Revision Questions for Section 4.2 - Connecting computational thinking and program design

4.2.1 Linear arrays

_____Links: Array | Sequential Search | Binary Search | Bubble Sort | Selection Sort

a. Describe the main characteristics of the **sequential** search and **binary** search algorithms.
b. Describe the main characteristics of the **bubble** sort and **selection** sort algorithms.
c. Use the array of names { “Mary”, “Beth”, “Tom”, “Bill”, “Cath” } to illustrate **bubble sort**.

4.2.2 Collections

_____Links: Collection Interface | Generic Programming | Lists and Sets

a. Name one **collection** class in Java.
b. Outline three standard **operations** (methods) that can be applied to this collection class.

4.2.3 Algorithm for a specific problem

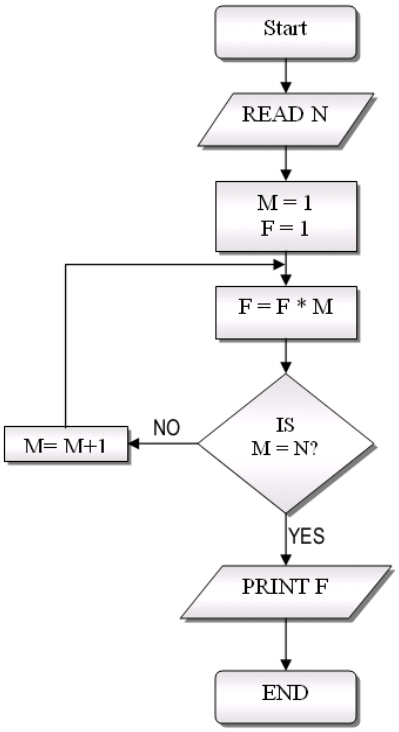
_____Links: Binary vs. Sequential |

a. Compare the advantages and disadvantages of **binary** search versus **sequential** search.
b. Describe **two** possible algorithms for reversing an array and compare their advantages and disadvantages.

4.2.4 Algorithm as flowchart

_____Links: Flowchart

a. State the **purpose** of the algorithm shown in the flowchart below.



b. State the **output** of the flowchart shown for an input of 7..

4.2.5 Algorithm as pseudocode

_____Links: Pseudocode | Algorithm Development

a. Explain the **purpose** of the algorithm shown as pseudocode below.

```
COUNT = 0
TOTAL = 0
loop N from 0 to 999
      if STOCK[N] > 0 then
            COUNT = COUNT + 1
            TOTAL = TOTAL + STOCK[N]
      end if
end loop
if NOT COUNT = 0 then
      M = TOTAL / COUNT
      output M
else
      output "There are no non-zero values"
end if
```

b. State the output of the pseudocode shown if the STOCK array contains {3,7,6,5,10,8}

4.2.6 Pseudocode for an algorithm

_____Links: Pseudocode | FizzBuzz

a. Create pseudocode for an algorithm to **reverse** the contents of an array.
b. Create pseudocode for an algorithm to play the number game ***fizz-buzz*** with the numbers 5 and 7.

4.2.7 Algorithms for a specific problem

_____Links: Rock-Paper-Scissors | Password Strength

a. Suggest an algorithm that will implement the game of ***rock, paper, scissors***.
b. Suggest an algorithm that will measure the strength of a chosen **password** and output either "WEAK", "MEDIUM" or "STRONG"

4.2.8 Efficiency of algorithms

_____Links: Algorithm Efficiency | AND gate

a. Suggest two algorithms that both search for an item in a sorted array, one **efficient** and one **inefficient**.
b. Rewrite this pseudocode for an algorithm that implements an **'AND'** gate more efficiently.

```
IF (A == 0) and (B == 0) THEN
        return 0
ELSE IF (A == 0) and (B == 1) THEN
        return 0
ELSE IF (A == 1) and (B == 0) THEN
        return 0
ELSE IF (A == 1) and (B == 1) THEN
        return 1
END IF
```

4.2.9 Repetition in algorithms

_____Links: Loops | While Loop | While and Do .. While

a. Determine the **number** of times the loops in these algorithm will be performed for an input of 7.

```
input n
count = 1
       while (count <= n)
            output("Count is: " + count)
       count = count + 1
end while
```

b.
```
input n
do
      output n
      n = n - 1
while (n - 2 > 0)
```

c.
```
input n
for i from n to n*10 step 2
      print i mod 2
end for
```