Worksheet 1: Pseudo Code Analysis

The goal of this worksheet is to practice analysing Pseudo Code algorithms using operation counting.

You are required to submit 4 questions from the worksheet. The remainder are for practice. The mandatory questions are: Q2, Q5, Q8, Q9.

Please check the updated lecture note for examples of how to do the analysis.

For the purposes of this worksheet, you can assume that each of the following operations are primitive:

print()
println()
read()

For each question, work out the number of operations for the algorithm and indicate what the Big-Oh estimation of running time would be for that algorithm.

Question 1

Algorithm program()
    Input: none
    Output: none
    x ← 5
    y ← 7
    s ← x + y
    println(x + " plus " + y + " is " + s)

Question 2

Algorithm program()
    Input: none
    Output: none
    print("enter a number: ")
    num ← read()
    print("you entered: ")
    if (num < 10000) print("0")
    if (num < 1000) print("0")
    if (num < 100) print("0")
    if (num < 10) print("0")
    println(num)

Question 3

```
Algorithm program()
    Input: none
    Output: none
    print(“enter a number: “)
    x← read()
    print(“enter a number: “)
    y← read()
    print(“result: “)
    if x > y then println(y + “,” + x)
    else if x < y then println(x + “,” + y)
    else println(x + “,” + y)
```

Question 4

```
Algorithm program()
    Input: none
    Output: none
    i ← 0
    s ← 0
    while i < 100 do
        if i % 10 == 0 then s ← s + i
        i ← i + 1
    println(“result: “ + s)
```

Question 5

```
Algorithm program()
    Input: none
    Output: none
    i ← 20
    while i >= 0 do
        if i < 20 then print(“,”)
        print(i)
        i ← i - 2
```

Question 6

```
Algorithm program()
        Input: none
        Output: none
        Let A be an array containing the string "HAPPY"
        l ← 0
        r ← 4
        while l < r do
                t ← A[l]
                A[l] ← A[r]
                A[r] ← t
                l ← l + 1
                r ← r – 1
        for each value, j, in the range 0 to 4 do
                print(A[j])
```

Question 7

```
Algorithm program()
        Input: none
        Output: none
        num ← 0
        A ← 0
        j ← 0
        while num <> -1 do
                num ← read()
                if num <> -1 then
                        A ← A + num
                        j ← j + 1
        println(j + " numbers entered with result: " + (A / j))
```

Question 8

```
Algorithm program()
        Input: none
        Output: none
        Let A be an array containing { 5, 7, 3, 12, 6, 11, 1, 19, 9, 4 }
        j ← 1
        t ← A[0]
        while j < 10 do
                A[j-1] ← A[j]
                j ← j + 1
        A[j-1] ← t
        for each value, j, in the range 0 to 9 do
                print(A[j] + " ")
```

Question 9

```
Algorithm fn(num, digits)
        Input: num and digits
        Output: output
        output ← ""
        mult = 1
        for each value, j, in the range 1 to digits do
                if num < mult then output ← output + "0"
                mult = mult * 10
        output ← output + num
        return output

Algorithm program()
        Input: None
        Output: None
        println(fn(75, 6))
```

Question 10

```
Algorithm program()
        Input: none
        Output: none
        Let A be an array containing { 5, 7, 3, 12, 6, 11, 1, 19, 9, 4 }
        j ← 0
        while j < 10 do
                m ← j
                for each value, k, in the range j+1 to 9 do
                        if A[m] > A[k] then m ← k
                if m <> j then
                        t ← A[j]
                        A[j] ← A[m]
                        A[m] ← t
                j ← j + 1
        for each value, j, in the range 0 to 9 do
                print(A[j] + " ")
```