MA1008 Introduction to Computational Thinking Quiz 1
Answer all the nine questions in the spaces provided
AY 2023/2024, Semester 1, Week 5

Your Name: ______________________________ Group: _____

Solutions

Alternative solutions may be possible for some code. So, please check an answer carefully if it is different from the answer given here.

1. State the functions of the following components in a microprocessor. (3 marks each)

   i. Arithmetic/logic unit — Performs arithmetic and logic operations.

   ii. Address bus — Conveys address information between the other functional units.

   iii. Register array — High speed internal storage for storing frequently accessed data.

2. i. Write a for loop that produces the factorial of an integer `j`. You may assume that `j` is given, and hence do not need to input its value. You also do not need to print the result. (4 marks)

```
f = 1                       # initialise factorial value
for i in range(1, j+1):     # can also have range(2, j+1)
    f = f*i
```

   ii. Using the above for loop, write Python code to evaluate this summation series:

   S = $\sum_{j=0}^{m}(x^{2j+1}/(2j+1)!)$ for given values of x and m. (8 marks)

```
sum = 0
for j in range(m+1):
    f = 1
    for i in range(1, 2*j+2):
        f = f*i
    sum += x**(2*j+1) / f
```

3. i. Given the function f(x) = $3x^2 + 2x - 15\cos(3x) - 10$, write Python statements that determine the value of f(x) given the values of x at x = x1 and x = x2, where x1 and x2 are known values. Name the variables that receive the respective function values as fx1 and fx2. You may assume that the math library has been imported using the statement `import math`. (3 marks)

```
fx1 = 3*x1*x1 + 2*x1 - 15*math.cos(3*x1) - 10
fx2 = 3*x2*x2 + 2*x2 - 15*math.cos(3*x2) - 10
```

ii. If the equation f(x) = 0 has at most one root in the interval between x1 and x2, then a root exists within the interval if the values of f(x1) and f(x2) have different signs, as illustrated in the figure below. If f(x1) = 0, then x1 is the root, else if f(x2) = 0, x2 is the root. Given f(x) and the function values fx1 and fx2 at x1 and x2 respectively, as in Part i above, write Python statements to (a) print "Root at x1" if f(x1) = 0 or (b) print "Root at x2" if f(x2) = 0 or, if both fails, (c) check if a root exists between x1 and x2, and print "One root" if one does and "No root" otherwise.

(8 marks)

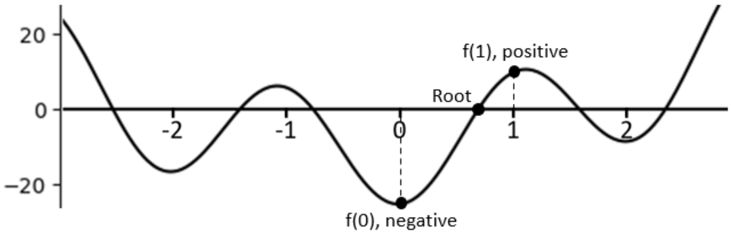

Figure: A root exists between x = 0 and x = 1 because f(0) and f(1) have different signs.

```
eps = 1e-8
if abs(fx1) < eps:
   print("Root at x1")
elif abs(fx2) < eps:
   print("Root at x2")
elif fx1*fx2 < 0:
   print("One root")
else:
   print("No root")
```

If a student doesn't use eps and simply writes
`if fx1 == 0:` or `if fx2 == 0:`
we can forgive them and award the full marks.

The second `elif` may be written as
`elif (fx1<0 and fx2>0) or (fx1>0 and fx2<0)`
though not so clever.

4. Write Python statements that print in descending order the even integers that lie between –500 and 500 and are multiples of 3 but not of 4. If an integer is a multiple of 6, print half of its value instead of the actual value. Print all the values consecutively as integers separated by a space.

(10 marks)

```
for i in range(500, -501, -2):  # descending order, even
   if i%3 == 0 and i%4 !=0:
      if i % 6 == 0:
         print(int(i/2), end = " ")  or  print(i//2, end = " ")
      else:
         print(i, end = " ")
```

The question has not stated between -500 and 500 inclusive. So if a student doesn't include the end values, we can forgive and give full marks.

5. In the following table, place True or False in the blank boxes. (1 mark each)

a	b	c	a or b and c	(a or not b) and c	not a and b and (a or c)
True	True	False	True	False	False
True	False	True	True	True	False
False	True	False	False	False	False
False	False	True	False	True	False

6. Given the value of `n`, the following program generates a diamond shape made of "%" spaced appropriately. Two examples for `n` = 3 and `n` = 4 are given below. Fill in the blanks to complete the program. (12 marks)

```
n = 3   %        n = 4    %
       % %               % %
      %   %             %   %
       % %             %     %
        %               %   %
                         % %
                          %
```

```
print(" "*(n-1) + "%")              # Print first line with one %

for i in range(1, n):               # Print the rest of top part

    print(" "*(n-i-1) + "%" + " "*(2*i-1) + "%")

for i in range(2, n):               # Print bottom part

    print(" "*(i-1) + "%" + " "*(2*(n-i)-1) + "%")

print(" "*(n-1) + "%")              # Print last line with one %
```

Alternate solutions possible for the blanks here.

7. What are printed in the following programs:

i.
```
for i in range(9, 27, 9):
    j = 0
    while j <= i:
        print(i+j//3*3, end = " ")
        j += 8
```
(5 marks)

`9 15 18 24 33`

ii.
```
for i in range (15, -10, -4):
    if i%3 == 0:
        continue
    print(i%3**2, end = " ")
```
(5 marks)

`2 7 8 4`

8. State whether each of the following statements is true or false. If you answer false, state why it is false. Underlined texts are Python code. (3 marks each)

i. `a *= b + c` is the same as `a = a * b + c`. False, right-hand side must be done as one expression first, like `a * (b + c)`.

ii. `a + b = c` is the same as `c = a + b`. False, an assignment statement must only have a variable on the left.

iii. `-a**b**c` is the same as `(-a)**(b**c)`. False. The power operator has a higher priority than the unary minus. Hence it should be `–(a**(b**c))`.

iv. `3*4.2 == 12.6` always returns `True`. False. Floating point number representations are not exact, and hence cannot guarantee that the two sides are exactly equal.

9. i. Given a variable `i` that carries a single digit integer and another integer `n` of any value, <u>without converting `i` or `n` to string</u>, write Python statements that print `True` if `i` exists in `n`, and `False` otherwise. For example, if `i` = 2 and `n` = 16092023, the result should be `True`. If `i` = 4, the result should be `False`. Print `True` only once if there are multiple occurrences of `i` in `n`. You do not need to provide input statements for `i` and `n`. (8 marks)

```
while n > 0:
   digit = n % 10  # get the last digit of n
   if digit == i:
      print(True)
      break        # This break ensures only printing once
   n = n//10       # remove last digit from n
else:   #  This else line could be changed to  if n == 0:
   print(False)
```

ii. Perform the same operation as in Part i but by first converting `i` and `n` to strings. (4 marks)

```
I = str(i)
N = str(n)
print (I in N)
```

<> <> <> THE END <> <> <>