USN	
-----	--



	Internal Assessment Test	1 - Nov 20)21				
Sub:	Application Development Using Python	Sub Code:	18CS55	Branch:	ISE	~~~	
	Answer any FIVE FULL Questions					CO	RBT
1a)	Explain the use of following functions/methods with co Use-0.5 Code-0.5 i) input() input() is used to read the data from the console and type. Example X =input('Enter x') #10 Y =input('Enter y') #20 print('sum = ',X+Y) ii) type() To know the type of the variable. Ex: type(x) <class iii) format() The format () method formats the specified value(s) placeholder. txt1 = ''My name is {fname}, I'm {age}''.format(fnat iv) remove() The remove() method is passed the value to list it is called on.Ex: >>> spam = ['cat', 'bat', 'rat', 'elephant'] >>> spam.remove('bat') >>> spam ['cat', 'rat', 'elephant']</class 	l provide t s 'int'> and insert me = ''Joh	he data in th t them inside n'', age = 36	e the string's	KS [4]	COI	L3
1b)	Identify the blocks in this code: if (i == 10): block 1 # First if statement if (i < 15): block 2 print("i is smaller than 15") # Nested - if statement # Will only be executed if statement above # it is true if (i < 12): block 3 print("i is smaller than 12 too") else: block 4 print("i is greater than 15") 4 blocks: If 3 Else 1	(4	*0.5)		[2]	CO1	L2
1c)	Explain the purpose of following keyword with code a 1. def We can also define our own functions that acc		(1*4 ents	4)	[4]	CO1 &CO 2	L3

Internal Assessment Test 1 – Nov 2021

```
• def hello(name):
                             print('Hello ' + name)
                        0

• hello('Alice')

                          hello('Bob')
                    2. None
                        >>> spam = print('Hello!')
                        Hello!
                        >>> None == spam
                        True
                 In Python there is a value called None, which represents the absence of a value.
             None is the only value of the None Type data type.
                     3. not in
                  We can determine whether a value is or isn't in a list with the in and not in
                  operators.
                  in and not in are used in expressions and connect two values: a value to look for in
                  a list and the listwhere it may be found and these expressions will evaluate to a
                  Boolean value
                     >>> 'howdy' in ['hello', 'hi', 'howdy', 'heyas']
                     True
                     >>> spam = ['hello', 'hi', 'howdy', 'heyas']
>>> 'cat' in spam
                     False
                     >>> 'howdy' not in spam
                     False
                     >>> 'cat' not in spam
                     True
                  4. del The del statement will delete values at an index in a list.
                  All of the values in the list after the deleted value will be moved
                  up one index.Ex:
                  >>> spam = ['cat', 'bat', 'rat', 'elephant']
                  >>> del spam[2]
                  >>> spam ['cat', 'bat', 'elephant']
                  >>> del spam[2]
2a)
         What is the difference between break and continue statement? Explain with programs(2+2)
                                                                                                         [4] CO1
                                                                                                                    L2
                                                        continue
          break
                                                        1.Like
                                                                 break
                                                                          statements,
                                                                                         continue
          1. There is a shortcut to getting the
                                                        statements are used inside loops
          program execution to break out of a
                                                        2. When the program execution
          while loop's clause early
                                                        reaches a continue statement, the
          2.If the execution reaches a break
                                                        program execution immediately jumps
          statement, it immediately exits the while
                                                        back to the start of the loop and
          loop's clause.
                                                        reevaluates the loop's condition.
```

0 0	<pre>ile True: print('Please type your name.') name = input() if name == 'your name': break int('Thank you!')</pre>	<pre>while True: print('Who are you?') name = input()</pre>			
examp Expla	ble code snippets. nation -3	al scope of the variables in Python with (3*2=6)	[6]	CO1	L2
Exam	<pre>ple-3 Parameters and variables that are a exist in that function's localscope. Variables that are assigned outside a A variable that exists in a local scope that exists in the globalscope is calle A variable must be one or the other; When a scope is destroyed, all the va forgotten. There is only one global scope, ar begins. When your programtermin its variables are forgotten. A local scope is created whenever a f this function exist withinthe local sco scope is destroyed, and these variab Scopes matter for several reasons: 1. Code in the global scope cannot 2. However, a local scope cannot 2. However, a local scope cannot 3. Code in a function's local scope o 4. We can use the same name for c scopes. That is, there can bea loo variable also named spam. def spam(): eggs = 'spam local' print(eggs)</pre>	it cannot be both local and global. alues stored in the scope's variables are and it is created when your program aates, the global scope is destroyed, and all function is called. Any variables assigned in ope. When the function returns, the local ales are forgotten. use any local variables. cannot use variables in any other local scope. different variables if they are in different cal variable named spam and a global local' local' local' local i is in a local scope when spam() is called. ts in a local scope when bacon() is called.	<u>e</u> .		

 A function is like a mini-program within a program. Get hollo(): print("heady(1)) print("heady(1)) print("heady(1)) print("heady(1)) print("heady(1)) print("heady(1)) print("heady(1)) blac() > The first line is a def statement (), which defines a function named hello(). > The code in the block that follows the def statement () is the body of the function. This code is executed when the function calls. > The hello() lines after the function () are function () are function. The hello() lines after the function () are function () are number of a value. > Each data type has its own set of methods. The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. > List_append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Output:: Thrue lipput::25 Output:: False Program: def fib(n): c=0 n=1 h=1 fin=0 or n=1: print(") redexquot;) else: 	3(a)	Explain the difference between function and methods with examples(2+2) Explanation and Example (2+2)	[4]	CO1	L2
• print("sway!") • print("sway!") print("swa!") print("swa!") print("swa!") print("swa!") print("swa!") prop: print("swa!") <td></td> <td></td> <td></td> <td></td> <td></td>					
hello() > Example: > The first line is a def statement ①, which defines a function named hello(). > The code in the block that follows the def statement ② is the body of the function. This code is executedwhen the function is called, not when the function is first defined. > The hello() lines after the function ③ are function calls. > In code, a function call is just the function's name followed by parentheses, possibly with some number of arguments in between the parentheses. Methods • A method is the same thing as a function, except it is "called on" a value. • Each data type has its own set of methods. • The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. • List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output=2 Input: 13 Output: False Program: def fib(n); c=0 a=1 b=1 if n==0 or n==1: print("True") else: else: b=a a=c if c==n; print("True") else:		<pre> def hello(): print('Howdy!') print('Howdy!!!') </pre>			
 The first line is a def statement ①, which defines a function named hello(). The code in the block that follows the def statement ③ is the body of the function. This code is executedwhen the function is called, not when the function is first defined. The hello() lines after the function ④ are function calls. In code, a function call is just the function's name followed by parentheses, possibly with some number of arguments in between the parentheses. Methods A method is the same thing as a function, except it is "called on" a value. Each data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Output=2 Input: 13 Output=2 Input: 13 Output: False Program: def fib(n): c=0 n=1 b=1 if n=0 or n==1: print("True") else: while e<n: c=a+b b=a a: if c==n: print("True") else: 		hello() hello()			
 The code in the block that follows the def statement is is the body of the function. This code is executedwhen the function is called, not when the function is first defined. The hello() lines after the function is are function calls. In code, a function call is just the function's name followed by parentheses, possibly with some number of arguments in between the parentheses. Methods A method is the same thing as a function, except it is "called on" a value. Each data type has its own set of methods. The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2–6) [6] COI L3 Program code=4 Output=2 Input: 13 Output: True Input::25 Output: False Program: def fib(n): c=0 a=1 b=1 if n==0 or n=1: print(")rue") else: while c&l(n: c=a+b) b=a a=c if c=m: print("True") else: 		Example:			
function. This code is executedwhen the function is called, not when the function is first defined. > The hello() lines after the function () are function calls. > In code, a function call is just the function's name followed by parentheses, possibly with some number of arguments in between the parentheses. Nethods • A method is the same thing as a function, except it is "called on" a value. • Each data type has its own set of methods. • The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. • List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output: 2 Input: 13 Output: 2 Input: 13 Input: 13 Output: False Program: def fib(n(): c.c0 a.1 b-1 a.1 b-1 a.1 b-1 a.2 a.2<					
function is first defined. > The hello() lines after the function I are function calls. > In code, a function call is just the function's name followed by parentheses, possibly with some number of arguments in between the parentheses. Methods • A method is the same thing as a function, except it is "called on" a value. • Each data type has its own set of methods. • The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. • List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output=2 Input: 13 [6] CO1 L3 Output: False Program: def fib(n): c=0 [6] Co1 L3 ef ib(n): c=0 a=1 b=1 if n==0 or n==1: print(") rue") else: else: while c&l(n:: c=+b) a b=a a=c if c==n: print("True") else: else: a a a					
 The hello() lines after the function are function calls. In code, a function call is just the function's name followed by parentheses, possibly with some number of arguments in between the parentheses. Methods A method is the same thing as a function, except it is "called on" a value. Each data type has its own set of methods. The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output: True Input: 13 Output: False Program: def fib(n): c=0 a=1 b=1 if n==0 or n==1: print("True") else: while c<n: c=a+b b=a ac-c if c==n; print("True") else: 					
 In code, a function call is just the function's name followed by parentheses, possibly with some number of arguments in between the parentheses. Methods A method is the same thing as a function, except it is "called on" a value. Each data type has its own set of methods. The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output=2 Input: 13 Output: True Input: 25 Output: False Program: def fib(n): c=0 a=1 b=1 if n==0 or n==1: print("True")) else: while c<n: c=a+b b=a a=c if c==n: print("True")) else: 		-			
possibly with some number of arguments in between the parentheses. Methods • A method is the same thing as a function, except it is "called on" a value. • Each data type has its own set of methods. • The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. • List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output=2 Input: 13 [6] CO1 L3 Output: True Input: 25 Output: False Forgram: def fib(n): c=0 [6] Co1 L3 a=1 b=1 if n==0 or n==1: print("True") else: while c <n: a="c" b="a" c="=n:" d="=n:" else:="" else:<<="" if="" print("true")="" print("true")="" td=""><td></td><td>_</td><td></td><td></td><td></td></n:>		_			
Methods • A method is the same thing as a function, except it is "called on" a value. • Each data type has its own set of methods. • The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. • List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output: 1 Input: 13 Input: 13 Input: 13 Input: 25 Input: 25 Output: False Frogram: def fib(n): c=0 a=1 a=1 b=1 if n==0 or n==1: print("True") else: while c <n: a="c" b="1" c="1<</td" else:="" if="" print("true")=""><td></td><td></td><td></td><td></td><td></td></n:>					
 A method is the same thing as a function, except it is "called on" a value. Each data type has its own set of methods. The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output=2 Input: 13 Output=2 Input: 13 Output: False Program: def fib(n): c=0 a=1 b=1 if n==0 or n==1: print("True") else: while c<n: c=a+b b=a a=c if c==n: print("True") else: 					
 Each data type has its own set of methods. The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output=2 Input: 13 Output=2 Input: 13 Output: True Input:25 Output: False Program: def fib(n): c=0 a=1 b=1 if n==0 or n==1: print("True") else: while c<n: c=a+b b=a a=c if c==n: print("True") else: 					
 The list data type, has several useful methods for finding, adding, removing, andotherwise manipulating values in a list. List.append() 3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output: 13 Output: True Input: 13 Output: False Program: def fib(n): c=0 a=1 b=1 if n==0 or n==1: print("True") else: while c<n: c=a+b b=a a=c if c==n: print("True") else: 					
removing, andotherwise manipulating values in a list. List.append() (3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output=2 Input: 13 Output: True Input: 25 Output: False Program: def fib(n): c=0 a=1 b=1 if n==0 or n==1: print("True") else: while c <n: a="c" b="a" c="=n:" else:<="" if="" print("true")="" td=""><td></td><td></td><td></td><td></td><td></td></n:>					
3(b) Write a Python Program to check if a given number is Fibonacci number or not? (4+2=6) [6] CO1 L3 Program code=4 Output=2 Input: 13 [6] CO1 L3 Output: True Input: 25 Output: False [6] C01 L3 Program: def fib(n):					
Program code=4 Output=2 Input: 13 Output: True Input:25 Output: False Program: def fib(n): c=0 a=1 b=1 if n==0 or n==1: print("True") else: while c <n: a="c" b="a" c="=n:" else:<="" if="" print("true")="" td=""><td></td><td>• List.append()</td><td></td><td></td><td></td></n:>		• List.append()			
Program code=4 Output=2 Input: 13 Output: True Input:25 Output: False Program: def fib(n): c=0 a=1 b=1 if n==0 or n==1: print("True") else: while c <n: a="c" b="a" c="=n:" else:<="" if="" print("true")="" td=""><td></td><td></td><td></td><td></td><td></td></n:>					
Output=2Input: 13Output:TrueInput:25Output:FalseProgram:def fib(n): $c=0$ $a=1$ $b=1$ if $n==0$ or $n==1$:print("True")else:while c<n: $c=a+b$ $b=a$ $a=c$ if $c==n$:print("True")else:	3(b)		[6]	CO1	L3
Input: 13 Output: True Input: 25 Output: False Program: def fib(n): c=0 a=1 b=1 if $n=0$ or $n==1$: print("True") else: while c <n: c=a+b b=a a=c if $c==n$: print("True") else:</n: 		8			
True Input:25 Output: False Program: def fib(n): c=0 a=1 b=1 if $n=0$ or $n==1$: print("True") else: while c <n: c=a+b b=a a=c if $c==n$: print("True") else: d=1</n: 					
Input:25 Output: False Program: def fib(n): c=0 a=1 b=1 if n==0 or n==1: print("True") else: while c <n: c=a+b b=a a=c if c==n: print("True") else:</n: 					
Output: False Program: def fib(n): c=0 a=1 b=1 if n==0 or n==1: print("True") else: while c <n: a="c" b="a" c="=n:" else:<="" if="" print("true")="" td=""><td></td><td></td><td></td><td></td><td></td></n:>					
FalseProgram: def fib(n): $c=0$ $a=1$ b=1if n==0 or n==1: print("True") else: while c<n: $c=a+b$ b=a $a=c$ if $c==n$: print("True") else:		*			
def fib(n): c=0 a=1 b=1 if n==0 or n==1: print("True") else: while c <n: a="c" b="a" c="=n:" else:<="" if="" print("true")="" td=""><td></td><td></td><td></td><td></td><td></td></n:>					
c=0 $a=1$ $b=1$ if n==0 or n==1: print("True") else: while c<n: $c=a+b$ $b=a$ $a=c$ if c==n: print("True") else:		Program:			
a=1 b=1 if n==0 or n==1: print("True") else: while c <n: c=a+b b=a a=c if c==n: print("True") else:</n: 					
b=1 if n==0 or n==1: print("True") else: while c <n: c=a+b b=a a=c if c==n: print("True") else:</n: 					
if $n=0$ or $n=1$: print("True") else: while c <n: c=a+b b=a a=c if $c==n$: print("True") else:</n: 					
else: while c <n: c=a+b b=a a=c if c==n: print("True") else:</n: 					
while c <n: c=a+b b=a a=c if c==n: print("True") else:</n: 					
c=a+b b=a a=c if c==n: print("True") else:					
b=a a=c if c==n: print("True") else:					
if c==n: print("True") else:					
print("True") else:					
else:					
print("raise")		print("False")			

	n=int(input()) fib(n)			
5 (a)	110(11)Predict the output and justify the answer(8*0.5) $1. 22/8 = 2$ $2. 7.7//7=1$ $3. (200-70)*10/5 = 260$ $4. not not not False=True$ $55%-6 = -1$ $6. (10<=16)$ and (not True) and (2==2)=False $7. 3**5+(2*10)/5-6 = 241$ $8. (False and True) or True =True$	[04]	CO1	L2
(b)	Write a program to generate the random numbers between 1 to 100. if the generated number is odd, add them in the List named ODD_LIST and if it is even number, add them in the list named EVEN_LIST. If the size of the both the lists become 15, stop the generation of numbers and display both the lists. Program code:5 Output:1		CO2	L3
	even_count, $odd_count = 0, 0$			
	even_list = []			
	odd_list = []			
	n = int(input("Enter the lower bound of range: "))			
	m = int(input("Enter the upper bound of range: "))			
	for i in range(n,m+1):			
	if i % 2 == 0:			
	even_count += 1			
	even_list.append(i)			
	else:			
	odd_count += 1			
	odd_list.append(i)			
	print("Total even numbers in the range $\{0\}$ to $\{1\}$ is $\{2\}$ ".format(n, m, even_count), "and numbers are", even_list)			
	print("Total odd numbers in the range {0} to {1} is {2}".format(n,m,odd_count), "and numbers are", odd_list)			
6 (a)	Explain with examples the way how indexing and slicing can be done in Lists (2+2) Explanation:2 Example:2 Indexing:	[4]	CO2	L2
	The integer inside the square brackets that follows the list is called an index. The first value in the list is at index 0, the second value is at index 1, the third value is at index 2, and so on. spam=['cat', 'bat', 'rat', 'elephant'] >>>spam[0] cat			
	Slicing:			
	Index is used to get a single value from a list, but a slice can get several values			
	from alist, in the form of a new list. A slice is typed between square brackets, like an index, but it has two integers separated by a colon.			
	Ex l: >>> spam = ['cat', 'bat', 'rat', 'elephant']			

			1		
	>>> spam[0:4] ['cat', 'bat', 'rat', 'elephant']				
	>>> spam[1:3] ['bat', 'rat']				
	>>> spam[0:-1] ['cat', 'bat', 'rat']				
6(b)	What are exceptions in Python? Handling Explanation:4 Example:2 <u>Exception Handling</u>	? How it can be handled in Python? Exception (4+2)	[6]	CO1	L2
	If we don't want to crash th	ne program due to errors instead we want the			
		andlethem, and then continue to run.			
	For example,				
	Program	<u>Output</u>			
	 the line number givenin the in spam() is causing an error Errors can be handled with The code that could potent execution moves to thestar We can put the previous diagonal 				
	•	ogram Output			
	<pre>def spam(divideBy): try: return 42 / divideBy except ZeroDivisionError: print('Error: Invalid argument. print(spam(2)) print(spam(12)) print(spam(0)) print(spam(1))</pre>	21.0 3.5 Error: Invalid argument.			
	any errors that occur in function cal	is in a try block will also be caught.			