FIT1045 Algorithmic Problem Solving – Tutorial 11.

Objectives

The **objectives of this tutorial** are:

- To introduce Big-O notation for simple functions.
- To get familiar with Binary Trees, Binary Search Trees and cost of their operations.
- To be able to find a Hamiltonian cycle in a graph.

Task 1

Part A

Count the number of steps taken by the following programs using the Simple Computation Model given in Lectures. (To check if your count is correct, count the number of steps taken when $n = 1$ and $n = 2$ by each of these programs.)

```
def program1(n):
    count=0
    i=0
    while i<n:
        j=0
        while j<n:
            count+=1
            j+=1
        i+=1
    return count
```

and

```
def program2(n):
    count =0
    i=0
    while i<n:
        j=i+1
        while(j<i+3):
            count+=1
            j+=1
        i+=1
    return count
```

Part B

State the complexity of `program1` and `program2` in Big-O notation.

Task 2

Part A

In groups, draw all binary trees with four nodes. Label the levels of the tree and give its height.

Part B

Revisit the algorithm for finding an item in a Binary Search Tree discussed in Tutorial 10.

For each tree, identify the node(s) that will take the most comparisons to find and the node that will take the least comparisons to find.

Using this information, identify the best of these trees in terms of minimising the maximum search time.

Part C

Using Big-O notation give the best and worst cases of searching in a BST and their running time.

Part D

Which of the trees identified in Task 2A can be heaps? What can we say about the height of a binary tree that is a heap?

Task 3

Part A

Find a Hamiltonian cycle in the graph given in Figure 1. Write down the vertices in the order they appear in the cycle.

Can you find a Hamiltonian cycle in the graph given in Figure 2? Explain.

Part B

Consider the decision problem **HAMILTONIAN CYCLE**.
Input: Graph G
Question: Does G have a Hamiltonian Cycle?

Write an algorithm for deciding if a graph has a Hamiltonian cycle. What is the complexity of your algorithm?

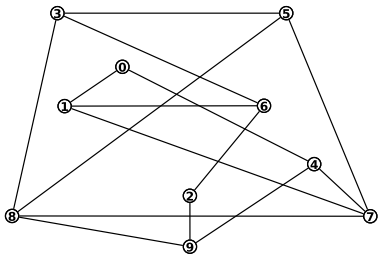

Figure 1: Graph G

Puzzle of the week

There is a magical tap in the Faculty of Information Technology. When the tap is opened an infinite flow of numbers come from it (numbers flow one at a time). The faculty decided to arrange a competition to develop an algorithm such that the algorithm should report the k biggest numbers (k is fixed) came out of the tap at any instance in time. The winner is the person whose algorithm takes the minimum time to execute.

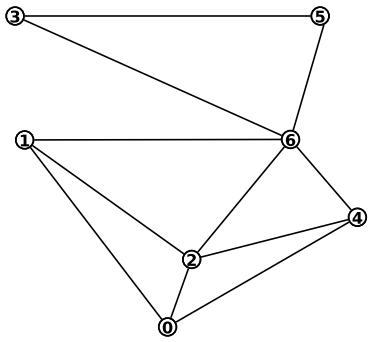

Figure 2: Graph H