Test Plan and Test Outputs

ICT283 – Data Structures and Abstractions – Assignment 2

32067232 – Jake Kroon

Description of testing strategy

In Testing my classes, I started at the very bottom of the class hierarchy and worked my way up. I started with the Date and Unit classes and then tested Results. From there I test my MyVector. As I need to pass in a BinaryTree of units to the Student class I then tested my BinaryTree. The Student Class came next and then finally the StudentIO class. I did this because if one of my classes at the bottom of the hierarchy, it can be assumed that the related functions in the parent classes would also function incorrectly causing the entire program to fail over time if not instantly.

So what I've done is written a program that will test each class within a method. There is a directory called "Testing" that contains this program. For each class I have tested the set and get methods, constructors and copy constructors if they exist. I've also tried to test bounds in some cases. Though given that I may assume the input file is correct, that is likely outside the scope of this assessment. Though it is good to know this for future developments on a project.

Testing is shown over the page on a class by class basis. Every time a new class is being tested I will write its name in a **bold** font in the left column where I number my tests. Every time I create a test I will begin by initialising an object. If I am using a pointer to initialise the object I will say so. I only use pointers on objects that would use a pointer in my actual program. Strings are not tested for boundaries because as far as I'm aware they take any input due to being a special data structure. I do however still test their set and get methods.

At the end of the unit testing, there is my test outputs, first it goes through the unit tests. These are screenshots of each class and they for the most part directly relate to the unit test table, except where information has been omitted due to being irrelevant. Below that is output from the release build of my program. All options are tested and shown to be working. the .csv files data can also be found at the end of this document. The .csv file is called students.

Test	**Description**	**Expected Output**	**Passed**
UNIT			
1	Check default constructor name variable set.	NO_INPUT	Passed
2	Check default constructor id variable set.	NO_INPUT	Passed
3	Check default constructor credits variable set.	0	Passed
4	SetCredits(3) and GetCredits()	3	Passed
5	SetName(“Data Structures and Abstractions”) and GetName()	Data Structures and Abstractions	Passed
6	SetID(“ICT283”) and GetID()	ICT283	Passed
7	SetCredits(-1)	Should wrap around and get the highest value (I haven't implemented input checking)	It did as expected but if the wrong input is entered an error will occur though this wont crash the program
8	SetCredits(64436)	64436	It did as expected, but no credits should be this large.
DATE			
1	Check default constructor day variable set	0	Passed
2	Check default constructor month variable set	0	Passed
3	Check default constructor year variable set	0	Passed
4	SetDay(1) and GetDay()	1	Passed
5	SetMonth(1) and GetMonth()	1	Passed
6	SetYear(2015) and GetYear()	2015	Passed

7	SetDay(-1)	Should wrap around and get the highest value	It does as expected, but clearly this is not a real day
8	SetYear(-1)	Should wrap around and get the highest value	It does as expected but, clearly this is not a realistic year
9	SetDay(65536)	65536	Passed obviously not a realistic day though
10	SetYear(65536)	65536	Passed obviously not a realistic day
			The same result will occur for Months as it does for Days and Years
Results			
1	Check default constructor result set	0	Passed
2	Check default constructor semester set	0	Passed
3	SetResult(90)	90	Passed
4	SetSemester(2) and GetSemester()	2	Passed
5	SetUnitId(“ICT283”) and GetUnitId()	ICT283	Passed
6	SetDay(1) and GetDay()	1	Passed
7	SetMonth(1) and GetMonth()	1	Passed
8	SetYear(2015) and GetYear()	2015	Passed
9	SetResult(0.000000000000000013) and GetResult()	Unexpected Result	Value was truncated, 0 was printed
10	SetResult(1111111111)	Unexpected Result	Value was truncated 2521176519 was printed
MyVector<unsigned>	Declared as pointer *vec		
1	Check expansion by initialising up to 3001 integers in a loop	Doesn't crash	Passed
2	GetSize(0)	3001	Passed
3	Check first index	0	Passed

4	Check last index	3000	Passed
5	Copy Constructor doesn't crash	Doesn't crash	Passed
6	Copied vector GetSize(0)	3001	Passed
7	Copied vector get first index	0	Passed
8	Copied vector get last index	3000	Passed
9	Check memory locations	Object Address not the same passed	Passed
10	Initialise MyVector object with parameter of 5	Not crash	Passed
11	Check expansion up to 9	Not Crash	Passed
12	GetSize()	10	Passed
13	First index	0	Passed
14	Last index	9	Passed
MyVector<Date>	Declared as pointer *dates		
1	Initialise 3000000 date objects	Not crash	Passed
2	GetSize	3000000	Passed
3	Test for index of -1	Throw exception	Passed
Student	Declared as pointer *s		
1	GetStudentId()	0	Passed
2	GetFirstName()	NOT INIT	Passed
3	GetLastName()	NOT INIT	Passed
4	SetStudentID(32067232) and GetStudentID	32067232	Passed
5	SetFirstName(“Jake”) and GetFirstName()	Jake	Passed
6	SetLastName(“Kroon”) and GetLastName()	Kroon	Passed
7	SetResult(0, 80) and GetResult(0)	80	Passed
8	SetResultSemester(0, 1) and GetResultSemester(0)	1	Passed
9	SetUnitId(0, “ICT283”) GetUnitId(0)	ICT283	Passed

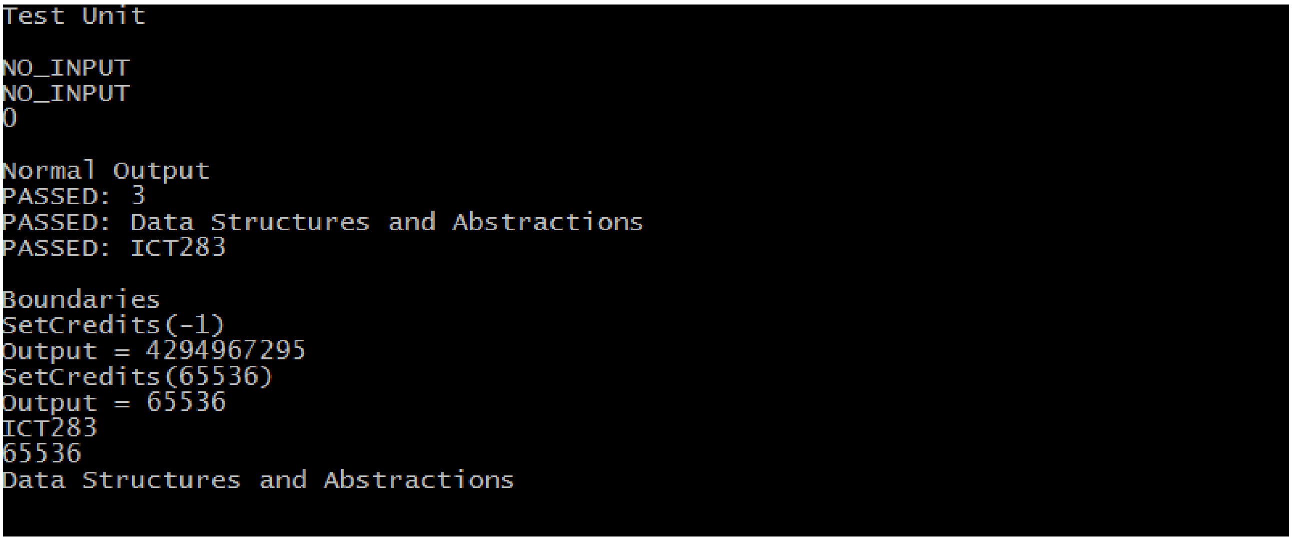
10	SetDay(0, 26) and GetDay(0)	26	Passsed
11	SetMonth(0, 5) and GetMonth(0)	5	Passed
12	SetYear(0, 2015) and GetYear(0)	2015	Passed
	The index of 1 has not been set		
13	GetResult(1)	0	Passed
14	GetResultSemester(1)	0	Passed
15	GetUnitId(1)	NULL	Passed
16	GetDay(1)	0	Passed
17	GetMonth(1)	0	Passed
18	GetYear(1)	0	Passed
	The indexes 0 – 5 have been set, please refer to the code to see the values		
19	GetHighestMark()	81	Passed
20	GetLowestMark()	50	Passed
21	CalculateGPA(unitTree)	2.6	passed
	The following are performed on the copy constructed object only		
22	Copy constructor test initialise	not crash	Passed
23	GetStudentId()	32067232	Passed
24	GetFirstName()	Jake	Passed
25	GetLastName()	Kroon	Passed
26	GetResult(0)	80	Passed
27	GetResultSemester(0)	1	Passed
28	GetUnitId(0)	ICT283	Passed
29	GetDay(0)	26	Passed
30	GetMonth(0)	July	Passed
31	GetYear(0)	2015	Passed
32	GetResult(final index)	81	Passed
33	Addresses don't match	Addresses do not match there for are not the same	Passed

34	student::SetUnit(“ICT283”, unitTree) static test	Not crash (unit set)	Passed
35	student::GetUnitName() static test	ICT283	Passed
36	CalculateGPA(unitTree, 1)	2	Passed
37	CalculateGPA(unitTree, 2)	3	Passed
38	CalculateGPA(unitTree, 3)	1	Passed
39	CalculateGPA(unitTree, 4)	4	Passed
40	CalculateGPA(unitTree, 5)	0	Passed
BinaryTree<int>			
1	Insert from 5 to and including 1	Don't crash	Passed
2	Insert from 5 to and including 20	Don't crash	Passed
3	PreOrder	Print output, no duplicates	Passed
4	InOrder	Print output no duplicates	Passed
5	Copy Constructor	Don't crash	Passed
6	Pre Order	Match above	Passed
7	In Order	Match above	Passed
8	Address don't match	Address don't match	Passed
9	Equals operator on new object	Address don't match	Passed
10	Search for 1 with function pointer	1	Passed
BinaryTree<Unit>			
1	Insert units from file unitInput.csv (contains duplicates)	Doesn't crash	Passed
2	Post Order	Prints all units, no duplicates	Passed
3	In Order	Print all units, no duplicates	Passed
4	Search (uses inputFile.csv and searches for every unit)	Print all found	Passed

	Please note during this all the overloaded operators were used and prove to be functioning as expected		
StudentIO	takes in students.csv in the constructor as well as outdata.txt, writes to outdata.txt		
1	InitialiseIO	Don't crash	Passed
2	HighestMarkOutput for 32067232	Print appropiate message to the console as well as write to file	Passed
3	Lowest Mark Output for 32067232	Print appropriate message to the console as well as write to file	Passed
4	GetGPACalcOutput for 32067232	Print appropriate message to the console as well as write to file	Passed
5	GetHighestLowest	Print appropriate message to the console as well write to file	Passed
6	GetYearGPA for 32067232	Print appropriate Message to the console as well as write to file	Passed

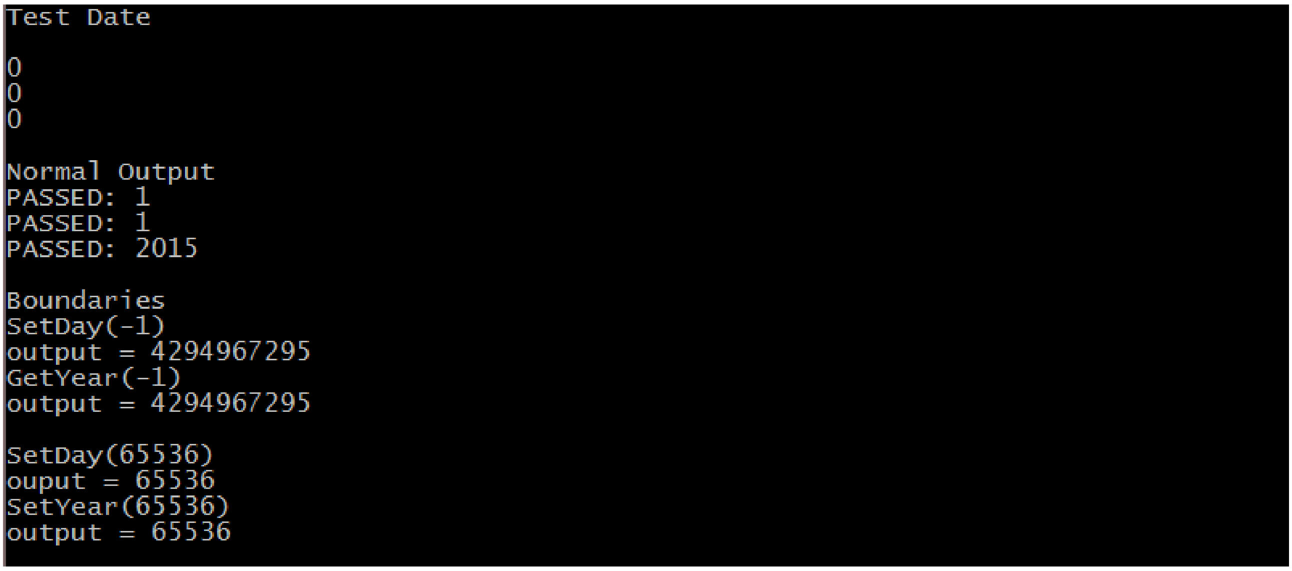
Images for unit tests

Unit



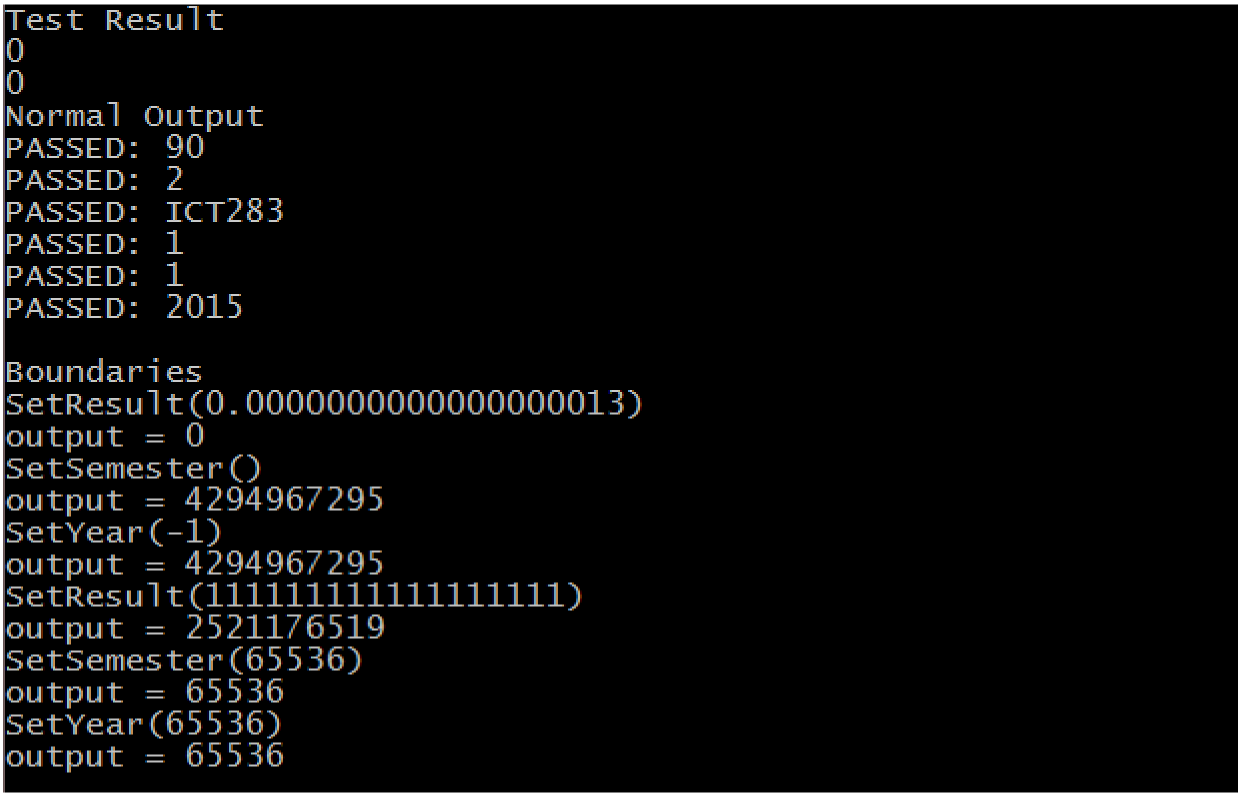
```
Test Unit

NO_INPUT
NO_INPUT
0

Normal Output
PASSED: 3
PASSED: Data Structures and Abstractions
PASSED: ICT283

Boundaries
SetCredits(-1)
Output = 4294967295
SetCredits(65536)
Output = 65536
ICT283
65536
Data Structures and Abstractions
```

Date



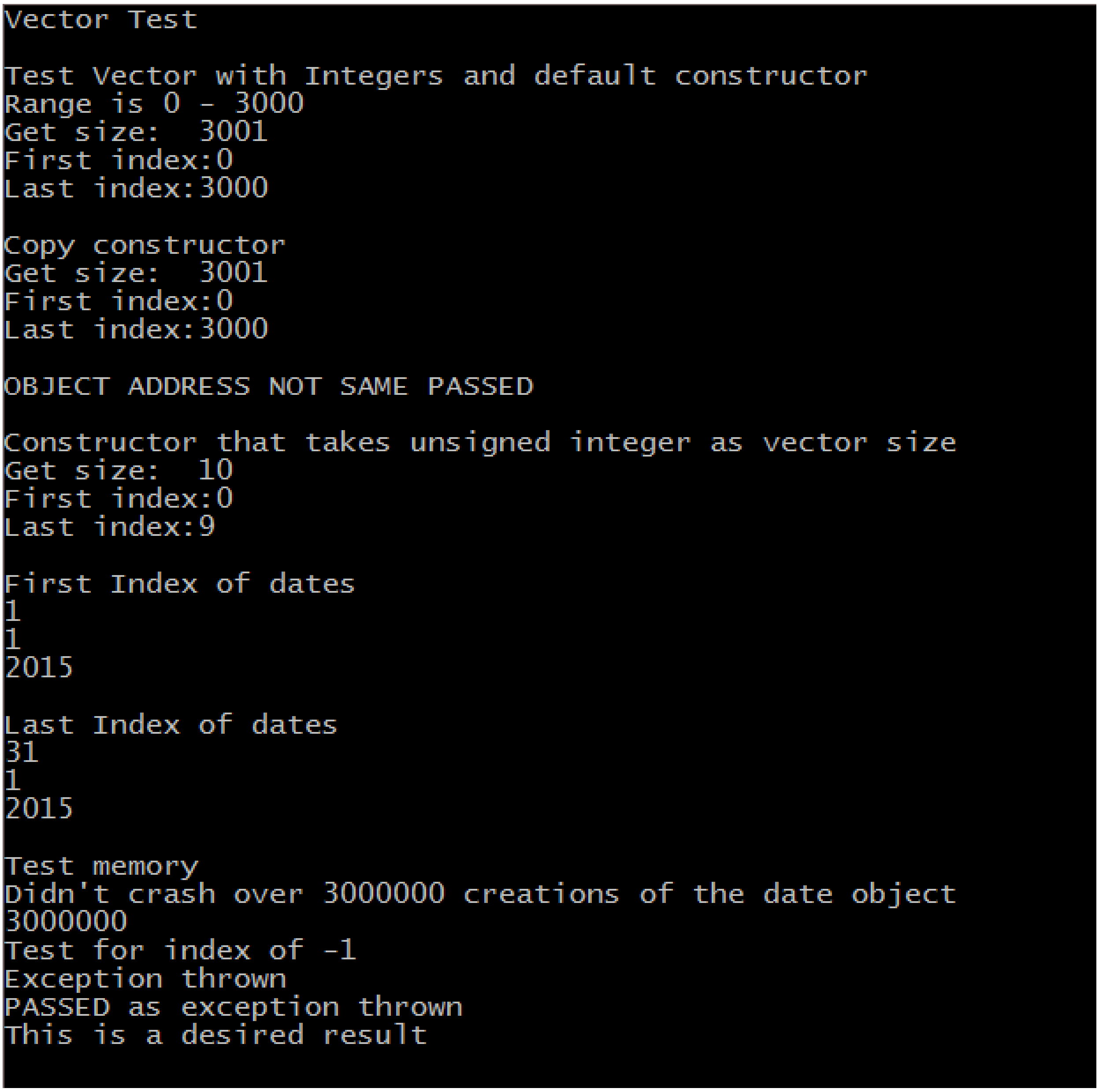
```
Test Date

0
0
0

Normal Output
PASSED: 1
PASSED: 1
PASSED: 2015

Boundaries
SetDay(-1)
output = 4294967295
GetYear(-1)
output = 4294967295

SetDay(65536)
ouput = 65536
SetYear(65536)
output = 65536
```

Result



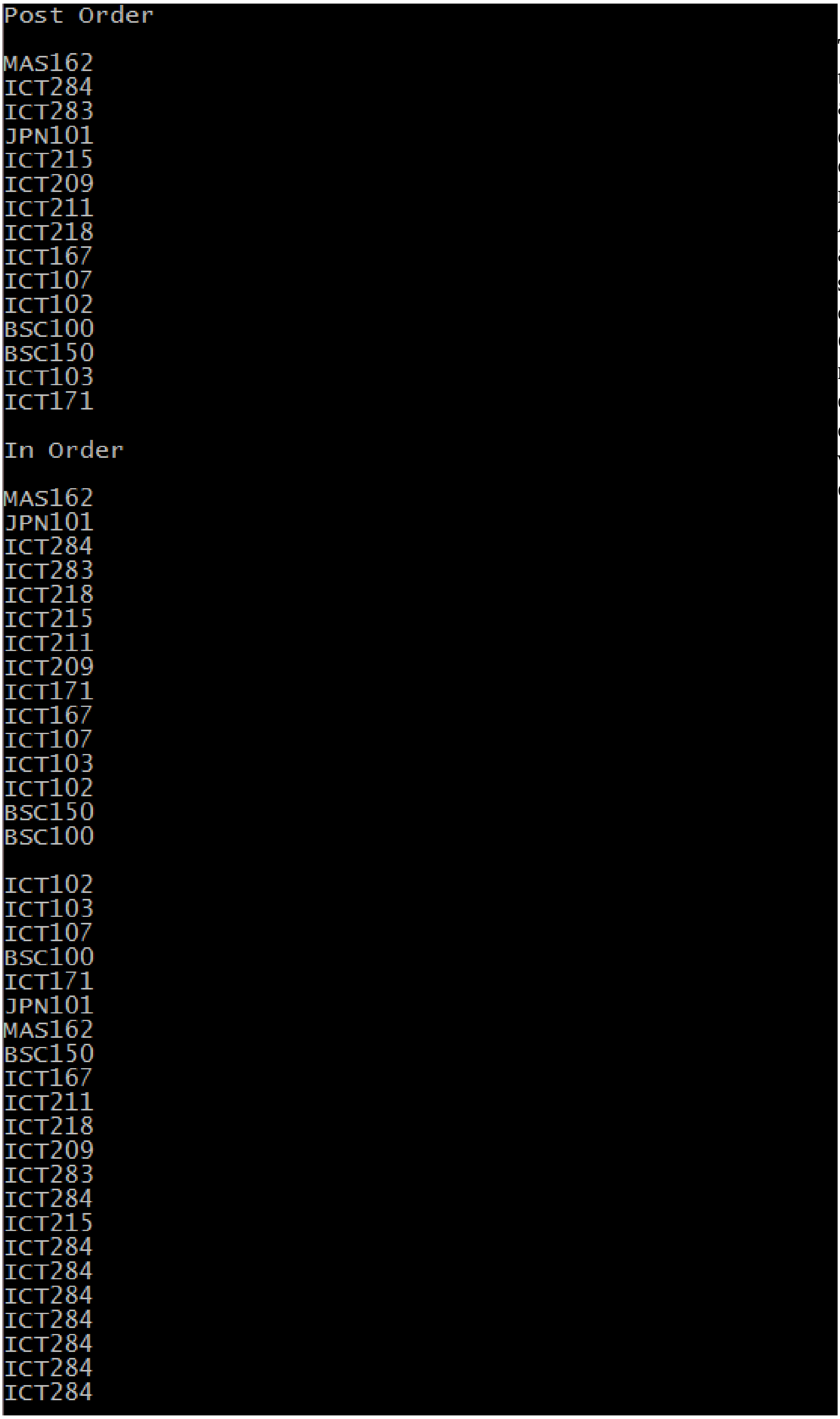
```
Test Result
0
0
Normal Output
PASSED: 90
PASSED: 2
PASSED: ICT283
PASSED: 1
PASSED: 1
PASSED: 2015

Boundaries
SetResult(0.0000000000000000013)
output = 0
SetSemester()
output = 4294967295
SetYear(-1)
output = 4294967295
SetResult(111111111111111111)
output = 2521176519
SetSemester(65536)
output = 65536
SetYear(65536)
output = 65536
```

MyVector



```
Vector Test

Test Vector with Integers and default constructor
Range is 0 - 3000
Get size:  3001
First index:0
Last index:3000

Copy constructor
Get size:  3001
First index:0
Last index:3000

OBJECT ADDRESS NOT SAME PASSED

Constructor that takes unsigned integer as vector size
Get size:  10
First index:0
Last index:9

First Index of dates
1
1
2015

Last Index of dates
31
1
2015

Test memory
Didn't crash over 3000000 creations of the date object
3000000
Test for index of -1
Exception thrown
PASSED as exception thrown
This is a desired result
```

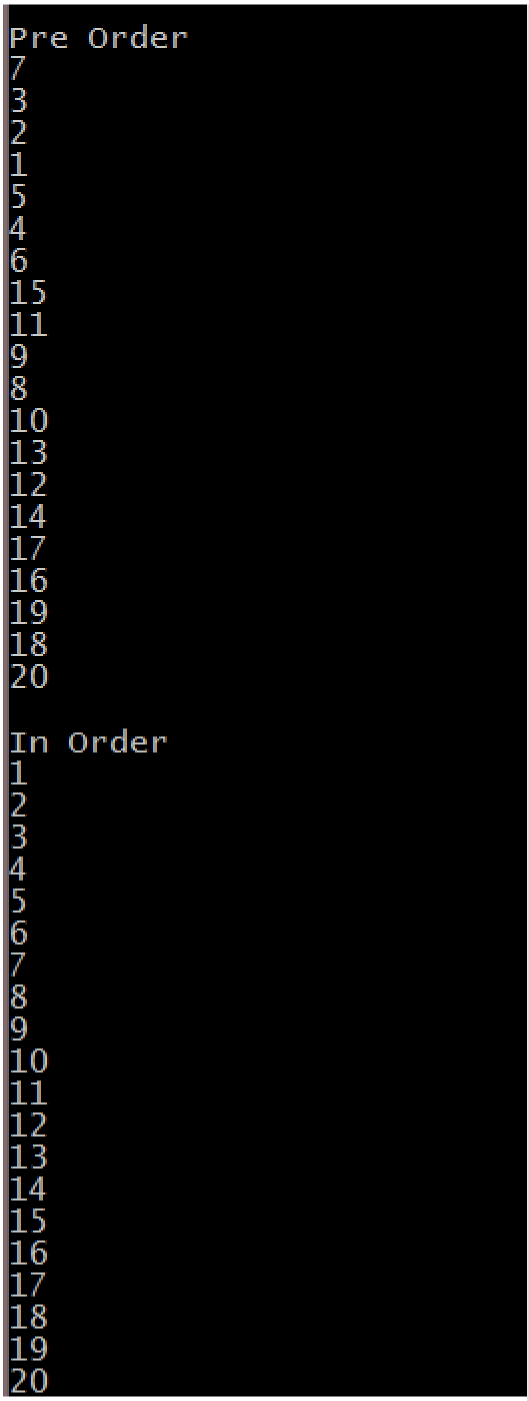
BinaryTree<Unit>



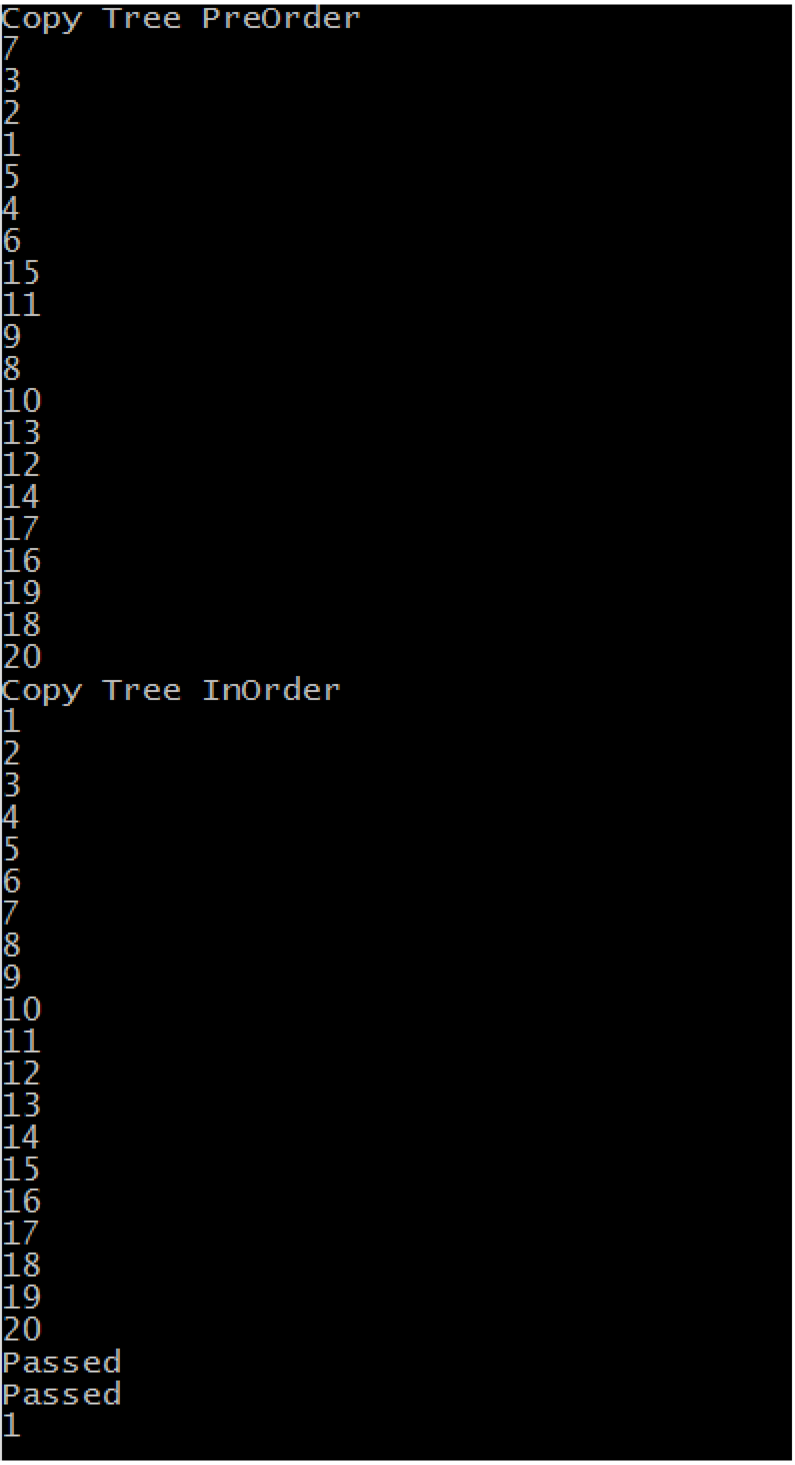
```
Post Order

MAS162
ICT284
ICT283
JPN101
ICT215
ICT209
ICT211
ICT218
ICT167
ICT107
ICT102
BSC100
BSC150
ICT103
ICT171

In Order

MAS162
JPN101
ICT284
ICT283
ICT218
ICT215
ICT211
ICT209
ICT171
ICT167
ICT107
ICT103
ICT102
BSC150
BSC100

ICT102
ICT103
ICT107
BSC100
ICT171
JPN101
MAS162
BSC150
ICT167
ICT211
ICT218
ICT209
ICT283
ICT284
ICT215
ICT284
ICT284
ICT284
ICT284
ICT284
ICT284
ICT284
```

Though at first this may not appear to be ordering correctly, if you refer to the ASCII table and look at how strings are compared in C++ as well as my overloaded operator for comparison you will find this is quite accurate.

BinaryTree<int>

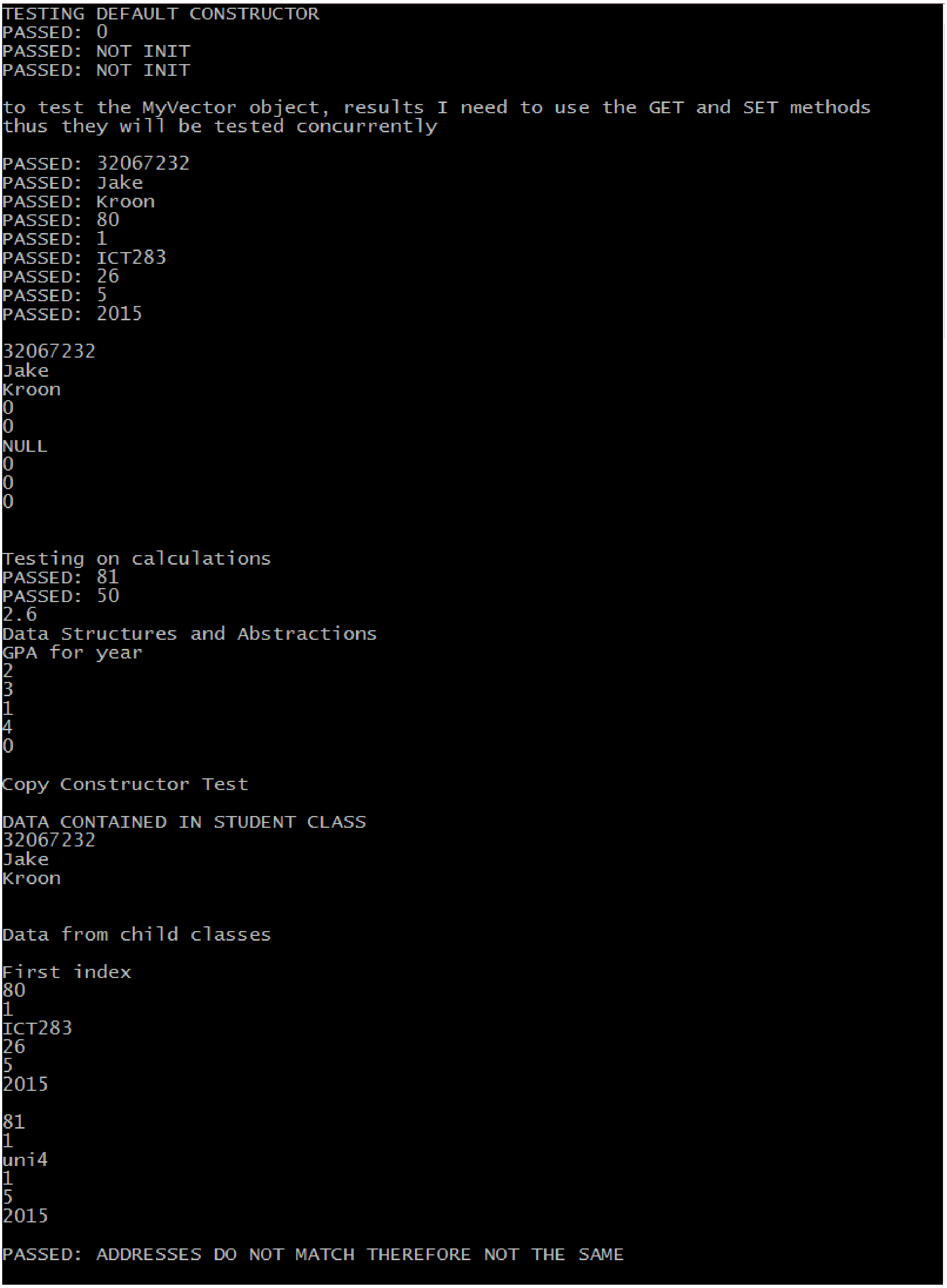


```
Pre Order
7
3
2
1
5
4
6
15
11
9
8
10
13
12
14
17
16
19
18
20

In Order
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```



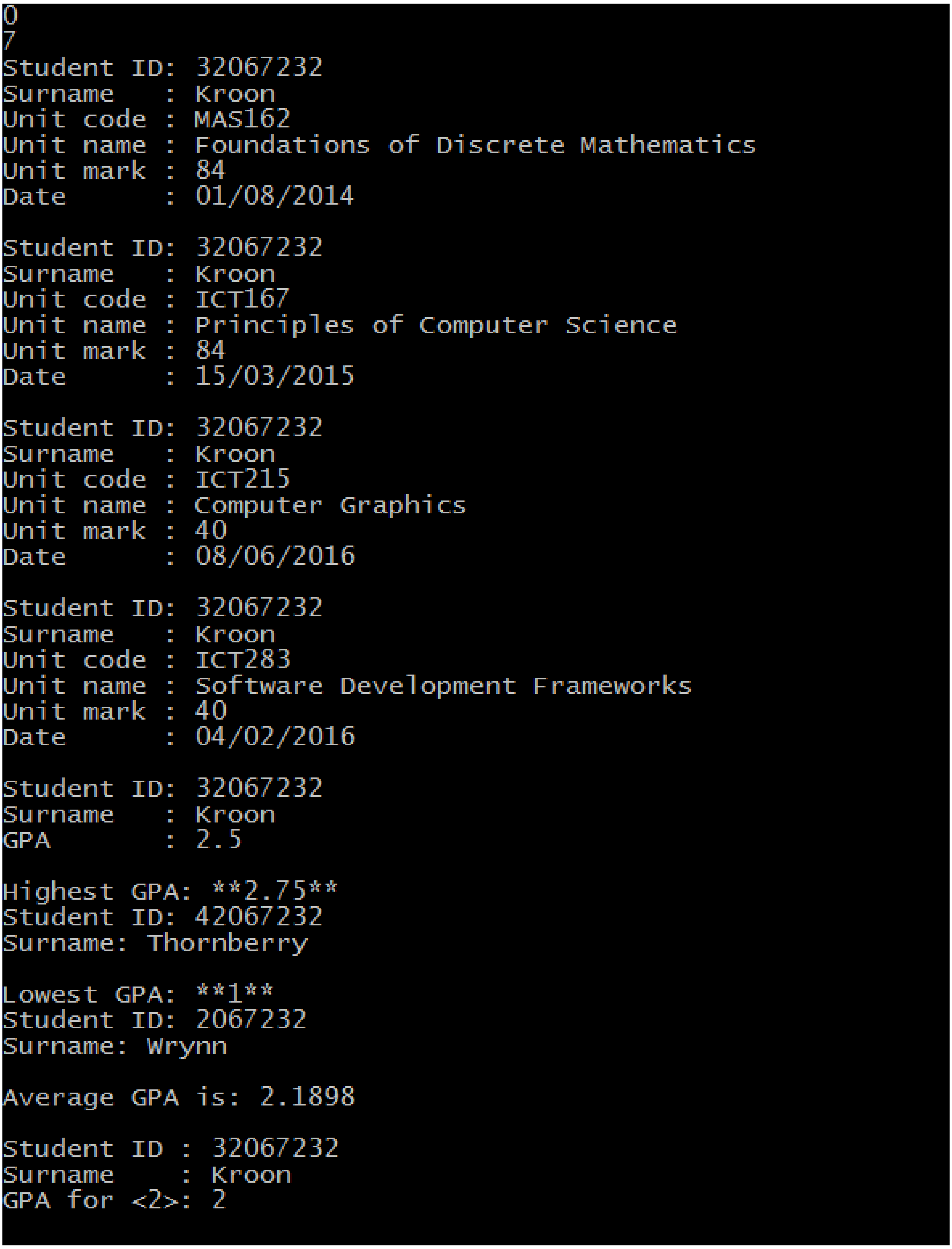
```
Copy Tree PreOrder
7
3
2
1
5
4
6
15
11
9
8
10
13
12
14
17
16
19
18
20
Copy Tree InOrder
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Passed
Passed
1
```

Student



```
TESTING DEFAULT CONSTRUCTOR
PASSED: 0
PASSED: NOT INIT
PASSED: NOT INIT

to test the MyVector object, results I need to use the GET and SET methods
thus they will be tested concurrently

PASSED: 32067232
PASSED: Jake
PASSED: Kroon
PASSED: 80
PASSED: 1
PASSED: ICT283
PASSED: 26
PASSED: 5
PASSED: 2015

32067232
Jake
Kroon
0
0
NULL
0
0
0


Testing on calculations
PASSED: 81
PASSED: 50
2.6
Data Structures and Abstractions
GPA for year
2
3
1
4
0

Copy Constructor Test

DATA CONTAINED IN STUDENT CLASS
32067232
Jake
Kroon


Data from child classes

First index
80
1
ICT283
26
5
2015

81
1
uni4
1
5
2015

PASSED: ADDRESSES DO NOT MATCH THEREFORE NOT THE SAME
```

StudentIO



```
0
7
Student ID: 32067232
Surname    : Kroon
Unit code : MAS162
Unit name : Foundations of Discrete Mathematics
Unit mark : 84
Date       : 01/08/2014

Student ID: 32067232
Surname    : Kroon
Unit code : ICT167
Unit name : Principles of Computer Science
Unit mark : 84
Date       : 15/03/2015

Student ID: 32067232
Surname    : Kroon
Unit code : ICT215
Unit name : Computer Graphics
Unit mark : 40
Date       : 08/06/2016

Student ID: 32067232
Surname    : Kroon
Unit code : ICT283
Unit name : Software Development Frameworks
Unit mark : 40
Date       : 04/02/2016

Student ID: 32067232
Surname    : Kroon
GPA        : 2.5

Highest GPA: **2.75**
Student ID: 42067232
Surname: Thornberry

Lowest GPA: **1**
Student ID: 2067232
Surname: Wrynn

Average GPA is: 2.1898

Student ID : 32067232
Surname     : Kroon
GPA for <2>: 2
```

CSV file contents – yes all the spaces and new lines are in the file, this is an accurate

representation.

```
32067232,Jake,Kroon,1,ICT102,Introduction to Computer Science,3,79,1/01/2013
32067232,Jake,Kroon,1,ICT103,Introduction to Data Communications,3,67,1/01/2013
32067232,Jake,Kroon,2,ICT107,Principles of Information Systems,3,66,1/06/2013
32067232,Jake,Kroon,2,BSC100,Building Blocks for Science Students,3,72,1/06/2013
32067232,Jake,Kroon,1,ICT171,Introduction to Server Environments and
Architectures,3,77,1/02/2014
32067232,Jake,Kroon,1,JPN101,Japanese 1,3,66,1/02/2014
32067232,Jake,Kroon,2,ICT284,Systems Analysis and Design,3,82,8/06/2016
32067232,Jake,Kroon,2,ICT215,


    Computer Graphics       ,3,   40     ,8/06/2016
22067232,Binary,Tree,1,ICT211,Web Computing,4,70,15/03/2015
22067232,Binary,Tree,2,ICT218,Databases,4,73,16/06/2015
22067232,Binary,Tree,1,ICT209,Data Structures and Abstractions,3,60,4/02/2016
22067232,Binary,Tree,1,ICT283,Software Development Frameworks,4,40,4/02/2016






12067232,Anduin,Wrynn,1,ICT283,Software Development Frameworks,4,40,4/02/2016
12067232,Anduin,Wrynn,1,ICT167,Principles of Computer Science,3,84,15/03/2015
12067232,Anduin,Wrynn,1,ICT211,Web Computing,4,70,15/03/2015
12067232,Anduin,Wrynn,2,ICT218,Databases,4,73,16/06/2015
12067232,Anduin,Wrynn,1,ICT209,Data Structures and Abstractions,3,60,4/02/2016
12067232,Anduin,Wrynn,1,ICT283,Software Development Frameworks,4,40,4/02/2016
12067232,Anduin,Wrynn,2,ICT284,Systems Analysis and Design,3,82,8/06/2016
12067232,Anduin,Wrynn,2,ICT215,     Computer Graphics       ,3,   40     ,8/0/2016
42067232,Nigel,Thornberry,1,ICT102,Introduction to Computer Science,3,79,1/01/2013
42067232,Nigel,Thornberry,2,MAS162,Foundations of Discrete Mathematics,3,84,1/08/2014
42067232,Nigel,Thornberry,2,BSC150,What is Science?,3,74,1/08/2014
2067232,Anduin,Wrynn,2,ICT215,     Computer Graphics       ,3,   50     ,8/06/2017
32067232,Jake,Kroon,2,MAS162,
Foundations of Discrete Mathematics,3,84,1/08/2014
                    32067232,Jake,Kroon,2,BSC150,What is Science?,3,74,1/08/2014
32067232,Jake,Kroon,1,ICT167,Principles of Computer Science,3,84,15/03/2015
32067232,Jake,Kroon,1,ICT211,Web Computing,4,70,15/03/2015
32067232,Jake,Kroon,2,ICT218,Databases,4,73,16/06/2015
32067232,Jake,Kroon,1,ICT209,Data Structures and Abstractions,3,60,4/02/2016
32067232,Jake,Kroon,1,ICT283,Software Development Frameworks,4,40,4/02/2016
42067232,Nigel,Thornberry,1,ICT103,Introduction to Data Communications,3,67,1/01/2013
42067232,Nigel,Thornberry,2,ICT107,Principles of Information Systems,3,66,1/06/2013
42067232,Nigel,Thornberry,2,BSC100,Building Blocks for Science Students,3,72,1/06/2013
42067232,Nigel,Thornberry,1,ICT171,Introduction to Server Environments and
Architectures,3,77,1/02/2014
42067232,Nigel,Thornberry,1,JPN101,Japanese 1,3,66,1/02/2014
52067232,Jake,Kroon,2,MAS162,Foundations of Discrete Mathematics,3,84,1/08/2014
52067232,Phuong,Nguyen,2,BSC150,What is Science?,3,74,
```

1 /

08

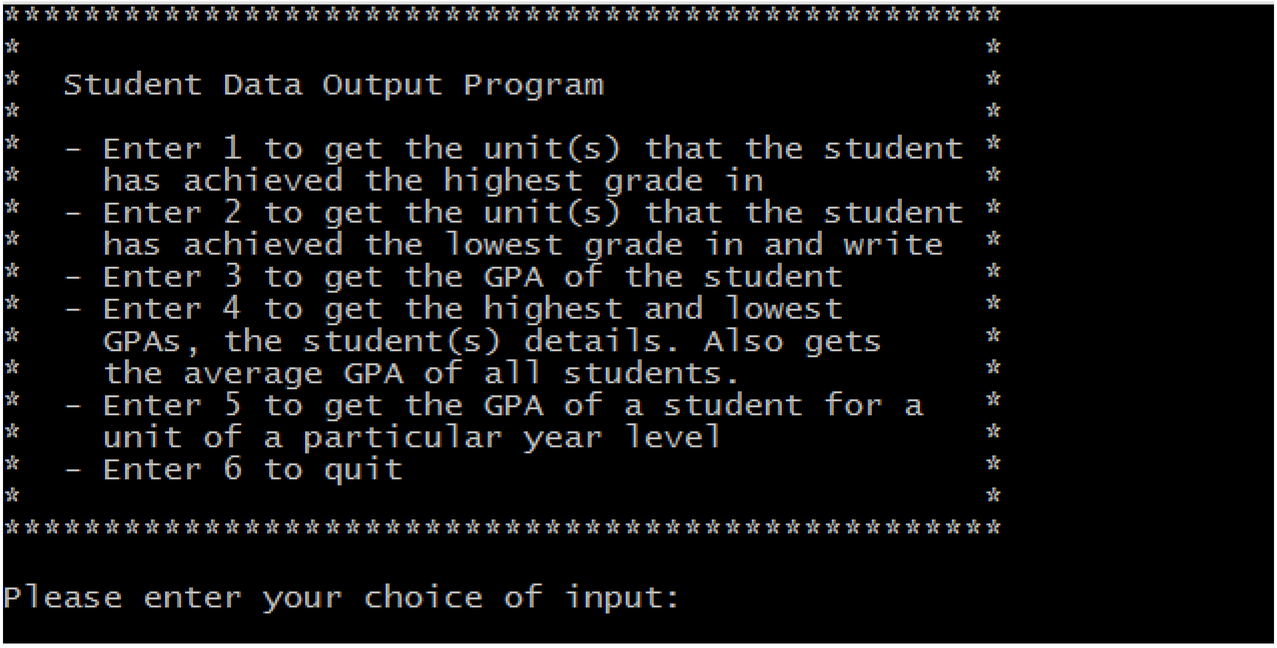
/2014

```
52067232,Phuong,Nguyen,1,ICT167,Principles of Computer Science,3,84,15/03/2015
52067232,Phuong,Nguyen,1,ICT211,Web Computing,4,70,15/03/2015
52067232,Phuong,Nguyen,1,ICT209,Data Structures and Abstractions,3,60,4/02/2016
52067232,Phuong,Nguyen,1,ICT283,Software Development Frameworks,4,40,4/02/2016
62067232,Jonsi,Birgisson,1,ICT102,Introduction to Computer Science,3,79,1/01/2013
62067232,Jonsi,Birgisson,1,ICT103,Introduction to Data Communications,3,67,1/01/2013
62067232,Jonsi,Birgisson,2,ICT107,Principles of Information Systems,3,66,1/06/2013
62067232,Jonsi,Birgisson,2,BSC100,Building Blocks for Science Students,3,72,1/06/2013
62067232,Jonsi,Birgisson,1,ICT171,Introduction to Server Environments and
Architectures,3,77,1/02/2014
```

I used the above input for my program and feel that it sufficiently tests all aspects of it. I have 6 students of varying amounts of studies.

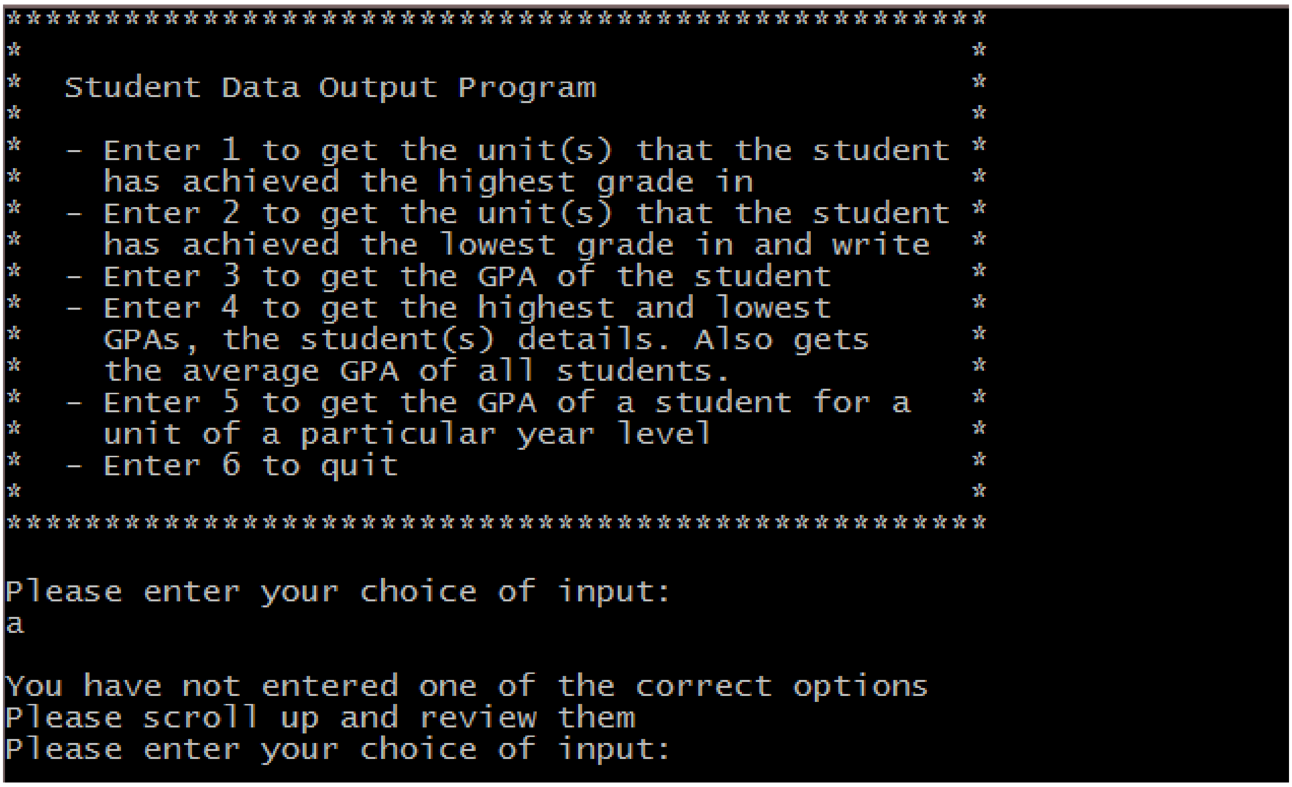
Sample run of program:

Menu

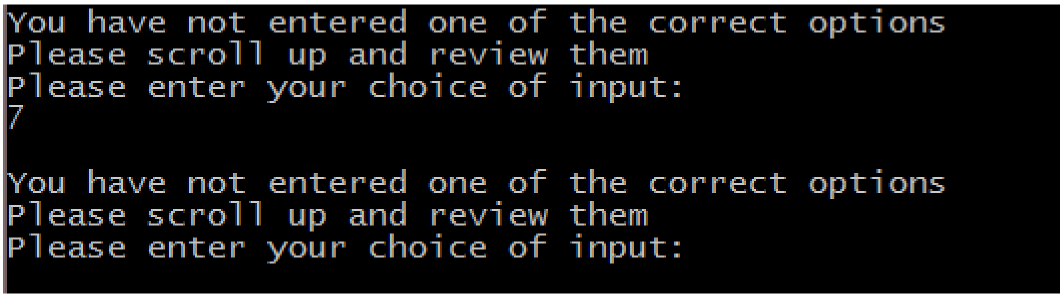


```
***************************************************
*                                                 *
*   Student Data Output Program                   *
*                                                 *
*   - Enter 1 to get the unit(s) that the student *
*     has achieved the highest grade in           *
*   - Enter 2 to get the unit(s) that the student *
*     has achieved the lowest grade in and write  *
*   - Enter 3 to get the GPA of the student       *
*   - Enter 4 to get the highest and lowest       *
*     GPAs, the student(s) details. Also gets     *
*     the average GPA of all students.            *
*   - Enter 5 to get the GPA of a student for a   *
*     unit of a particular year level             *
*   - Enter 6 to quit                             *
*                                                 *
***************************************************

Please enter your choice of input:
```

Bad Input

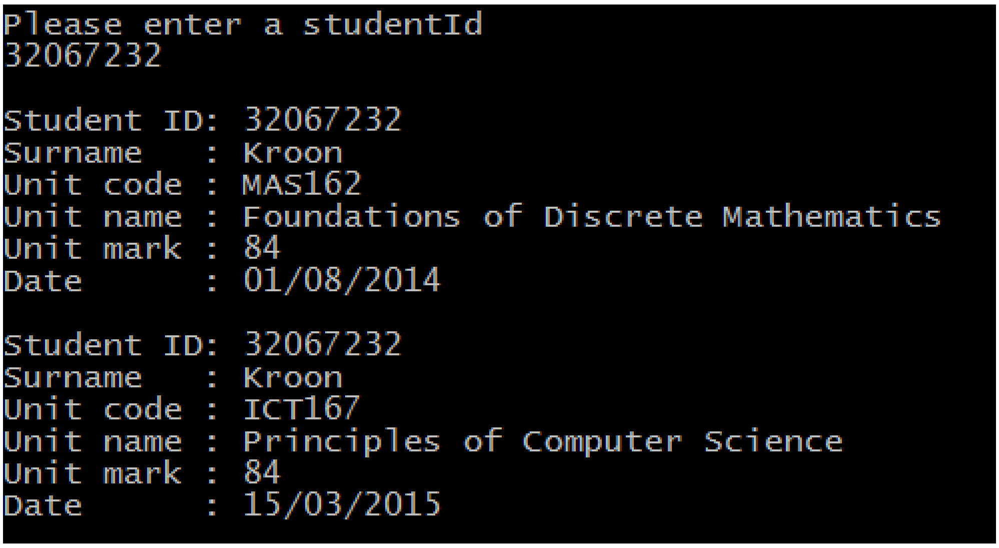


```
***************************************************
*                                                 *
*   Student Data Output Program                   *
*                                                 *
*   - Enter 1 to get the unit(s) that the student *
*     has achieved the highest grade in           *
*   - Enter 2 to get the unit(s) that the student *
*     has achieved the lowest grade in and write  *
*   - Enter 3 to get the GPA of the student       *
*   - Enter 4 to get the highest and lowest       *
*     GPAs, the student(s) details. Also gets     *
*     the average GPA of all students.            *
*   - Enter 5 to get the GPA of a student for a   *
*     unit of a particular year level             *
*   - Enter 6 to quit                             *
*                                                 *
***************************************************

Please enter your choice of input:
a

You have not entered one of the correct options
Please scroll up and review them
Please enter your choice of input:
```

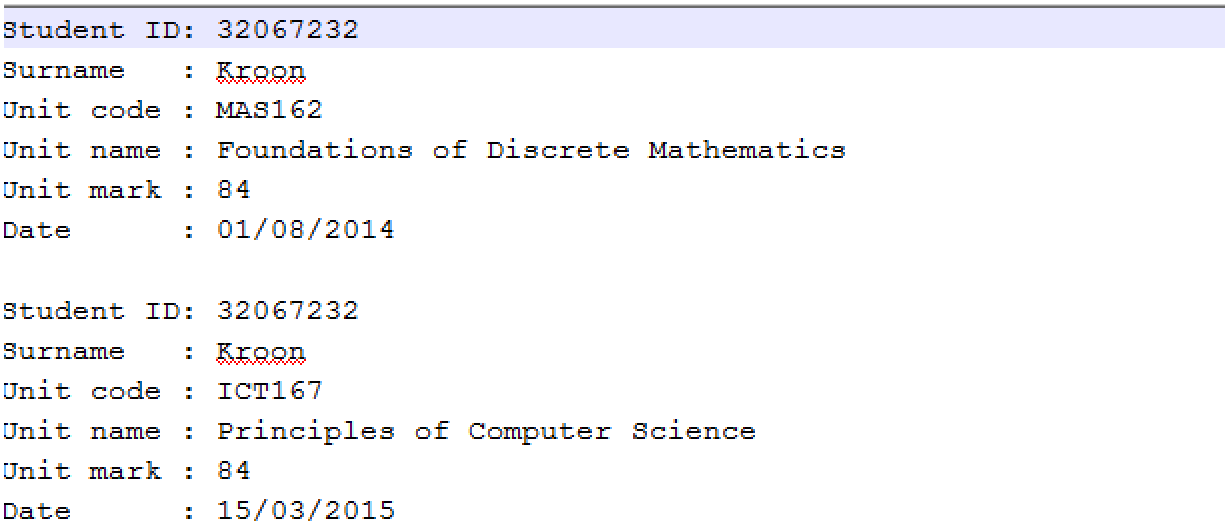


```
You have not entered one of the correct options
Please scroll up and review them
Please enter your choice of input:
7

You have not entered one of the correct options
Please scroll up and review them
Please enter your choice of input:
```

Option 1



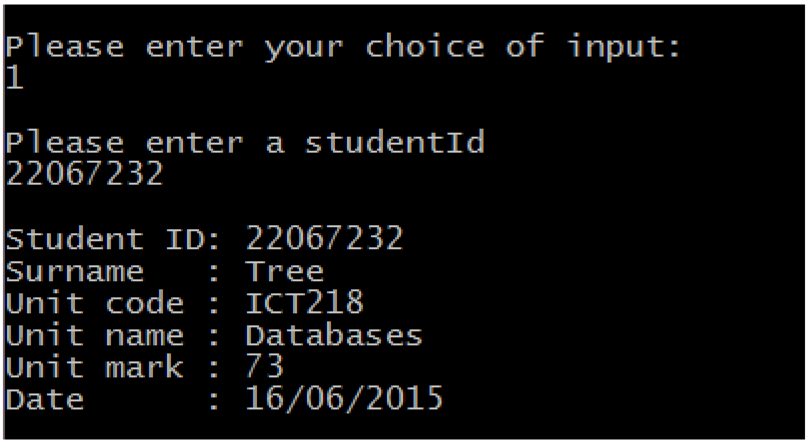
```
Please enter a studentId
32067232

Student ID: 32067232
Surname    : Kroon
Unit code : MAS162
Unit name : Foundations of Discrete Mathematics
Unit mark : 84
Date       : 01/08/2014

Student ID: 32067232
Surname    : Kroon
Unit code : ICT167
Unit name : Principles of Computer Science
Unit mark : 84
Date       : 15/03/2015
```



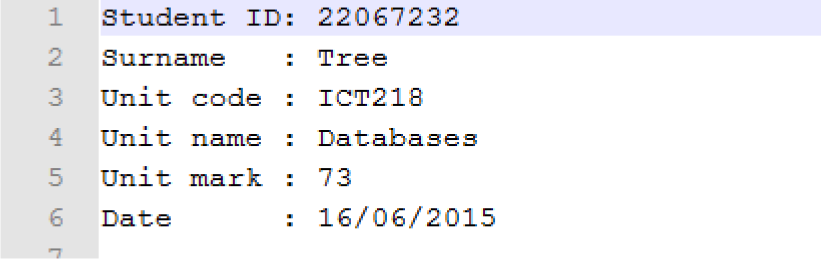
```
Student ID: 32067232
Surname    : Kroon
Unit code : MAS162
Unit name : Foundations of Discrete Mathematics
Unit mark : 84
Date       : 01/08/2014

Student ID: 32067232
Surname    : Kroon
Unit code : ICT167
Unit name : Principles of Computer Science
Unit mark : 84
Date       : 15/03/2015
```

The above screenshots demonstrate that for the student with the id of 32067232 option 1 functions as expected

Option 1



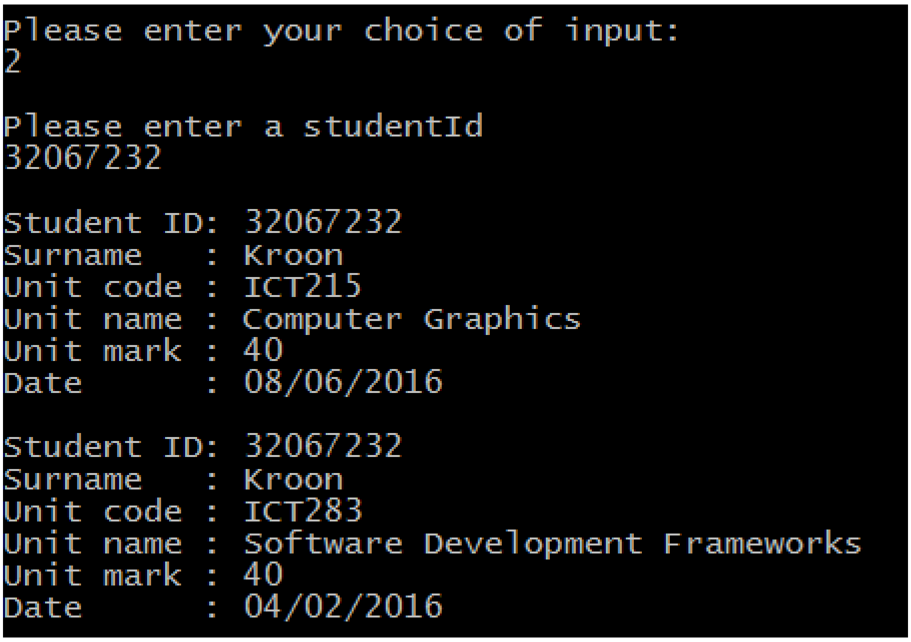
```
Please enter your choice of input:
1

Please enter a studentId
22067232

Student ID: 22067232
Surname    : Tree
Unit code : ICT218
Unit name : Databases
Unit mark : 73
Date       : 16/06/2015
```



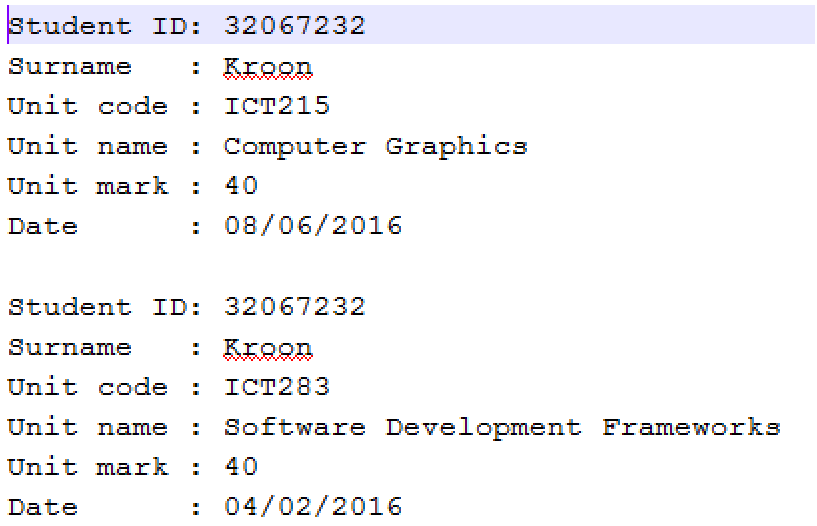
```
Student ID: 22067232
Surname    : Tree
Unit code : ICT218
Unit name : Databases
Unit mark : 73
Date       : 16/06/2015
```

The above screenshots demonstrate that this works for more than one student functioning as expected.

Option 2



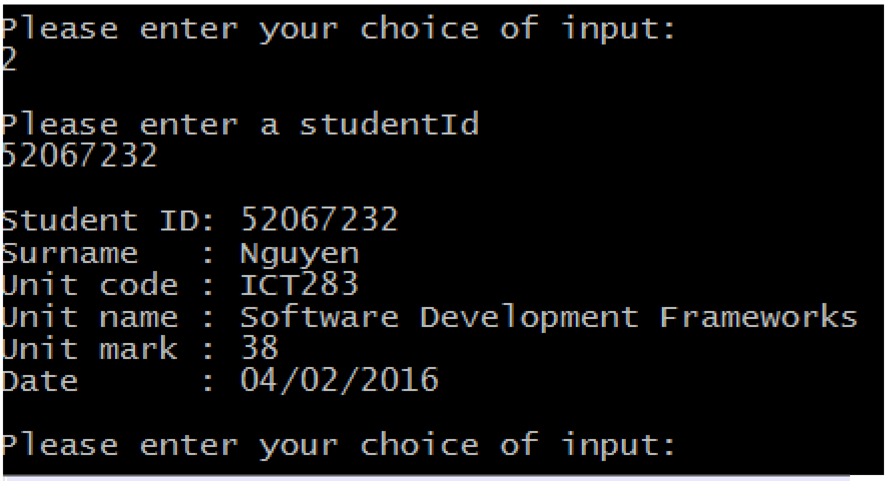
```
Please enter your choice of input:
2

Please enter a studentId
32067232

Student ID: 32067232
Surname    : Kroon
Unit code : ICT215
Unit name : Computer Graphics
Unit mark : 40
Date       : 08/06/2016

Student ID: 32067232
Surname    : Kroon
Unit code : ICT283
Unit name : Software Development Frameworks
Unit mark : 40
Date       : 04/02/2016
```



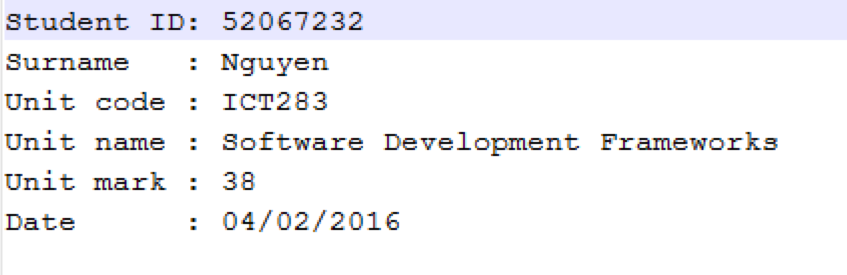
```
Student ID: 32067232
Surname    : Kroon
Unit code : ICT215
Unit name : Computer Graphics
Unit mark : 40
Date       : 08/06/2016

Student ID: 32067232
Surname    : Kroon
Unit code : ICT283
Unit name : Software Development Frameworks
Unit mark : 40
Date       : 04/02/2016
```

The above screenshots show that for the student with the id 32067232 option 2 functions as expected

Option 2



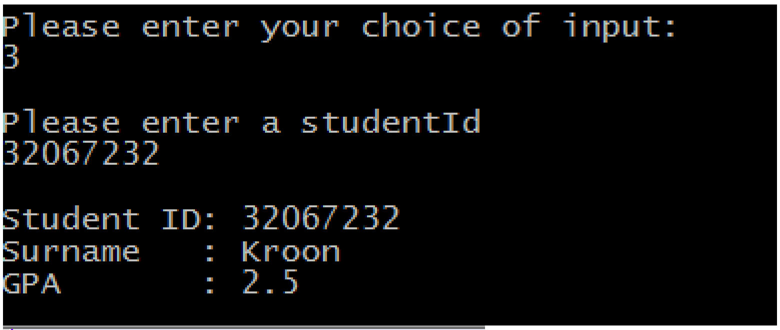
```
Please enter your choice of input:
2

Please enter a studentId
52067232

Student ID: 52067232
Surname    : Nguyen
Unit code : ICT283
Unit name : Software Development Frameworks
Unit mark : 38
Date       : 04/02/2016

Please enter your choice of input:
```



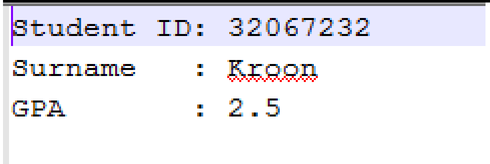
```
Student ID: 52067232
Surname    : Nguyen
Unit code : ICT283
Unit name : Software Development Frameworks
Unit mark : 38
Date       : 04/02/2016
```

The above screenshots demonstrate that option 2 will function correctly for more than one student

Option 3

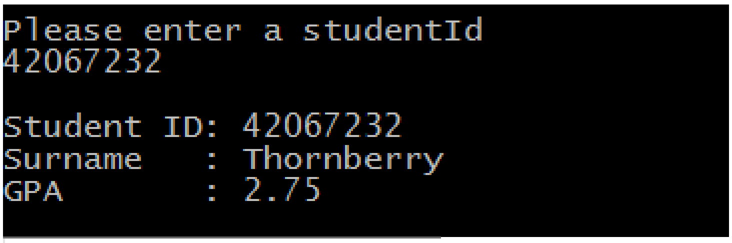


```
Please enter your choice of input:
3

Please enter a studentId
32067232

Student ID: 32067232
Surname    : Kroon
GPA        : 2.5
```

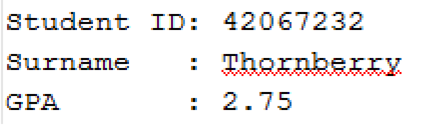


```
Student ID: 32067232
Surname    : Kroon
GPA        : 2.5
```

The above screenshots demonstrate that option 3 functions as expected for the student with the id of 32067232



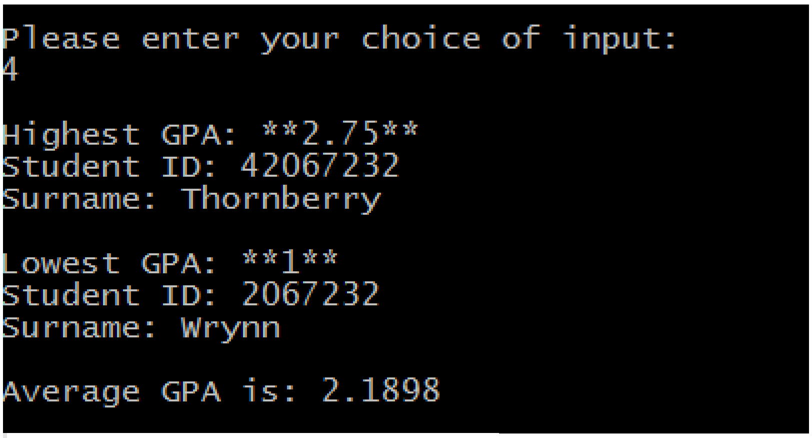
```
Please enter a studentId
42067232

Student ID: 42067232
Surname    : Thornberry
GPA        : 2.75
```



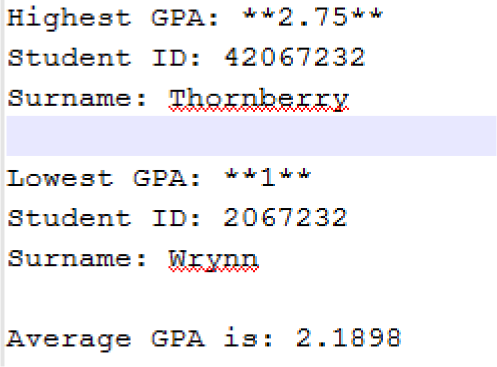
```
Student ID: 42067232
Surname    : Thornberry
GPA        : 2.75
```

The above screenshots demonstrate that option 3 functions as expected for more than one student

Option 4



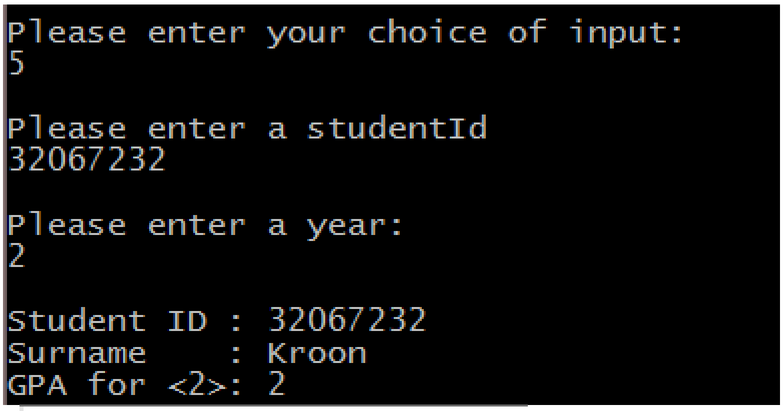
```
Please enter your choice of input:
4

Highest GPA: **2.75**
Student ID: 42067232
Surname: Thornberry

Lowest GPA: **1**
Student ID: 2067232
Surname: Wrynn

Average GPA is: 2.1898
```



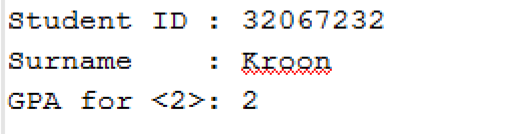
```
Highest GPA: **2.75**
Student ID: 42067232
Surname: Thornberry

Lowest GPA: **1**
Student ID: 2067232
Surname: Wrynn

Average GPA is: 2.1898
```

The above screenshots demonstrate that option 4 functions as expected

Option 5

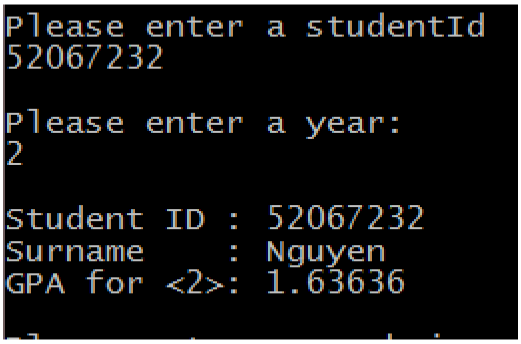


```
Please enter your choice of input:
5

Please enter a studentId
32067232

Please enter a year:
2

Student ID : 32067232
Surname    : Kroon
GPA for <2>: 2
```

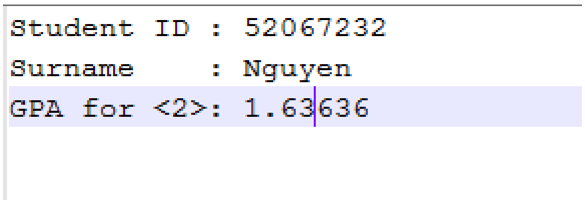


```
Student ID : 32067232
Surname    : Kroon
GPA for <2>: 2
```

The above screenshot demonstrates that for the student with the id of 32067232 this function works as expected.

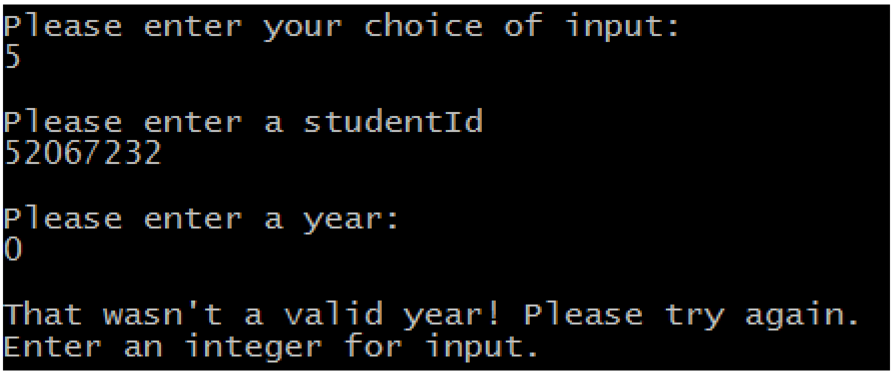


```
Please enter a studentId
52067232

Please enter a year:
2

Student ID : 52067232
Surname    : Nguyen
GPA for <2>: 1.63636
```



```
Student ID : 52067232
Surname    : Nguyen
GPA for <2>: 1.63636
```

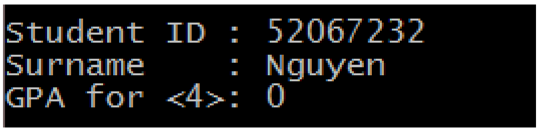
The above screenshot demonstrates that for the student with this function works for more than one student as expected.



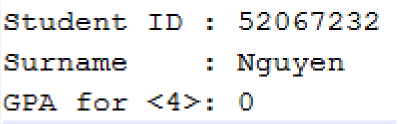
```
Please enter your choice of input:
5

Please enter a studentId
52067232

Please enter a year:
0

That wasn't a valid year! Please try again.
Enter an integer for input.
```

The above screenshot demonstrates that the year must be a valid year (i.e. integer > 0)

Option 5



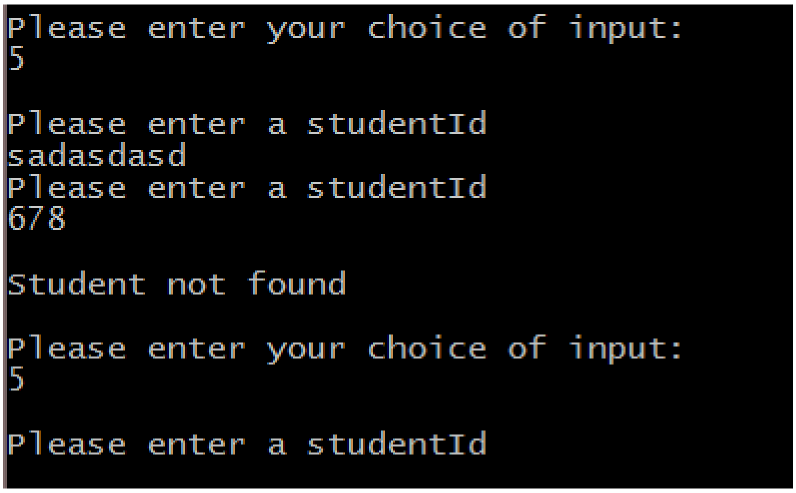
```
Student ID : 52067232
Surname    : Nguyen
GPA for <4>: 0
```



```
Student ID : 52067232
Surname    : Nguyen
GPA for <4>: 0
```

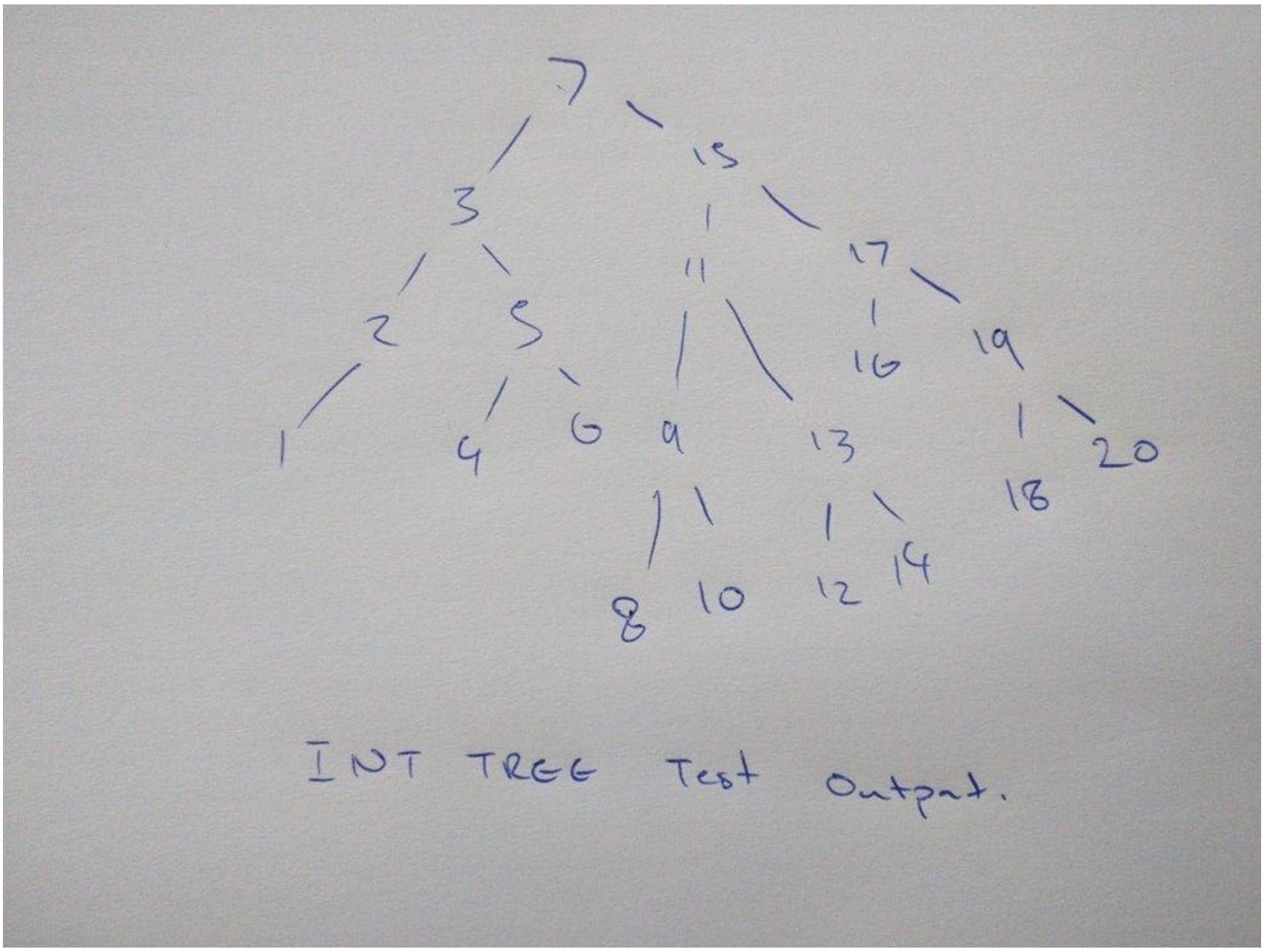
The above screenshots demonstrate that if the student hasn't completed any units in the year entered, the GPA value will be returned as 0.

On enter 6 the program closes as expected.



```
Please enter your choice of input:
5

Please enter a studentId
sadasdasd
Please enter a studentId
678

Student not found

Please enter your choice of input:
5

Please enter a studentId
```

The above screenshot demonstrates that if a string if non numeric characters are entered the user must enter another valid student Id, and if the integer value is entered but not found to be a valid student id, the student must re enter their choice of input as they may have not chosen the correct option.



The above hand drawn diagram shows that the Binary Tree balances correctly due to the automatic balancing on input feature of an AVL tree.