Algorithms and Analysis – COSC2123/1285

Assignment 1

(Please check the Assignment Specification for details)

Dr. Yongli Ren

(yongli.ren@rmit.edu.au)

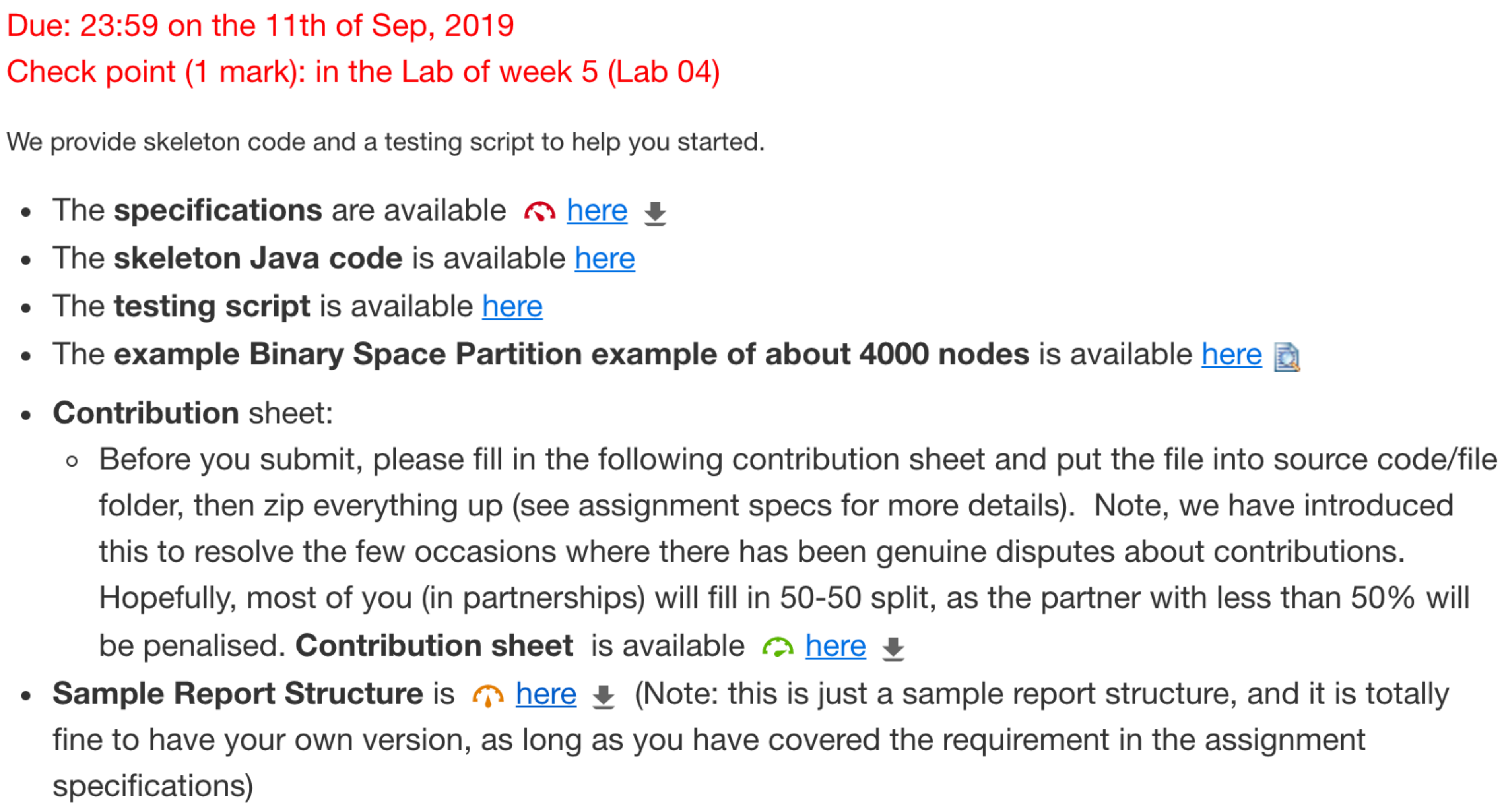
Computer Science & IT

School of Science

RMIT UNIVERSITY

Assignment 1
- Binary Space Partitioning (BSP)

- Pair (Group of 2) Assignment



Due: 23:59 on the 11th of Sep, 2019

Check point (1 mark): in the Lab of week 5 (Lab 04)

We provide skeleton code and a testing script to help you started.

- The **specifications** are available here
- The **skeleton Java code** is available here
- The **testing script** is available here
- The **example Binary Space Partition example of about 4000 nodes** is available here
- **Contribution** sheet:
  - Before you submit, please fill in the following contribution sheet and put the file into source code/file folder, then zip everything up (see assignment specs for more details). Note, we have introduced this to resolve the few occasions where there has been genuine disputes about contributions. Hopefully, most of you (in partnerships) will fill in 50-50 split, as the partner with less than 50% will be penalised. **Contribution sheet** is available here
- **Sample Report Structure** is here (Note: this is just a sample report structure, and it is totally fine to have your own version, as long as you have covered the requirement in the assignment specifications)

Assignment 1
- Binary Space Partitioning (BSP)

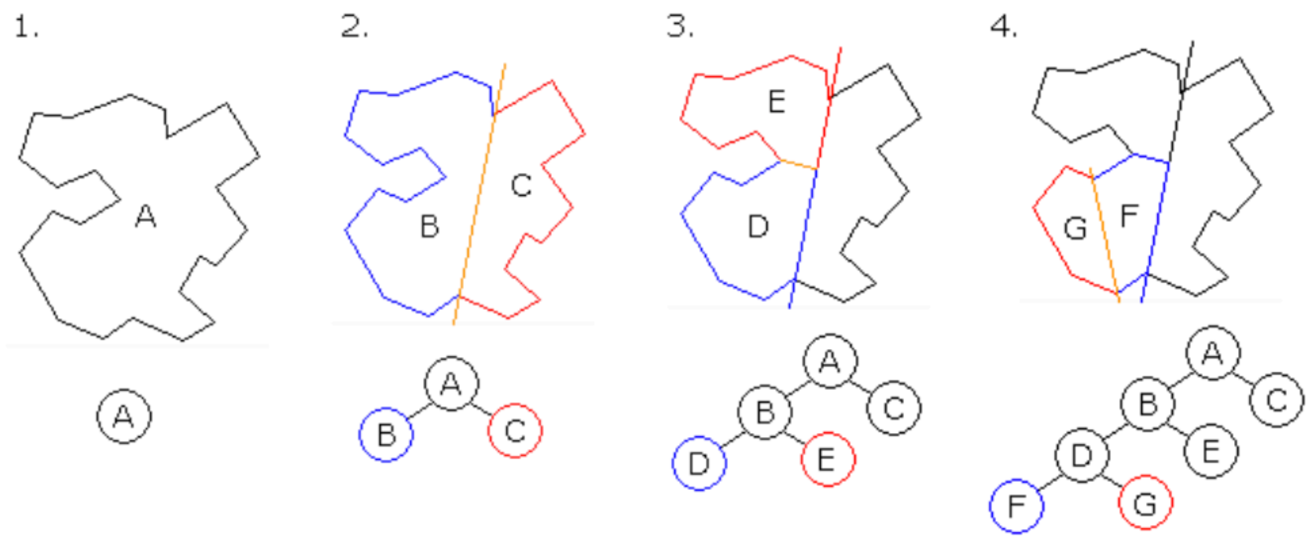


Figure 2: Example of BSP [2]

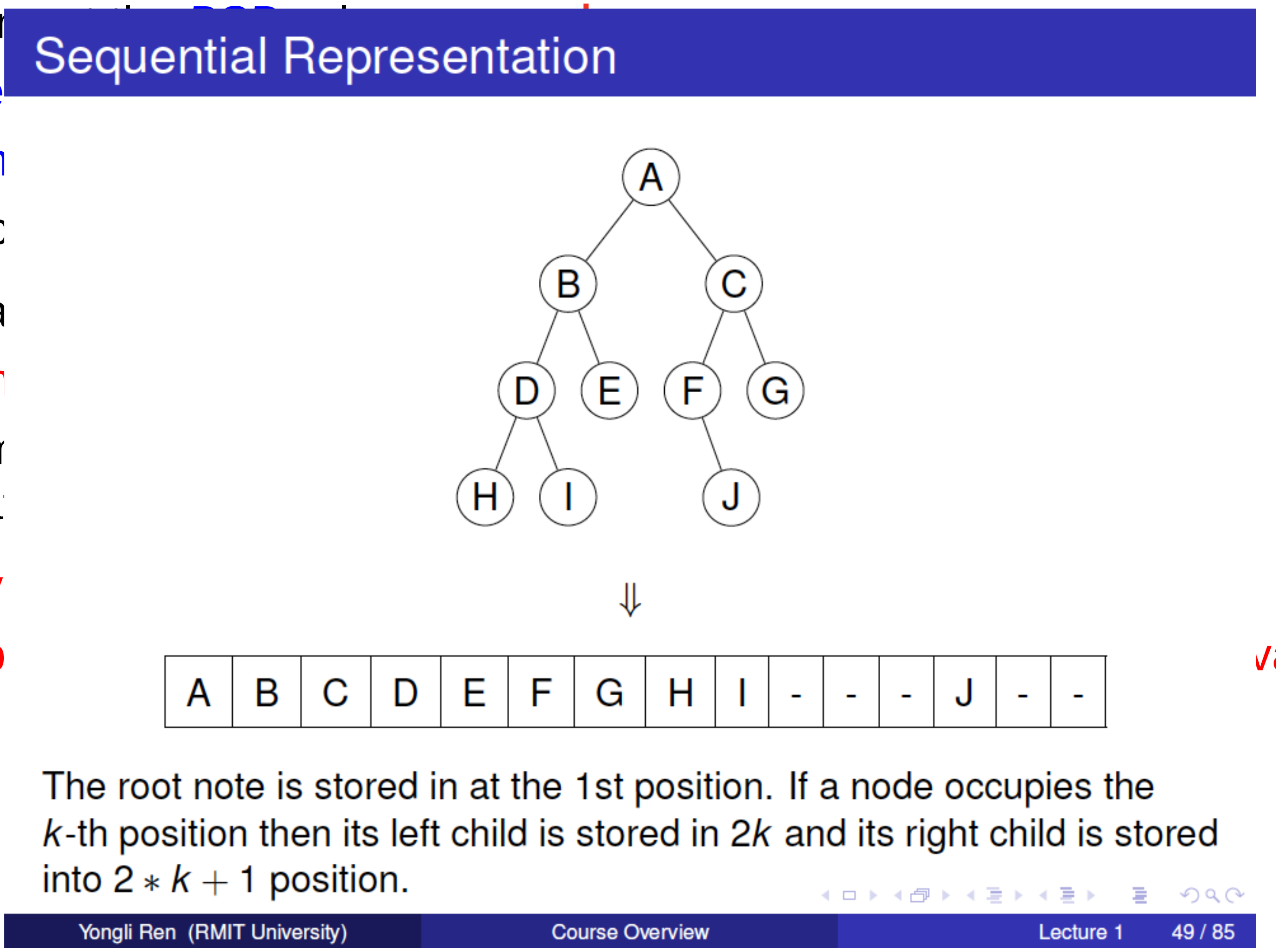
https://www.youtube.com/watch?v=yTRzfKh4Tg0

Task A

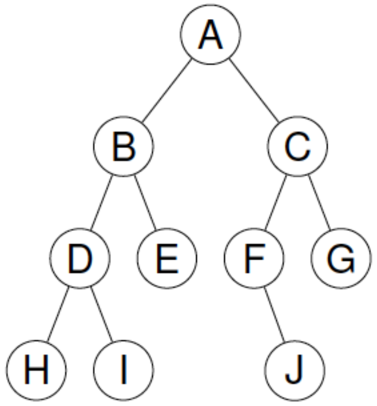
- Implement the Tree Representations and their Operations (5 marks)
  - implement the *BSP* using — Lecture 1/4 and Lab 05
    - Sequential Representation, using a 1D array.
    - Linked Representation, using a linked list (program your own).
  - support the required operations.
- For the above two data structures,
  - you must program your own implementation,
  - You must implement your own nodes and methods to handle the operations.
- Extra java class: Yes
  - But, put it in the format of in-class implementation (within existing java files)

Task A

- Implement the Tree Representations and their Operations (5 marks)
  - implen
    - Se
    - Lin
  - suppo
- For the a
  - you m
  - You m opera
- Extra jav
  - But, p va files)



Sequential Representation



$\Downarrow$

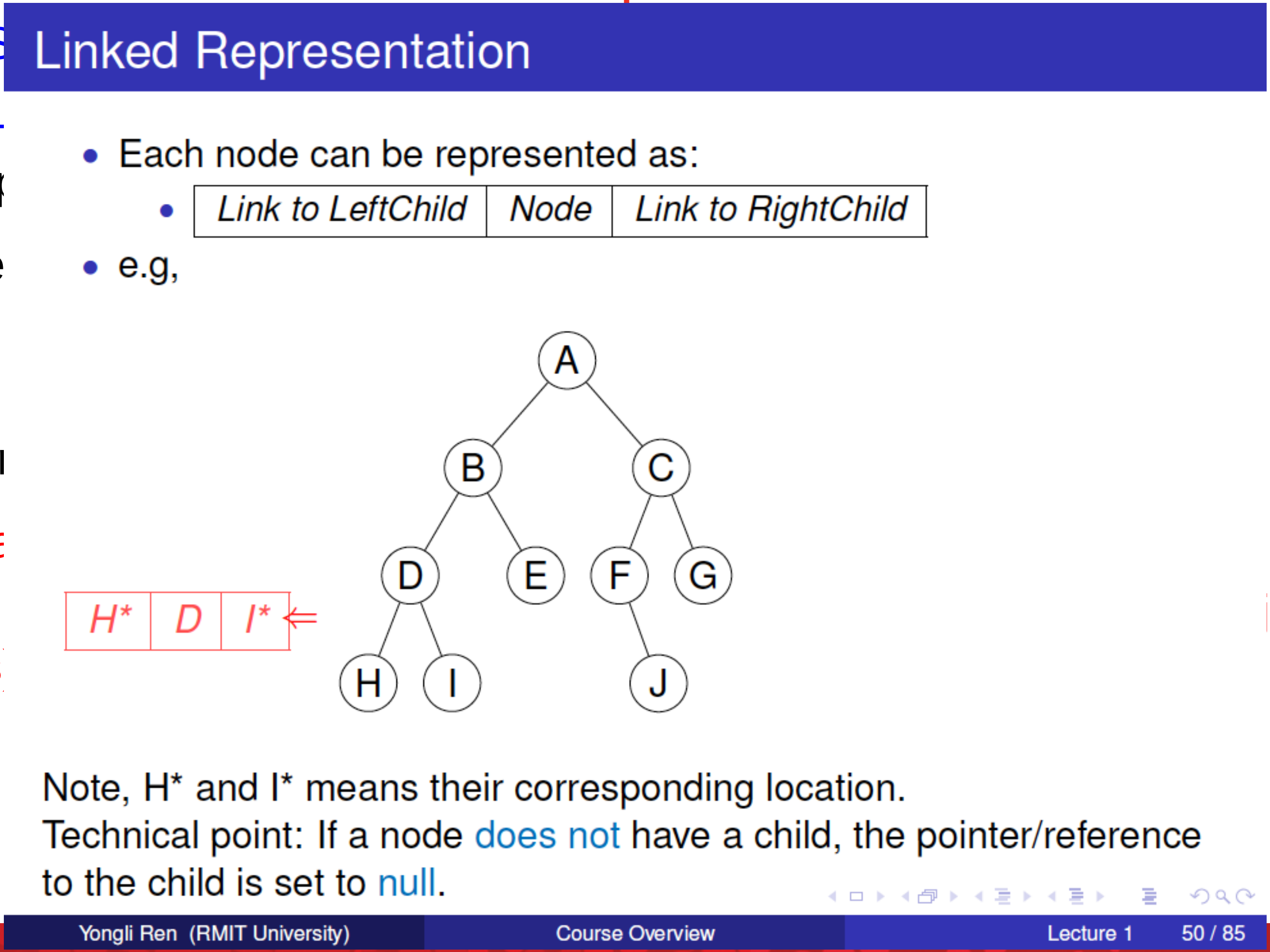
A	B	C	D	E	F	G	H	I	-	-	-	J	-	-

The root note is stored in at the 1st position. If a node occupies the k-th position then its left child is stored in $2k$ and its right child is stored into $2 * k + 1$ position.

Task A

- Implement the Tree Representations and their Operations (5 marks)
  - implement the *BSP* using
    - S
    - L
  - supp
- For the
  - you
  - You
    ope
- Extra ja
  - But, ava
    files

Lecture 1/4 and Lab 05



Linked Representation

- Each node can be represented as:
  -

Link to LeftChild	*Node*	*Link to RightChild*

- e.g,

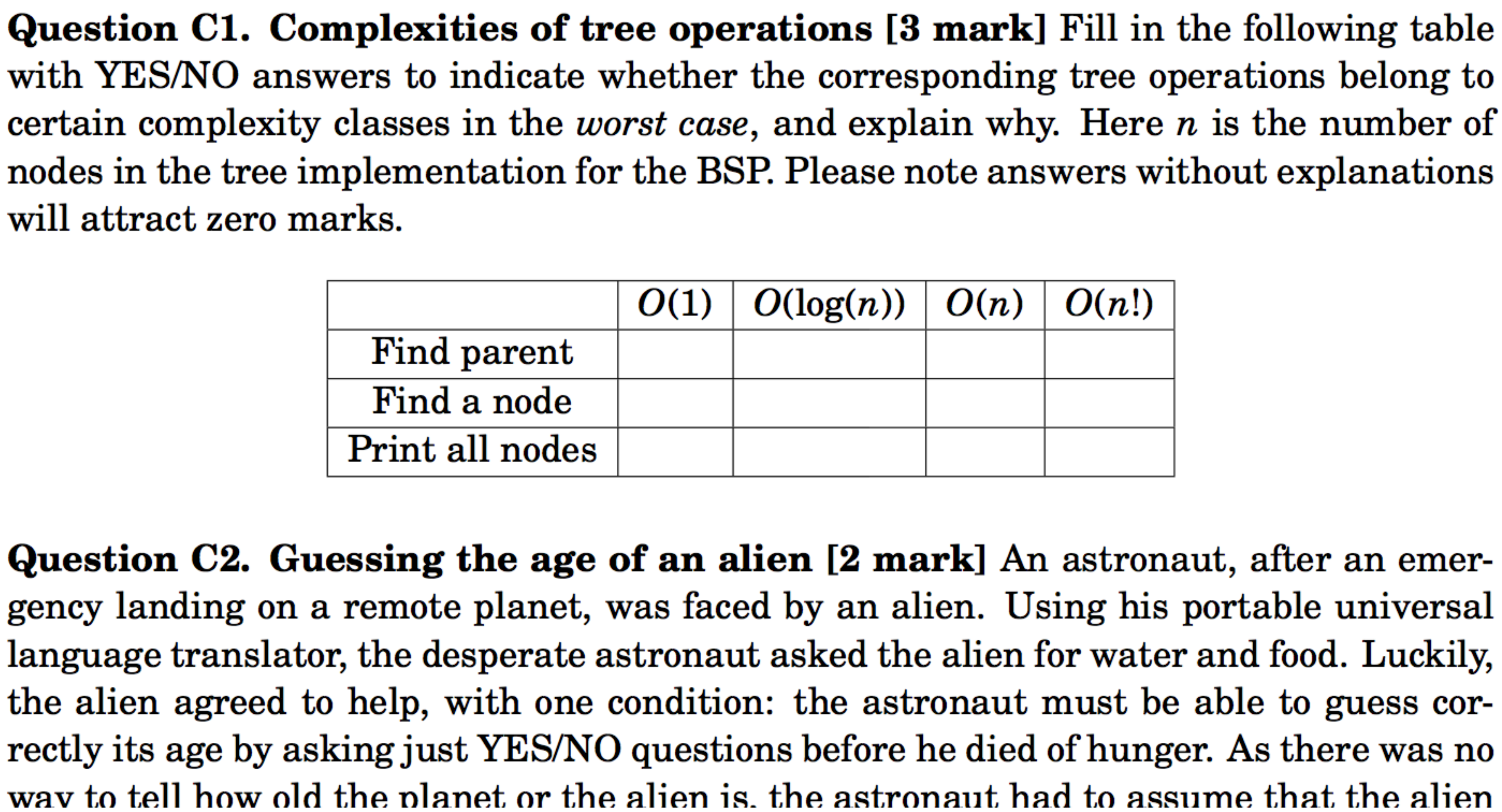
*H**	*D*	*I** ⇐

Note, H* and I* means their corresponding location.
Technical point: If a node does not have a child, the pointer/reference to the child is set to null.

Task B

- Task B: Evaluate your Data Structures (5 marks)
  - Write data generator; **Lab 2.**
  - Evaluate your TWO implemented structures, **Lab 2**
  - In terms of their time complexities for **Assignment specification**
    - the different operations:
      - E.g. Addition, Finding nodes, and traversal. **Lecture 4 and Lab 5**
    - different use case scenarios:
      - E.g. Growing BSP Tree (Additions), Finding node and its parent and children nodes, Printing the nodes (Traversal).
  - Report your analysis and evaluation of the different implementations.
    - Analyse, compare and discuss your results across different trees (with various complexities, e.g. depth, and the number of nodes) , representations and scenarios. **Report restructure & marking guide**
    - Provide your explanation on why you think the results are as you observed.
  - The report should be 6 pages or less, in font size 12.

Task C



Question C1. Complexities of tree operations [3 mark] Fill in the following table with YES/NO answers to indicate whether the corresponding tree operations belong to certain complexity classes in the *worst case*, and explain why. Here n is the number of nodes in the tree implementation for the BSP. Please note answers without explanations will attract zero marks.

	$O(1)$	$O(\log(n))$	$O(n)$	$O(n!)$
Find parent				
Find a node				
Print all nodes				

Question C2. Guessing the age of an alien [2 mark] An astronaut, after an emergency landing on a remote planet, was faced by an alien. Using his portable universal language translator, the desperate astronaut asked the alien for water and food. Luckily, the alien agreed to help, with one condition: the astronaut must be able to guess correctly its age by asking just YES/NO questions before he died of hunger. As there was no way to tell how old the planet or the alien is, the astronaut had to assume that the alien

Assessment

- Implementation (5/15)
  - Checkpoint (1 marks) in week 5’s Lab (Lab 04):
    - demo your implementation of the *Sequential representation* and *its operations*
- Report (10/15)

The Python Testing Code

- We provide a *python* script that
  - automates testing, based on input files of operations.
  - These are fed into the java framework which calls your implementations.
- The outputs resulting from any search or print operations are stored, then compared with the expected output.
- We have provided one sample input and expected files for your testing and examination (test1.* files).
- For our evaluation of the correctness of your implementation,
  - we will use the same python script and but different input/output/expected files,
  - so to avoid unexpected failures, please do not change the python script nor TreeTester.java.

Thanks!