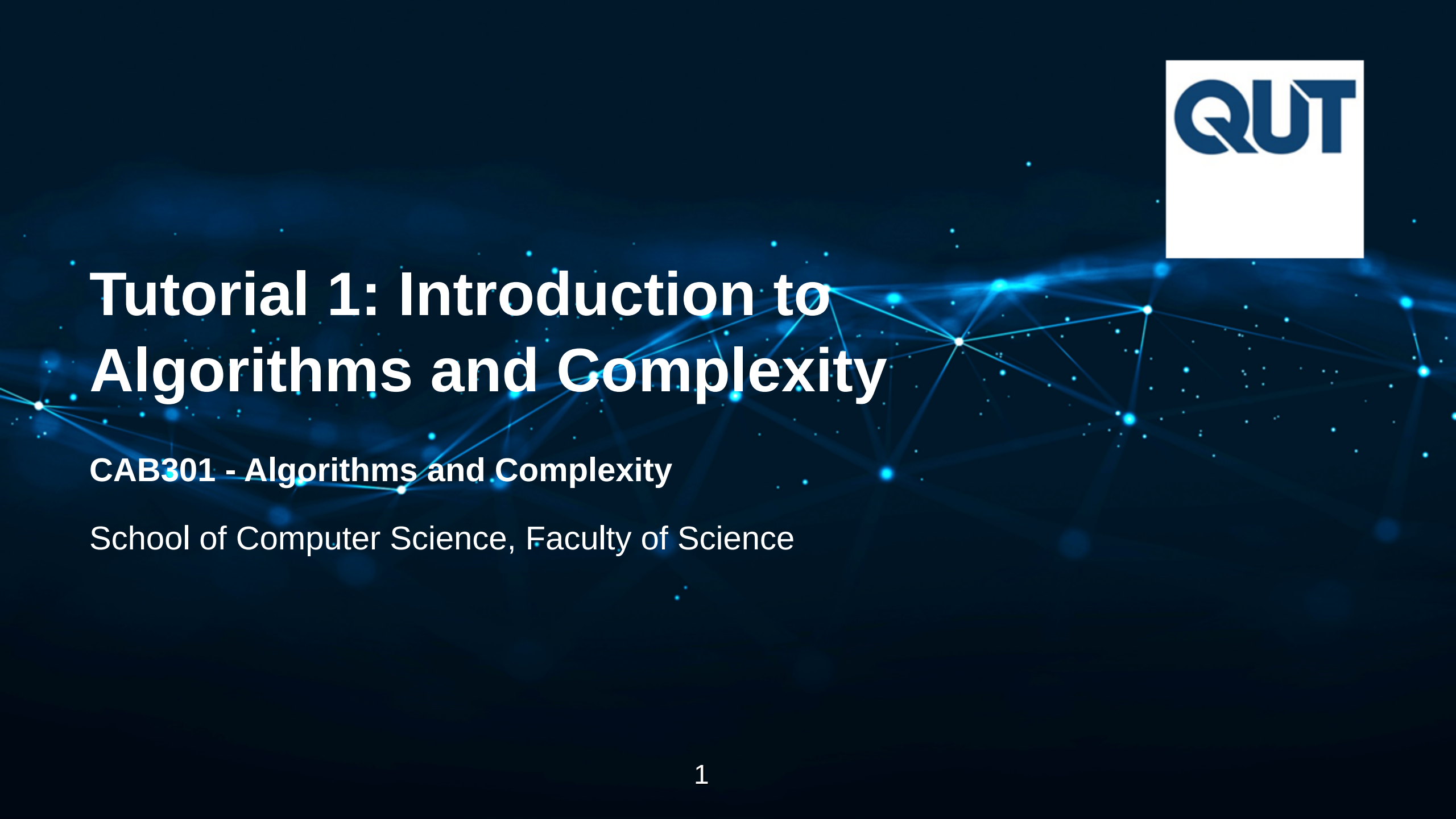

Tutorial 1: Introduction to Algorithms and Complexity

CAB301 - Algorithms and Complexity

School of Computer Science, Faculty of Science

ACKNOWLEDGEMENT OF TRADITIONAL OWNERS

QUT acknowledges the Turrbal and Yugara, as the First Nations owners of the lands where QUT now stands. We pay respect to their Elders, lores, customs and creation spirits. We recognise that these lands have always been places of teaching, research and learning.

QUT acknowledges the important role Aboriginal and Torres Strait Islander people play within the QUT community.

About Me

Agenda

1. Introduction to CAB301
2. Tutorial Questions
    i. Questions 1 and 2: definitions from lecture
    ii. Question 3: create an algorithm
    iii. Question 4: analyse and improve an algorithm
3. Programming Tasks
    i. Find distance between two closest elements in an array of numbers
    ii. Implement a **sorted list**

Introduction to CAB301

In a nutshell, CAB301 focuses on:

- Time efficiency analysis (Big-$\mathcal{O}$ notation)
  - **Theoretical**, through mathematical proofs
  - **Empirical**, through experiments
- Implement and use (add, insert, delete, traverse, etc.) of different data structures
  - Linear (arrays, lists, stacks, queues)
  - Non-linear (trees, graphs)
- Other algorithms (hashing, string matching)

Introduction to CAB301 - Assessment

- Assignment 1 (due week 6) and 2 (due week 9) are somewhat similar:
  - Implement **Abstract Data Types** (ADTs) and algorithms to manipulate them
  - Writing a **report**, using appropriate mathematical notation and pseudocode
  - **Time efficiency analysis** in Big-$\mathcal{O}$ notation
  - Including a **test plan** to validate the correctness of the implementation
- Assignment 3:
  - Software development project
  - Implement more complex algorithms

Tutorial Questions - Question 1 and 2

Q1: What is an algorithm and what are its characteristics? Are algorithms language specific?

Q2: What is pseudocode?

Hint: Refer to Week 1 lecture slides from pages 4 to 7.

Tutorial Questions - Question 3

Describe the standard algorithm for finding the **decimal representation** of a **positive binary number** (e.g., 1011 = 11).

a) in English

b) in the pseudocode defined in Lecture 1

> **ALGORITHM** BinaryToDecimal(m)
> // Given positive binary number m with n digits $(b_k b_{k-1} \ldots b_1 b_0)$
> // Returns decimal representation of m
> ...
> **return** ...

Tutorial Questions - Question 4

Try to improve the following algorithm for finding the distance between the two closest elements in an array of numbers.

<small>

ALGORITHM ClosestDistance($A[0..n-1]$)
// Input: Array $A[0..n-1]$ of n numbers
// Output: Minimum distance between two of its elements
$dmin \leftarrow \infty$
for $i \leftarrow 0$ **to** $n-1$ **do**
  for $j \leftarrow 0$ **to** $n-1$ **do**
    if $i \neq j$ **and** $|A[i] - A[j]| < dmin$
      $dmin \leftarrow |A[i] - A[j]|$

Programming Tasks - Question 5

Implement the find closest distance algorithm in C#, and test it like:

```
int[] numbers = { 4, 10, 12, 3, 6, 9, 7, 15 };
int result = MinimumDistance(numbers);
Console.WriteLine($"Minimum distance: {result}");
// Expected output: 1 (the closest numbers are 6 and 7)
```

Programming Tasks - Question 6

Implement a sorted list in C# with the following methods:

- `Insert(int value)` to add a new element to the list at the correct position
- `Delete(int value)` to remove an element from the list
- `Search(int value)` to find an element in the list
- `IsEmpty()` and `IsFull()` to check if the list is empty or full

There should also be two properties:

- `Count` to return the number of elements in the list
- `Capacity` to return the maximum number of elements the list can hold