0 TABLE OF CONTENTS

1 STUDENT.JAVA

1.1 STATIC TESTING

1.1.1 STEPS TAKEN AND THE FAULTS IDENTIFIED

TEST	STEPS TAKEN	FAULTS IDENTIFIED
Student.java variables	1. Check all variables not in methods	1. ModuleRecord array had an error whereby rather than declaring '13' as the size the constant 'MAX_NUMBER_OF_MODULES' should've been used
Student method	1. I checked to ensure the `studentName` is equal to `name` 2. I checked to ensure the `studentNumber` is equal to `number`	1. None 2. None
addModule method	1. I checked the parameter to ensure it takes in a module object	1. None
removeModule method	1. I checked the parameter to ensure take in a module object	1. None
checkModuleSelection method	1. Looked at all variables to check they were relevant 2. I looked at the expressions in the for loop to check out-of-the-ordinary code 3. I then took a look at line 52 (next line) 4. I continued to take a look until another issue at line 56 5. Finally, I went through the return statement and spotted 3 faults on line 70 and 1 fault on line 71	1. I noticed Boolean projectModule is set to false when it doesn't need to be as by default it is false 2. A fault was spotted in the termination expression, rather than "<" it should be "<=" **NOTE:** At the time of this testing phase I thought this was the correct solution, I later realised it wasn't and had to debug it 3. Every time the for loop goes around it's printing only semester 1 credits and it's in the wrong location – It should be placed at the end of the if statements. 2 other print statements should be added to print sem2 and sem3 credits. 4. The fault is it's getting only semester 1's credits rather than the total credits 5. Line 70: First: There is a "!" at the beginning of the return statement Second: projectModule shouldn't have a "!" in front of it as it'll flip the value of what it should be Third: Before MAXIMUM_LEVEL_6_CREDITS the symbol should be "<=" rather than ">"

		Line 71 Before VALID_NUMBER_OF_REGISTERED_CREDITS rather than "!=" it should be "="
getName method	1. Checked to ensure it is returning something appropriate	1. None
recordMark method	1. Checked to ensure parameters are correct 2. Check it's correctly recording the mark	1. None 2. recordMark had been set to 'i', rather than 'mark' – Therefore marks weren't correctly being stored
getAverage	1. Checked to ensure for loop visits every module 2. Ensure calculations are correct	1. None 2. There was an error with rounding, adding the 0.5 should be in the numerator and not the denominator
qualification method	1. Check the if statements conditions 2. Check the return of each if statement 3. Check the else clause is returning an appropriate value	1. None 2. Two faults were identified Firstly on line 115 it should return "MSC_WITH_DISTINCTION" rather than "MSC_WITH_MERIT" Secondly, line 118 should return "MSC_WITH_MERIT" rather than "MSC_WITH_DISTINCTION" 3. None
findModuleIndex method	1. Check parameter is correct 2. Check for loop 3. Check calculation 4. Check return value	1. None 2. For the termination expression rather than "<=" it should be "<" 3. None 4. None
findFreeLocation method	1. Check for loop 2. Check return value	1. None 2. None
checkAllModulesPassed	1. Check for loop 2. Check return value 3. Check comment to ensure method is doing what it's supposed to	1. None 2. None 3. Whilst it's doing what it's supposed to the comment says 'private' implying the method should be private rather than public

1.2 JUNIT TESTING

1.2.1 STEPS TAKEN

TEST	STEPS TAKEN	FAULTS IDENTIFIED
getName Test	1. Create student object with name variable associated to "John Smith" 2. Test s.getName is equal to name variable	1. None 2. None

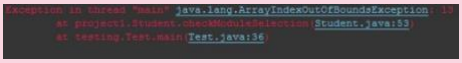
1.3 WHITE BOX TESTING – TEST CLASS

1.3.1 STEPS TAKEN

TEST	STEPS TAKEN	FAULTS IDENTIFIED
Student Constructor	1. Test constructor whether it accepts a String and an Int 2. Test constructor under normal circumstances – i.e. input 2 strings	1. None, it ran as expected – Resulted in "compilation problem" identifying the constructor was 'undefined' 2. None, no errors or faults occurred while compiling
addModule method & findFreeLocaiton method	1. Test without a Module object 2. Test with Module object	1. None, met expected outcome whereby error states "m cannot be resolved to a variable" 2. None, no errors or faults occurred while compiling
removeModule method	1. Test without a Module object 2. Test with a Module object	1. None, met expected outcome whereby error states "m cannot be resolved to a variable" 2. None, no errors or faults occurred while compiling
checkModuleSelection method	1. Test with the student not having any modules 2. (After Cause Elimination Debug) Rerun step 1 3. Test with the student having 13 modules	1. Yes:  2. None, no errors or faults while compiling 3. None, no errors or faults while compiling
recordMark method & FindModuleIndex method	1. Test by setting Student 1's Module 0 mark to 86	1. None, no errors or faults while compiling
getAverage method	1. Test by setting all Student 2's marks and running the method	1. None, no errors or faults while compiling
getQualification method & checkAllModules Passed method	1. Test by running it on Student 2	1. None, no errors or faults while compiling

1.4 DEBUGGING – CAUSE ELIMINATION

TEST	STEPS TAKEN	FAULTS IDENTIFIED

checkModuleSelection method	1. I previously produced the error, from this I decided to change the int i value from 1 to 0 and remove previous ammendments (i.e. from the static testing)	 1. Issue was resolved

2 MODULE.JAVA

2.1 STATIC TESTING

2.1.1 STEPS TAKEN AND THE FAULTS IDENTIFIED

TEST	STEPS TAKEN	FAULTS IDENTIFIED
Module constructor	1. Checked parameters 2. Check each variable equals the correct variable	1. None 2. "sem2Credits = sem1;" should equal "sem2"
getCode method	1. Checked what the method returned	1. None
getTitle method	1. Checked what the method returned	1. None
getSemesterOneCredits	1. Checked what the method returned	1. None
getSemesterTwoCredits	1. Checked what the method returned	1. None
getSemesterThreeCredits	1. Checked what the method returned	1. None
getTotalCredits	1. Check the calculation was correct for return	1. The calculation only calculated semester 2 and 3 credits – Not taking semester 1 into consideration. The calculation must be changed to: "sem1Credits + sem2Credits + sem2Credits"
getLevel	1. Checked what the method returned	1. None

2.2 JUNIT TESTING

2.2.1 STEPS TAKEN

TEST	STEPS TAKEN	FAULTS IDENTIFIED
getCode method	1. Test to check global object 'm' has the correct code	1. None
getTitle method	1. Test to check global object 'm' has the correct title	1. None
getSemesterOneCredits method	1. Test to check global object 'm' has the correct credits	1. None
getSemesterTwoCredits method	1. Test to check global object 'm' has the correct credits	1. None
getSemesterThreeCredits method	1. Test to check global object 'm' has the correct credits	1. None

getTotalCredits	1. Test to ensure calculation of total credits was functional	1. None
getLevel	1. Test to check global object 'm' has the correct level associated to it	1. None

2.3 WHITE BOX TESTING

2.3.1 STEPS TAKEN

TEST	STEPS TAKEN	FAULTS IDENTIFIED
Module Constructor	1. Create another module	1. None, no errors or faults while compiling

3 MODULERECORD.JAVA

3.1 STATIC TESTING

3.1.1 STEPS TAKEN AND THE FAULTS IDENTIFIED

TEST	STEPS TAKEN	FAULTS IDENTIFIED
ModuleRecord constructor (Line 9)	2. Checked parameters 3. Checked each variable equals the correct variable	2. None 3. None
Module Record constructor (Line 15)	1. Checked parameters 2. Checked the variable equals the correct variable	1. None 2. None
recordMark method	1. Checked parameter 2. Checked calculation	1. None 2. None
getModuleMark method	1. Checked what the method is returning	1. None
getModuleResult method	1. Checked if statement conditions 2. Check return for if statement 3. Check return statement of else	1. None 2. None 3. None
getModule	1. Checked what the method returned	1. None
Global variables	2. Checked Module m to see whether it is appropriate 3. Checked mark to see whether it is appropriate	2. None 3. Although it works as it is it could be set to private rather than public

3.2 JUNIT TESTING

3.2.1 STEPS TAKEN

TEST	STEPS TAKEN	FAULTS IDENTIFIED
getModuleMark method	1. Using assertEquals to check if object 'mr2' has the module mark of 10	1. None
recordMark method	1. Using assertEquals to check if object 'mr' has a module mark of 100 which was set using the recordMark method	1. None
getModuleResult1	1. Using assertEquals to check if object 'mr' has a module result of pass by setting the mark using the recordMark method	1. None

getModuleResult2	1. Using assertEquals to check if object 'mr2' has a module result of a fail by setting the mark using the constructor	1. None
getModule	1. Using assertEquals to check if object 'mr' (module records) has the 'm' (module) object	1. None

3.3 WHITE BOX TESTING

3.3.1 STEPS TAKEN

TEST	STEPS TAKEN	FAULTS IDENTIFIED
ModuleRecordConstructor[1]	1. Create a module record for Module 0	1. None, no errors or faults while compiling
ModuleRecordConstructor[2]	1. Create a module record for Module 1	1. None, no errors or faults while compiling

4 VISUALISATION_OUT.JAR

4.1 BLACK BOX TESTING

FAULTS IDENTIFIED	
1. When printing the module and it's result it must only have a print statement that prints the first module and not the rest which is repeated until the maximum amount of modules	2. The graph produced doesn't have labels on it so it's hard to understand which module got what score – So I can't tell if it meets the spec "bars should be sorted by module code"
3. Qualification doesn't properly calculate whether the student has passed or failed the qualification – i.e. if all modules are passed they still fail the qualification	4. Mark isn't calculated properly, the highest value can go to is 60

5 OPINION ON STRATEGIES

- Overall I think my strategy was effective:
    1. I started with Static Tests of all methods to ensure there weren't any visible or obvious errors – Ultimately saving time later on
    2. Then I completed a Junit test on most of each classes methods to check the expected outcome meets the actual outcome
    3. I continued with testing by testing the rest of the methods in each class using the Test class to ensure no errors or mistakes are occurring
    4. Finally, I completed a Black Box test that would test all classes together (including the library)
- Despite some tests revealing little-to-none errors I would still consider this strategy a success as each class works on their own independently and those tests that revealed no errors just proves that this code is functional
- Testing methods multiple times via a Static Test then a Dynamic Test may not be time-efficient but it certainly proved to effective as it identified a fault with a correction I previously made
- Completing all these tests have identified that overall the program doesn't meet the specifications due to the library – Had the library had no issues also then it would most certainly meet the spec