompts the user for Celsius temperature to Fahrenheit, and printout the converted temperature by handling the exception. [07] CO1 L3

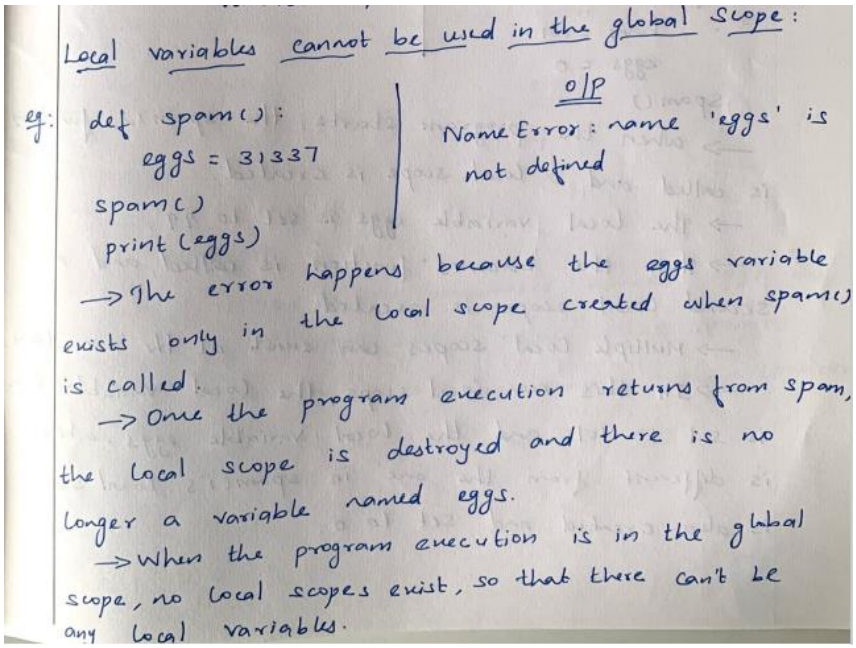
► If an error is encountered, the python program crashes.
► Practically, it is better if errors were handled gracefully.
► Detect errors, handle them, continue to run.
► Eg. Divide-by-zero error
► Syntactically the program is correct and hence these errors cannot be detected until runtime.

```
temp = input('Enter Celsius Temperature:')
#celcius=(farenheit-32)%1.8
#farenheit=celcius*1.8+32
try:
 celcius = float(temp)
 fahrenheit = celcius * 9.0 / 5.0 + 32
 print("Equivalent Temparature in Fahrenheit : ",fahrenheit )
except:
 print('Please enter a number')
```

```
Enter Celsius Temperature: 32
Equivalent Temparature in Fahrenheit :  89.6
```

3 (a)	What is local and global scope of variable in python? Explain the different scenarios with an example snippet.	[07]	CO1	L2





→ when bacon() returns, the local scope for that call is destroyed. The program execution continues in the spam() function to print the value of eggs.

→ since the local scope for the call to spam() still exist here, the eggs variable is set to 99.

→ The local variables in one function are completely seperate from the local variables in another function.

Global variables can be read from a local scope:

```
def spam():
    print(eggs)
eggs = 42
spam()
print(eggs)
```

o/p

```
42
42
```

→ since there is no parameter named eggs or any code that assigns eggs a value in the spam() function, when eggs is used in spam(), python considers it a reference to the global variable eggs.

Local and Global variables with the same name:

eg:

```
① def spam():
⑧     eggs = 'spam local'
⑨     print(eggs)
② def bacon():
④     eggs = 'bacon local'
⑤     print(eggs)
⑥     spam()
⑩     print(eggs)
① eggs = 'global'
② bacon()
⑪ print(eggs)
```

o/p

```
bacon local
spam local
bacon local
global
```

→ Local and global variables can have the same name in python.

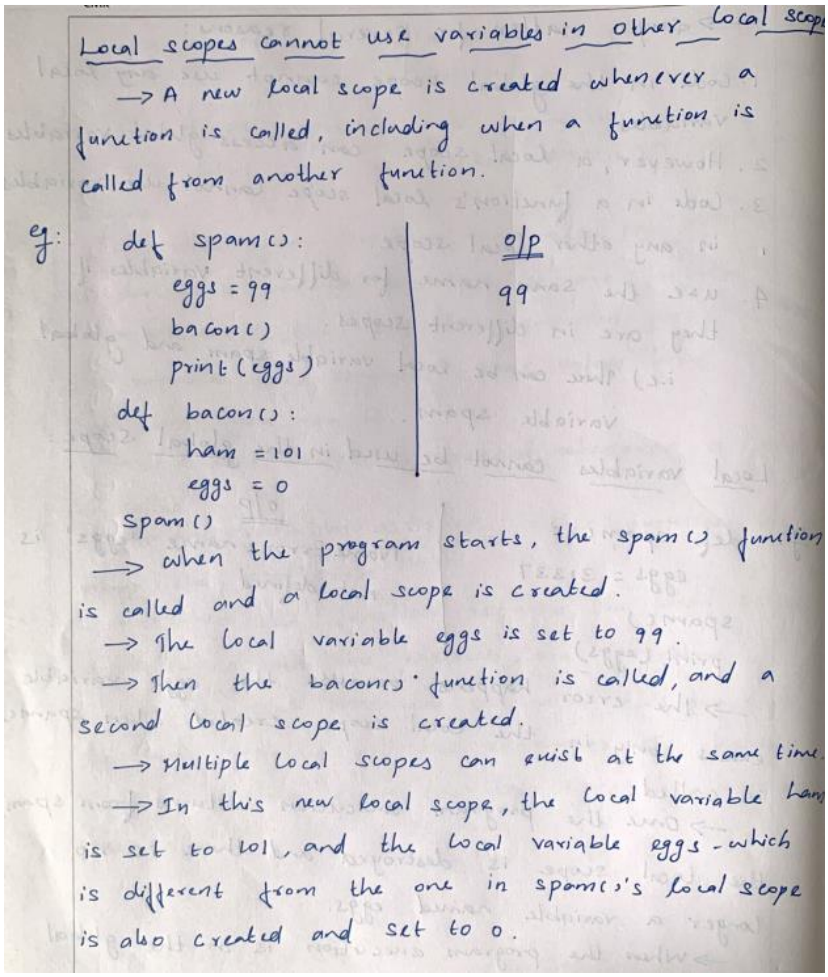
→ There are actually 3 different variables in this program. i.e) eggs. The variables are as follows:

① A variable named eggs that exists in a local scope when spam() is called.

② A variable named eggs that exists in a local scope when bacon() is called.

③ A variable named eggs that exists in the global scope.

→ Therefore avoid same name for local and global variable.



(b) Write a python program to calculate the area of circle. Read radius from user and print the results. [03] CO1 L3

```
PI = 3.14
r = float(input("Enter the radius of a circle:"))
area = PI * r * r
print("Area of a circle = %.2f" %area)
```

```
Enter the radius of a circle: 2
Area of a circle = 12.56
```

4 (a) What is a function? How to define a function in python? Explain keyword arguments and default parameters with suitable example. [05] CO1 L2

► A function is a mini-program within a program.
► Purpose: group code that gets executed multiple times
► Avoid duplication of code
► Deduplication makes your programs shorter, easier to read, and easier to update
► Pass values to functions (parameters)
► A parameter is a variable that an argument is stored in when a function is called
► hello("Alice") : variable name is automatically set to "Alice".
► Value stored in parameter is destroyed when function returns
► The scope is local def hello(name): print("Hello " + name) hello("Alice") hello("Bob") print(name) Output: Hello Alice Hello Bob NameError: name 'name' is not defined
► Some arguments are identified by the position.

► Eg. random.randint(1,10) signifies start, stop
► keyword arguments are identified by the keyword put before them in the function call
► print() function has two optional parameters – end and sep
► end=‘’ : useful in getting rid of the new line

```
>>>print('hello', end='')
>>> print('cats','dogs','mice')
>>> print('cats','dogs','mice',sep=',')
```

(b) Write a python program to create a function called collatz() which reads as parameter named number. If the number is even, it should print and return number//2 and if the number is odd then it should print and return 3*number+1. The function should keep calling on that number until the function returns a value 1. [05] CO1 L3

```
#Write a python program to create a function called collatz() which reads as parameter named number.
#If the number is even, it should print and return number//2 and if the number is odd then it should print and return 3*number+1
#The function should keep calling on that number until the function returns a value 1.
def collatz(number):
    if number%2==0:
        return number//2
    else:
        return 3*number+1
while True:
    num=int(input('Enter a number'))
    x=collatz(num)
    print(x)
    if x==1:
        break
    else:
        continue
```

```
Enter a number 4
2
Enter a number 6
3
Enter a number 7
22
Enter a number 8
4
Enter a number 1
4
Enter a number 2
1
```

5 (a) write a list comprehension to display the leap years from the years 1890 to 2021. [05] CO2 L3

```
#write a list comprehension to display the leap years from the years 1890 to 2021.
#newlist = [expression for item in iterable if condition == True]
leap=[x for x in range(1890,2022) if (x%4==0) and (x%100!=0) or (x%400==0)]
print(leap)
```

```
[1892, 1896, 1904, 1908, 1912, 1916, 1920, 1924, 1928, 1932, 1936, 1940, 1944, 1948, 1952, 1956, 1960, 1964,
1968, 1972, 1976, 1980, 1984, 1988, 1992, 1996, 2000, 2004, 2008, 2012, 2016, 2020]
```

(b) For a given list num=[45,22,14,65,97,72], write a python program to replace all the integers divisible by 3 with “ppp” and all integers divisible by 5 with “qqq” and replace all the integers divisible by both 3 and 5 with “pppqqq” and display the output. [05] CO2 L3

```python
#For a given list num=[45,22,14,65,97,72], write a python program to replace all the
#integers divisible by 3 with "ppp" and all integers divisible by 5 with "qqq" and
#replace all the integers divisible by both 3 and 5 with "pppqqq" and display the output
num=[]
n=[45,22,14,65,97,72]
for i in n:
    if (i%3==0) and (i%5 == 0):
        num.insert(i,'pppqqq')
    elif i%5==0:
        num.insert(i,'qqq')
    elif i%3==0:
        num.insert(i,'ppp')
    else:
        num.append(i)
print(num)
```

```
['pppqqq', 22, 14, 'qqq', 97, 'ppp']
```

6 (a) What is list? Explain append(), insert(), and remove() methods with examples? Explain the concept of slicing and indexing with proper examples. [06] CO2 L2

► list is a value that contains multiple values in an ordered sequence
► list value - value that can be stored in a variable or passed to a function like any other value
► a list begins with an opening square bracket and ends with a closing square bracket, []
► Items are separated with commas (comma delimited)
► [] is an empty list – contains no values
>>> [1,2,3] [1,2,3]
>>> ['python','dbms','os'] ['python','dbms','os']
>>> ['Max',23,83.5,True] ['Max',23,83.5,True]
>>> student=['Max',23,83.5,True]
>>> student ['Max',23,83.5,True]
>>> type(student)
>>>[]
► Lists can also contain other list values.
► If only one value is used for the index, full list value is printed.
► If two index values are used, the second indicates the value in to access inside the list value.
► Negative Indices :
► -1 : indicates last value in a list
► -2 : indicates second last value and so on.
>>> student=['Max',23,[75,80,62]]
>>> student[2] [75,80,62]
>>> marks=[52,82,92,98]
>>> marks[-1] 98
► append() : adds argument to the end of the list.
► insert() : can insert a value at any index in a list. ** notice that the return value of append() and insert() is None.
► The list is modified in place
► Methods belong to certain data types.
► append() and insert() does not work with other data types – generates an Attribute Error
>>> subjects = ['dbms','os','python']
>>> subjects.append('Java')
>>> subjects.insert(1, 'aca')
>>> subjects [‘dbms’,’aca’,’os’,’python’,’Java’]

>>>x=5
>>>x.append(6) AttributeError
>>> x.insert(1,7) AttributeError
► remove() – removes the occurrence of the value from the list
► If there are multiple instances, the first instance is removed.
► If it is not present, ValueError is generated.
>>> subjects= ['dbms', 'aca', 'os', 'python', 'Java', ‘os’]
>>> subjects.remove('dbms')
>>> subjects ['aca', 'os', 'python', 'Java',’os’]
>>> subjects.remove('os')
>>> subjects ['aca', 'python', 'Java',’os’]

(b) Write a python program to take a sentence/text and sort the words according to their length in descending order using different levels of sorting using list of tuples. [04] CO2 L3

```
sent='The world is beautiful'
x=list()
words=sent.split(' ')
for word in words:
    x.append((len(word),word))
x.sort(reverse=True)
print(x)
```

```
[(9, 'beautiful'), (5, 'world'), (3, 'The'), (2, 'is')]
```

7 Write output for the following programs. [2 * 5 = 10 Marks] [10] CO1, CO2 L3

(a)
```
str1="CMR_IT_CSE"
str2=""
for s in str1:
    if s=="_":
        continue
    str2+=s
print(str2)
```

(b)
```
str=0
for i in range(10,1,-1):
    str-=i
print(str)
```

(c)
```
a=[1,2,3,4,5,6,7,8,9]
a[:2]=10,20,30,40,50
print(a)
```

(d)
```
a,b=1,0
a=a and b
b=a and b
a=a and b
print(a)
```

(e)
```
for i in range(10):
    if i==5:
        break
    else:
        print(i,end=' ')
```

```
str1="CMR_IT_CSE"
str2=""
for s in str1:
    if s=="_":
        continue
    str2+=s
print(str2)
```

```
CMRITCSE
```

```
str=0
for i in range(10,1,-1):
    str-=i
print(str)
```

```
-54
```

```
a,b=1,0
a=a and b
b=a and b
a=a and b
print(a)
```

```
0
```

```
for i in range(10):
    if i==5:
        break
    else:
        print(i,end=' ')
```

```
0 1 2 3 4
```

```
a=[1,2,3,4,5,6,7,8,9]
a[:2]=10,20,30,40,50
print(a)
```

```
[10, 20, 30, 40, 50, 3, 4, 5, 6, 7, 8, 9]
```