**TEST 1 – F# Programming test (50 marks)**

**BACKGROUND:**

Figure 1a. and Figure 1b. on the next page shows two networks A and B, respectively. These networks are arbitrary trees of:

type NetworkItem =

| Switch of string \* int \* List<NetworkItem>

| Node of string \* int

| Printer of string

Where:

* Switch has a name (string), a number of ports (int) and a list of a subtree of type NetworkItem.
* Node has a user name (string) and an amount of RAM (int).
* Printer has a name (string).

The F# implementation for network A:

let AedgeA =

Switch("edgeA",16,[Node("bob",4); Node("jim",8); Node("alison",4)])

let AofficeB =

Switch("officeB",4,[Node("amy",2); Printer("office-b-printer")])

let AedgeB =

Switch("edgeB",8,[Node("sarah",2); Node("adam",12); Node("maureen",16);

AofficeB])

let networkA =

Switch("core",4,[AedgeA; AedgeB])

And for network B:

let BedgeA =

Switch("edgeA",16,[Node("bob",4); Node("jim",8); Node("alison",4)])

let BofficeB =

Switch("officeB",4,[Node("amy",2); Node("maureen", 10); Printer("office-b-printer")])

let BedgeB =

Switch("edgeB",8,[Node("sarah",2); Node("adam",12); Node("maureen",16); Node("bob", 12); BofficeB])

let networkB =

Switch("core",4,[BedgeA; BedgeB])

A screenshot of a computer

Description automatically generated with low confidence

Figure 1a. Network A

A screenshot of a computer

Description automatically generated with low confidence

Figure 1b. Network B

**PREPARATION:**

* Download the project Fs\_Test\_2021-22 from the CMP-7009A Blackboard
* Program.fs, as part of this project has the code as shown on the previous page.

**SPECIFIC TASKS:**

**Test 1 - Task 1**

Create a function networkCap that takes a network switch as a parameter and returns a tuple of type: int \* int, where:

* The first element of the tuple is the **number of spare ports** in the entire network considering that **each node/printer in the network takes up one port from a switch and two switches connect bi-directionally**. The “core” switch is the root and only connects to the two switches below it.
* The second element of the tuple is the **total amount of RAM** in the entire network.

Important notes:

* The “core” switch has no upwards connections.
* The function should also work when called with an internal network switch of which its port to a switch higher up in the network has to be subtracted as well!

Hint: You could create an additional parameter to specify the switch of the following enum type:

type select =

| root = 0 // Root switch

| inter = 1 // Intermediate switch

| other = 2 // Anything else, e.g. node, printer

Example output (testers):

On networkA: returns (18, 48)

On networkB: returns (16, 70)

On BEdgeB: returns (2, 54)

On AEdgeA: returns (12, 16)

[25 marks]

**Test 1 - Task 2**

Create a function FindDuplicateUser that takes a network switch as a parameter and returns a tuple of type: bool \* List<string>, where:

* The first element of the tuple is a Boolean variable that is true if there are duplicate users and false if there are none.
* The second element is the list of duplicate names.

You should also print the list of duplicate users using a function printList that takes a parameter of type List<string> and then prints each string element.

Example output (testers):

On networkA: returns false and empty list

On networkB: returns true and [ “bob”; “maureen” ]

On BedgeB: returns true and [“maureen”]

[25 marks]

**General hint for both tasks**:

To facilitate operations on each of the two elements (fst, snd) of the return tuples in both tasks during recursive calls, a helper function will be needed. For example for the first task, we need to do addition, so function addTuple a b, with tuples a and b would perform tuple addition like: (a1, a2) + (b2, b2) = (a1+a2, b1+ b2)

**Figure 2. shows a console output of all testers.**

**ADDITIONAL NOTES:** **You can use all lecture materials and examples on blackboard. Although you can use the internet as well for reference material, e.g. F# on msdn, it is advised not to overuse the internet as it is unlikely that you will find any material that directly leads to solution of Tasks 1 & 2.**

**SUBMISSION:**

After you finished, please rename Program.fs to Program\_ID.fs where ID is your student ID and

Also make sure you put your student ID in the header of the Program\_ID.fs file.

If you created and used any .fsx files as drafts, you can submit these as well.

Text

Description automatically generated

Figure 2. Example output of program.

**END OF TEST 1 – PLEASE CONTINUE TO TEST 2 ON THE NEXT PAGE**