UNIVERSITY OF LONDON

GOLDSMITHS COLLEGE

Department of Computing

B. Sc. Examination 2019

IS50001C Foundations of Programming

Duration: 2 hours 15 minutes

Date and time:

This paper is in two parts: part A and part B. Part A carries 40 marks, and each question from part B carries 30 marks. ***You should answer ALL questions from part A and TWO questions from part B.*** *If you attempt all 3 questions in part B, only the* ***first two*** *that you attempt will be marked. The marks for each part of a question are indicated at the end of the part in [.] brackets.*

There are 100 marks available on this paper.

You are not allowed to use any electronic device (such as mobile telephones, laptops, calculators, tablets) during the exam.

THIS PAPER MUST NOT BE REMOVED
FROM THE EXAMINATION ROOM

Part A

You should attempt all of these questions

Each multiple-choice question has one (and only one) correct answer.
For each question, write your choice on your *answer* book.

(1) What does execution of the following lines of Python code generate on the screen?

```
a = 2
b = 1
print('a-b =', str(a-b))
```

a. A run-time error
b. `a-b = 1`
c. `a-b = -1`
d. Nothing on the screen; the local printer will print "`a-b = str(a-b)`"

[2]

(2) What is the final value of variable `y` after executing the following code?

```
y = int("0")
x = 1 - y
y = x
```

a. 0
b. 1
c. -1
d. None of the above.

[2]

(3) What is produced on the screen when the following 2 lines of code are executed?

```
t = "13"
print(len(t))
```

a. An error message.
b. 1
c. 2
d. 13

[2]

(4) What type will variable `v` be after the following line is executed?

```
v = input("Please enter a value:")
```

a. Float
b. Integer
c. String
d. None of the above

[2]

(5) What will be printed on the screen when the following Python code is executed?

```
x = "1"
print(float(x))
```

a. The string "`float(1)`"
b. `1.0`
c. `1`
d. The string "`float(x)`"

[2]

(6) What is the value of variable `c` after executing the following code?

```
(c,d) = (0,1)
d = d-c
```

a. 0
b. -1
c. 1
d. None of the above

[2]

(7) Which message will the statements below produce on the screen?

```
var1 = abs(-5)
var2 = 0
if var1 > var2:
   print("Value ", var1, 'is the largest.')
else:
   print("Value ", var2, 'is the largest.')
```

a. `Value 5 is the largest.`
b. `Value -5 is the largest.`
c. `Value 0 is the largest.`
d. An error message.

[2]

(8) What does the following snippet of code print on the screen?

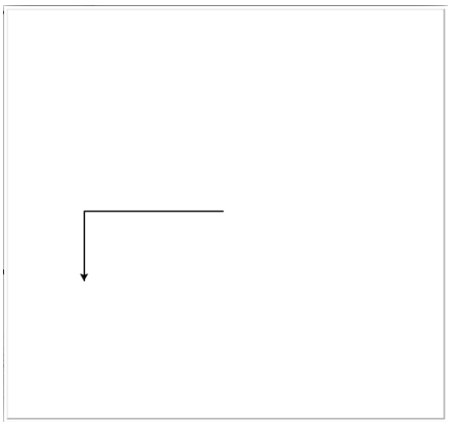
```
import random
print(random.randrange(1,50,2))
```

a. All the odd numbers comprised between 1 and 50, in random order
b. A random float between 1 and 50 (included)
c. A random even number between 2 and 48 (included)
d. A random odd number between 1 and 49 (included)

[2]

(9) In which order should these Python statements be executed for the turtle to draw the figure plotted in the graphic window (canvas) shown on the right?

```
1  jo = turtle.Turtle()
2  jo.forward(75)
3  import turtle
4  wn = turtle.Screen()
5  jo.forward(150)
6  jo.right(180)
7  jo.left(90)
```



a. 3, 4, 1, 5, 6, 7, 2
b. 3, 4, 1, 6, 2, 7, 5
c. 3, 4, 1, 6, 5, 7, 2
d. None of the above

[2]

(10) What do the following lines of Python code produce as output?

```
n = 2
print(4%3 > 1 or n > 0)
```

a. An error message
b. The string `"4%3 > 1 or n > 0"`
c. `False`
d. `True`

[2]

---

(11) How many times does the following ***for*** loop print "Hello" on the screen?

```
for i in [1,4,1]:
    print("Hello")
```

a. 2 times
b. 3 times
c. 4 times
d. None of the above

[4]

(12) What does execution of the following ***for*** loop produce on the screen?

```
n = 2
for j in range(n+n, n, -1):
    print(j)
```

a. The numbers 4, 3 (printed on separate lines)
b. The numbers 4, 3, 2 (printed on separate lines)
c. The numbers 3, 2, 1 (printed on separate lines)
d. An error message

[4]

(13) What does the following Python program produce as output?

```
def myOptions(x,y,z):
    if x > 0:
        return y-z
    elif x < 0:
        return x-z
    else:
        return z-y

x = 1
print(myOptions(0,1,2))
```

a. 1
b. -1
c. -2
d. None of the above

[4]

(14) What does the following code snippet produce on the screen?

```
s = "PA"
s = s + s[:1]
print(s)
```

a. An error message
b. `PA`
c. `PAP`
d. `PAPA`

[4]

(15) What is the output of the following code extract?

```
L = [3, 2, 1]
L = [4, 5, 6] + L
print(L[0])
```

a. 4
b. 3
c. 7
d. None of the above

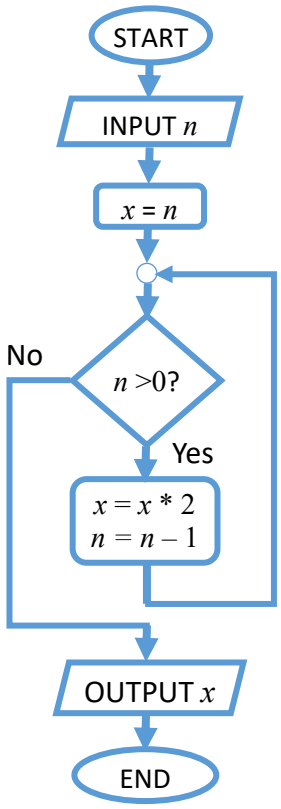
[4]

Part B

You should attempt two of these three questions

QUESTION B1

(a) Consider the algorithm described by the flowchart below, with n integer and $n > 0$:



i. How many times is the body of the loop executed if the value n entered by the user is 1? And if n=2?

[2]

ii. In general, for any given value of n, how many times is the loop body executed? (assume $n > 0$)

[3]

iii. What happens to variables x and n in the body of the loop?

[3]

iv. In view of the above, what value is printed at the end, for any value of n that the user enters?

[4]

v. Write a Python program that implements this algorithm using a ***while*** loop (**Note**: your code should *not* contain any new function definitions).

[8]

(b) What is wrong with the following program? Provide a brief explanation, indicating what should be added to or changed in the code to remove any error(s).

```
1  def multiply(a,b):
2      c = b*a
3
4  result = multiply(4,2)
5  print(c)
```

[10]

QUESTION B2

(a) Consider the following Python program, in which the lines have been numbered for convenience:

```
1  import turtle
2  john = turtle.Turtle()
3  wn = turtle.Screen()
4  john.color('red')
5  john.backward(100)
6  john.forward(100)
7  john.left(90)
```

i. Assuming that a newly created turtle initially faces *East*, draw a diagram on your answer book showing what the graphic window (canvas) will look like after execution of lines 1–7. Write the colour of any lines or segments drawn in it and indicate the turtle's final *location* (specify the *x* and *y* co-ordinates) and *heading* (direction).

[3]

ii. Explain what happens to Python's internal representation of names and objects when line 2 – `john = turtle.Turtle()` – is executed. **HINT**: draw a reference diagram ("state snapshot") to complement your explanation.

[3]

iii. Now suppose that after lines 1–7 above, the following code is executed:

```
8  mary = turtle.Turtle()
9  mary.color('blue')
10 mary = john
11 mary.forward(100)
```

What will the canvas look like at the end? Draw a new diagram, showing the final result of executing lines 1–11 and indicating the colour of all the segments in it.

[4]

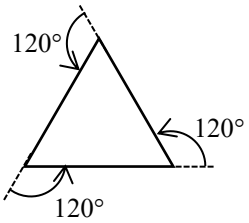
iv. Provide a brief explanation for your answer to point **iii.** above: what happens to variables `john` and `mary` when the assignment statement `mary = john` (line 10) is executed? What happens to the turtle objects these two variables refer to?

To support your explanation, you may draw one (or more) reference diagrams.

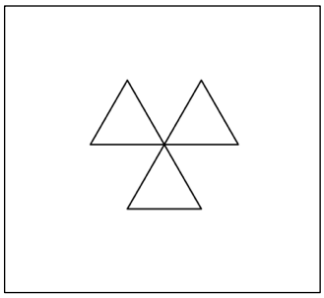
[4]

(b) Consider the user-defined function `drawTriangle(t,sz)` below, which uses the turtle `t` to draw an equilateral triangle having `sz` pixel-long sides (recall that the external angle of an equilateral triangle is 120°, as shown in the figure below):

```
def drawTriangle(t,sz):
    for i in [0,1,2]:
       t.forward(sz)
       t.left(120)
```



Write a new user-defined function, called `Koch(t,sz)`, with 2 input parameters (`t`, `sz`) and such that it calls `drawTriangle()` three times to draw the figure shown below (having each side `sz` pixels long):



[7]

(c) We wish to define a new class in Python for representing a shopping list, i.e., a list of items to be purchased at the grocery. Assume the following class header is given:

```
class shoppingList:
```

i. Write the "`__init__(self)`" class constructor that you'll need, assuming that a newly created shopping list should contain no items.

[3]

ii. Enrich the class with a method called `addItem(name)` which enables adding a new item to the shopping list. For example, to create a list containing the two items `"apples"` and `"tomatoes"`, the user should be able to write:

```
myList = shoppingList()
myList.addItem("apples")
myList.addItem("tomatoes")
```

[6]

QUESTION B3

(a) The following user-defined Python function takes a list `x` of integers as input parameter:

```
def incognito(x):
    s = 0
    for i in x:
      if i == -1:
          s += 1
    return s
```

i. If the argument `x` is the list `[2,1,0]`, what will the value of the loop variable `i` be during the first iteration of the ***for*** loop? And during the last one?

[2]

ii. If the function is passed an empty list as input, i.e., `incognito([])`, how many times will the body of the ***for*** loop be executed?

[2]

iii. If list `x` contains *n* integers, how many times will the body of the ***for*** loop be executed?

[2]

iv. What value is returned by the function if argument `x` is the list `[5,-1,-1]`? And if `x` is the list `[0,2,4,-5,6]`?

[2]

v. What value does the function return in general, for *any* given list `x` of integers?

[2]

(**b**) Write a new user-defined function called `takeAway(x, y)` that takes two strings `x`, `y` and returns the original string `x` after removing (in a *case-sensitive* manner) all characters that also appear in `y` from it.

For example, `takeAway('Rubber', 'bar')` should return `'Rue'`: in fact, `'b'` and `'r'` appear also in `'bar'`, and when these are removed from `'Rubber'`, only `'Rue'` is left. (Note that `'R'` is not removed).

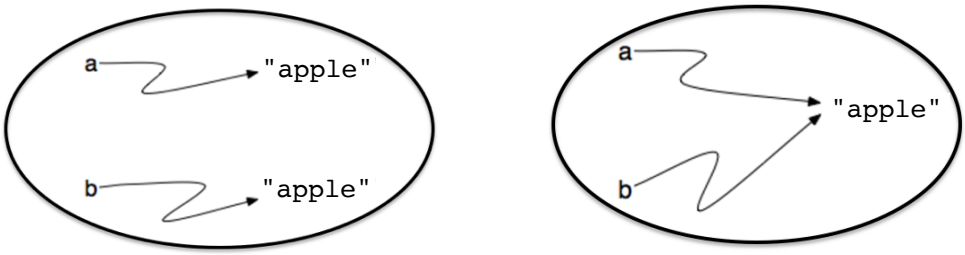
NOTE: your code should *not* use any operators other than "`in`", "`not in`" and "`+`".

[8]

(c) Consider the following two lines of Python code, which have been numbered for convenience:

```
1  a = "apple"
2  b = "apple"
```

The reference diagrams below depict two different ways in which Python's internal memory could represent variables `a` and `b` and the string object(s) they refer to:



L R

On the left (L), two (identical) string objects are created; on the right (R), only one.

i. After executing lines 1–2 above, is there a way to interrogate Python about which of these situations (L or R) its memory is in? If yes, how? If not, why not?

[4]

ii. Will Python's memory actually be as shown in the L or in the R diagram? Why? Briefly motivate your answer.

[3]

iii. Now suppose that after lines 1–2 above, two additional assignments are executed:

```
3  a = ['a','p','p','l','e']
4  b = ['a','p','p','l','e']
```

On your answer book, draw a single reference diagram (state snapshot) illustrating Python's internal state after lines 1-2-3-4 have been executed.

[3]

iv. Provide a *short* explanation in support of your answer to question **iii.** above.

[2]

End of Exam