CSC2001F Assignment 1
Binary Search Tree
TWLCOM001 Comfort Twala

R E P O R T

OO Design:

fileHandler class is the helper class that opens the file with names, creates Student objects and populates the data structures depending on which you need, using dataArray() method for when you require an array and dataTree() method for when you require a BinarySearchTree.
Student class is the data class. It stores the student's ID, name and surname. It interacts with both the data structure app classes ArrayApp and BSTApp.
ArrayApp class creates an array of Students objects and uses printAllStudents() to print out all Student objects and printStudent(studentID) to print out a specific Student objects depending on the studentID.
BSTApp class creates a BinarySearchTree of Student objects and uses printAllStudents() to print out all Student objects and printStudent(studentID) to print out a specific Student objects depending on the studentID.
AccessArrayApp class is the program class that is used to invoke ArrayApp method.
AccessBSTApp class is the program class that is used to invoke BSTApp method.

Goal of Experiment:

The Goal of the experiment was to compare an Array to a Binary Search Tree and determine each data structure's time complexity through searching objects within that data structure and capturing how many operations each data structure took to find said object.

I executed the experiment by using 10 subsets of the Student objects starting with 500 objects, then 1000, till 5000. (n = 500, 1000, ..., 5000). Each object in all subsets where ran using AccessArrayApp and AccessBSTApp and the results were captured (*saved in data/experiment)*. From these results a minimum, maximum and average value were determined for each subset and were captured (*saved in data/experiment/analysis*). From captured data a data table was created (*saved in data/experiment/results/stats.csv*) and graph was plotted (*saved in data/experiment/results/graph.png*) to visualise the difference between both the data structures. Experiment was conducted using python scripts (*saved in script*) for efficiency, accuracy and most importantly automation.

Experiment was successful and the difference is clearly visible.

Test results:
Part 1: Array

AccessArrayApp:
 -known: (*saved in data/part1/arrayTest/known*)
    1. /usr/bin/java -cp bin AccessArrayApp BKSAVA009
       Ava Beukes
    2. /usr/bin/java -cp bin AccessArrayApp MLTLUK019
       Luke Malatji
    3. /usr/bin/java -cp bin AccessArrayApp TSTSIP016
       Siphesihle Tsotetsi

 -unknown: (*saved in data/part1/arrayTest/unknown*)
    1. */usr/bin/java -cp bin AccessArrayApp TWLCOM001*
       Access Denied!
    2. */usr/bin/java -cp bin AccessArrayApp MILWRD032*
       Access Denied!
    3. */usr/bin/java -cp bin AccessArrayApp TRFKON002*
       Access Denied!

 -none: (*saved in data/part1/arrayTest*)
    /usr/bin/java -cp bin AccessArrayApp
    MLLNOA014 Noah Maluleke
    WTBJAY001 Jayden Witbooi
    KHZOMA010 Omaatla Khoza
    MLTLUK019 Luke Malatji
    NKNTHA021 Thato Nkuna
    ....
    MSXROR015 Rorisang Mosia
    DNLAYA006 Ayabonga Daniels
    CHKOFE015 Ofentse Chauke
    MNGREA015 Reatlegile Moeng
    SHBCAL017 Caleb Shabangu

Part 1: BST

AccessBSTApp:

-known: (*saved in data/part1/BSTTest/known*)

1. /usr/bin/java -cp bin *AccessBSTApp* BXXAMO012
   Amogelang Booi
2. /usr/bin/java -cp bin *AccessBSTApp* MLLASE002
   Asemahle Maluleke
3. /usr/bin/java -cp bin *AccessBSTApp* TSTPHE008
   Phenyo Tsotetsi

-unknown: (*saved in data/part1/BSTTest/unknown*)

1. */usr/bin/java -cp bin AccessBSTApp KAITOO001*
   Access Denied!
2. */usr/bin/java -cp bin AccessBSTApp POPLAP071*
   Access Denied!
3. */usr/bin/java -cp bin AccessBSTApp SAMJRP007*
   Access Denied!

-none: (*saved in data/part1/BSTTest*)

```
/usr/bin/java -cp bin AccessBSTApp
BKSALW003 Alwande Beukes
BKSAMA002 Amahle Beukes
BKSAMA008 Amahle Beukes
BKSAMO002 Amohelang Beukes
BKSAMO027 Amogelang Beukes
.....
WTBTHA010 Thato Witbooi
WTBTSH002 Tshegofatso Witbooi
WTBTSH025 Tshegofatso Witbooi
WTBTSH028 Tshegofatso Witbooi
WTBWAR001 Warona Witbooi
```

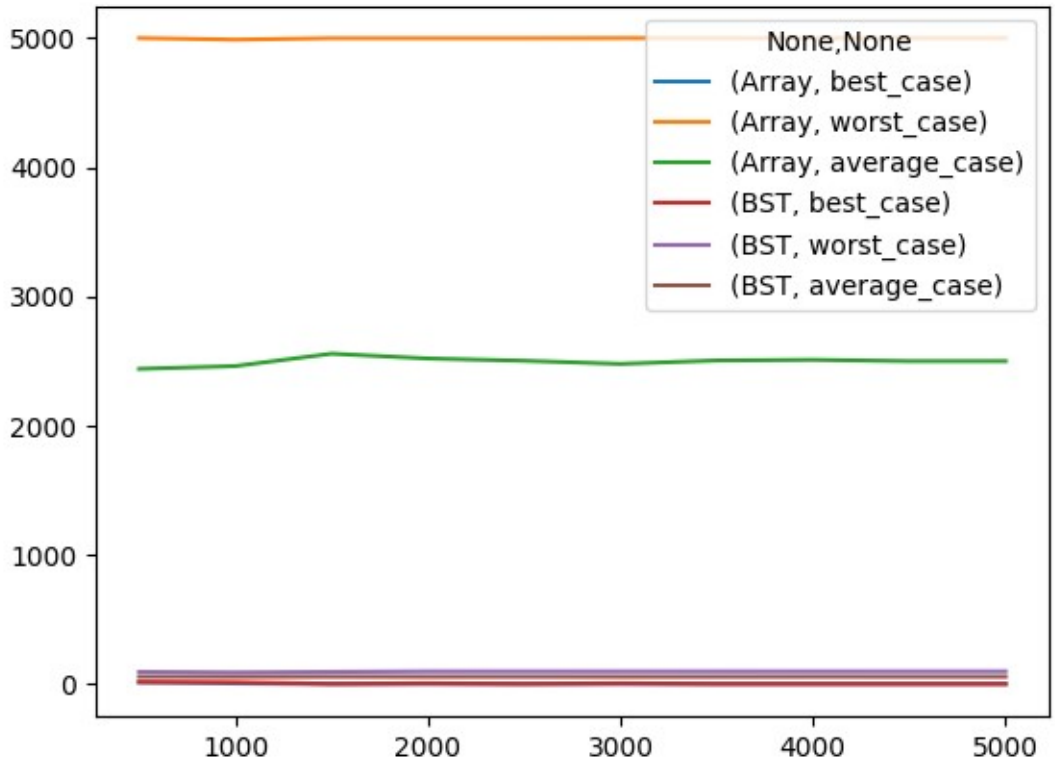
Results – tables and/or graph

Table: Array vs BST (min, max, average)
(*saved in data/experiment/results/stats.csv*)

	Array	Array	Array	BST	BST	BST
	best_case	worst_case	average_case	best_case	worst_case	average_case
500	18.0	5000.0	2440.396	16.0	96.0	59.04
1000	7.0	4988.0	2462.084	16.0	92.0	58.844
1500	1.0	4999.0	2557.920666666667	4.0	96.0	59.29866666666667
2000	2.0	4999.0	2520.7015	8.0	100.0	58.992
2500	1.0	4999.0	2503.106	4.0	100.0	59.1472
3000	2.0	5000.0	2478.984	8.0	100.0	58.90133333333333
3500	1.0	5000.0	2505.4465714285716	4.0	100.0	58.91314285714286
4000	1.0	4998.0	2510.45075	4.0	100.0	59.153
4500	1.0	5000.0	2500.41	4.0	100.0	59.07822222222222
5000	1.0	5000.0	2500.5	4.0	100.0	59.0384

**** Generated using grapher.py*

Graph: Operation count vs n



**** Generated with grapher.py*

Discussion of Results:

The following results of the Array vs Binary Search Tree battle showcase that the Binary Search Tree is ultimately the better data structure to use for searching between the 2.
The Best case for the Array is 1, and that is when the subset contains the first element of the Array, while the best case in this case of the BST is 4.
The worst case for the Array is 5000, meaning it searched throughout the whole Array to find object, while the worst case for the BST is 100. That is 50 times smaller than that of the Array. Using the worst case we can deduct that out tree height was 100.
The average case for the Array was approximately 2500, that is half the length of the Array. Meaning on average half the Array is searched for an Object. The average case for the BST is ~59. Meaning that on average 59 nodes are searched and that is only 1,18% of the data amount.
The time time complexity of the BST is O(h) with h being the height so this BST has a time complexity of O(100).
The time complexity of the Array is O(n), meaning this Array has a time complexity of O(5000).
Clearly the Binary Search Tree is the better option for Search.

Creativity:

-- Automated the data testing using large data set for a more accurate representation and analysis.
-- Created python scripts to run experiment, analyse the data results and present it in a easy to read/understand manner. (*saved in data/expirement/analysis*)
-- Created python script to use analysed data to create data table and draw the graph. (*saved in data/expirement/results/*)
-- Added README.md for project usage details.
-- Added targets in Makefile to run python scripts

Git Usage:

```
commit 180741b270bd2ac732348b5134fc2ae651500198
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Wed Apr 7 13:58:44 2021 +0200

39    stats generated by 'make stats'

commit 9823381c0033a1b0ffb21f65c909462a0ebfa105
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Wed Apr 7 13:55:41 2021 +0200

38    analysis results generated by 'make analysis'

commit 14b8540ea8237e3acdc360205dab6d4f82f40d17
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Wed Apr 7 13:52:20 2021 +0200

37    experiment results generated by 'make experiment'

commit 471830994d353faf541b0aeead3a192b6ae644a6
```

```
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Wed Apr 7 02:22:16 2021 +0200

36    Modified Makefile and README for grapher/stats script

commit 7f69b59722cb8eea2cdb25fed1789cd8f57fcc15
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Wed Apr 7 02:17:58 2021 +0200

35    grapher script code added and tested

commit e910b0a358a289145bed25b21a55fee5b798cfa8
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Wed Apr 7 02:17:02 2021 +0200

34    analyser code added and tested

commit b4cfcdc9e172604fff8bd8a30465fae9729cc826
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Wed Apr 7 00:28:42 2021 +0200

33    code refractured

commit 9fabe87d89de28f218657aa29440e4ec1100add1
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Wed Apr 7 00:26:37 2021 +0200

32    added grapher script to create csv data stats and graph them

commit 2f4e139f8201cf27965ad7c44b56692b156a5bf3
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Tue Apr 6 23:15:48 2021 +0200

31    code refractoring

commit 7ded1131167ccf1f01f5ca2f0438a40ac5181719
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Tue Apr 6 23:12:09 2021 +0200

30    documentation updated
.....
commit 219a6b78a8b33f76548d66d512ff9262c6befe80
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Mon Apr 5 23:12:00 2021 +0200

5    AccessArrayApp main method added

commit 746132fbc90309ca7edf9417b0a3110cadb12cf2
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Mon Apr 5 23:07:04 2021 +0200

4    ArrayApp constructor and methods added

commit 7b90d99259b48f5d708f50f1e2c19f3fc694e953
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Mon Apr 5 23:03:12 2021 +0200

3    Student constructor and methods added

commit e89f026910b5860692c15d09a104112df3c8fd56
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Mon Apr 5 22:58:14 2021 +0200

2    fileHandler constructor and methods added

commit edc51e546a189771a42442236f5f6df2a8f7ebbf
Author: Comfort-Twala <kontreitroos@gmail.com>
Date:   Mon Apr 5 16:45:47 2021 +0200

1    Project Skeleton
```