# Computer Science 3A

## Practical Assignment 5

4 April 2024

Time: 4 April 2024 — 17h00 Marks: 50

Practical assignments must be uploaded to `eve.uj.ac.za` **before** 17h00. Late submissions **will not be accepted**, and will therefore not be marked. You are **not allowed to collaborate** with any other student.

**Text Search and Closest Good Restaurant Finder using ArrayLists and Binary Trees**

Good restaurants that are not part of a franchise can sometimes be hard to find in Johannesburg. You have been tasked with the job of implementing a program that uses ArrayLists to read the Restaurant objects and load their names into a Binary Search Tree (BST - A binary tree that contains elements with specific lexicographic ordering that can be searched in logarithmic time). The user can then provide a query to search for a restaurant name and gives them the option to provide GPS coordinates (in longitude:latitude format) from the command line (such as 27.9989:-26.1833 for UJ or you can use `http://www.latlong.net/convert-address-to-lat-long.html`) and find the closest restaurant to that location, according to an ArrayList based database loaded from a binary file of serialized objects that have been provided (that contains ten good restaurants reviewed by me - ProfVDH).

**A reminder on ArrayLists**

ArrayLists as a data structure allow you to use an array while storing an "infinite" number of elements. The strategy for dealing with the dynamic expansion of the underlying array can be managed in two ways:

- Using an incremental approach — the array is expanded by a fixed number of elements when it gets too full.
- Using a doubling approach — the array is expanded by doubling the number of elements when it gets too full.

You are required to implement an ArrayList that realises the above strategies. Both strategies should be implemented, and in the constructor for your ArrayList, you should be able to specify a value of one (1) to indicate an incremental strategy or two (2) if a doubling strategy is being followed. This should be managed when the array is expanded.

You must complete the classes and methods marked by:

```
//TODO: COMPLETE CODE HERE
```

To test your implementation, a *Main* class has been provided. You should not add any extra functions to the provided classes; only complete the methods that have been indicated.

The following files must be submitted to EVE:

1. *studentnumber*_p5.zip

# Marksheet

1. ArrayList: add [5]
2. ArrayList: remove [5]
3. ArrayList: expandArray [8]
4. ArrayList: iterator [2]
5. BinaryTree: nodeDepth [5]
6. BinaryTree: PreorderElementTraversal [5]
7. BinaryTree: insertRecursive [5]
8. BinaryTree: binarySearchRecursive [5]
9. Compilation and Correct execution. [10]