**Programming I Project 1 Documentation**

Introduction: first draft

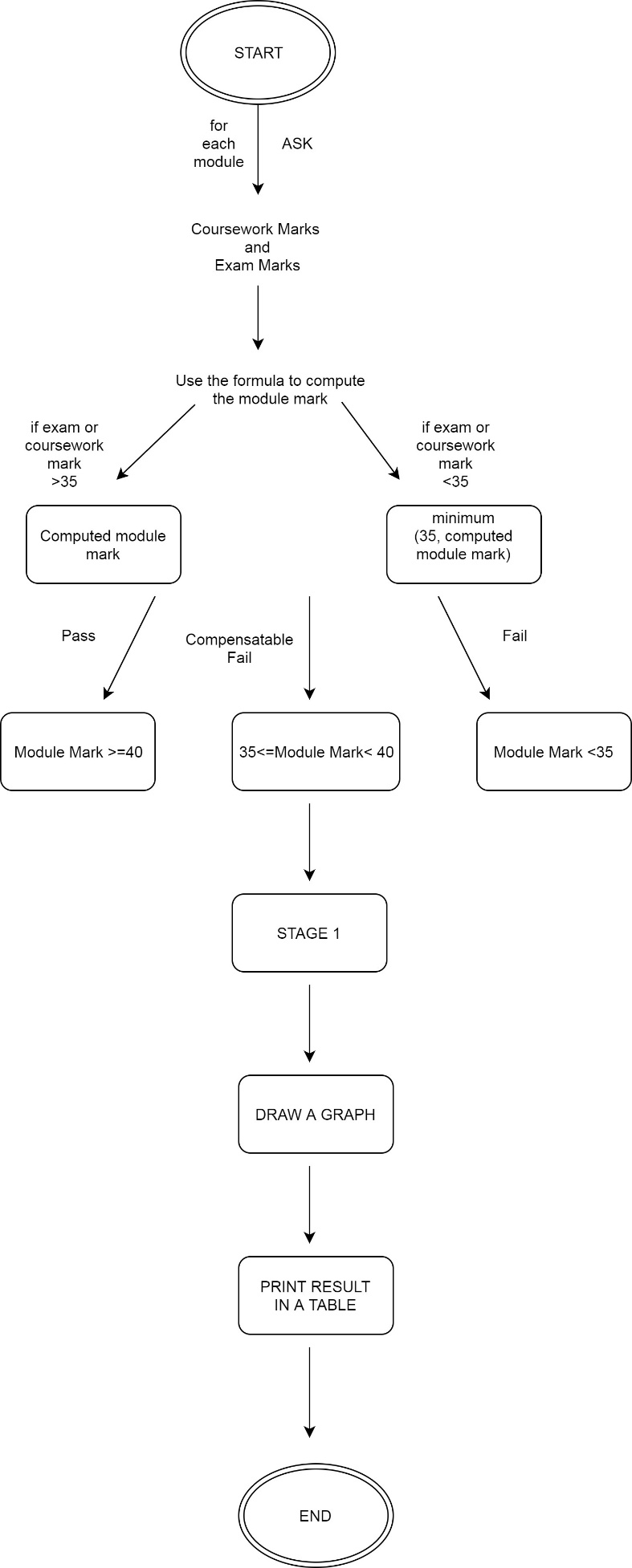
In computer science, the reason for creating programs is to give users useful and easy to use tools in order to allow them to find the faster way to get what they need.

While creating my program I came through difficulties and challenges.

In this documentation, I will outline the process I went through to solve these problems.

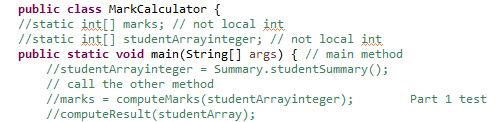
At first, I encountered a problem while organizing the structure of the program due to the fact that we had an enormous freedom in interpreting the text and creating and organizing the tasks.

So I tried to draw a scheme of how the program should have been designed.



Purpose of the program

After a first draft, I started to write some lines for the software that elaborate Stage 1 student marks. It is still possible to read an early version of the code for Part 1 in the MarkCalculator class. The tasks were divided into two parts: part one that gave the bases of the program and a second part was more focused on how to improve the user’s experience.



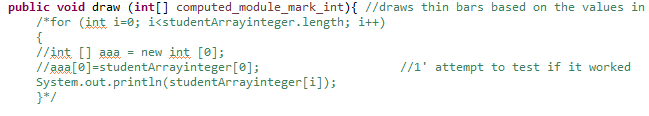
In this class, given students exam and coursework marks, the program calculates the mark for each module.

Students take six modules in Stage 1: CSC1021 Programming I, CSC1022 Programming II, CSC1023 The Software Engineering Professional which has coursework only, CSC1024 Computer Architecture, CSC1025 Mathematics for Computer Science, CSC1026 Website Design and Construction.

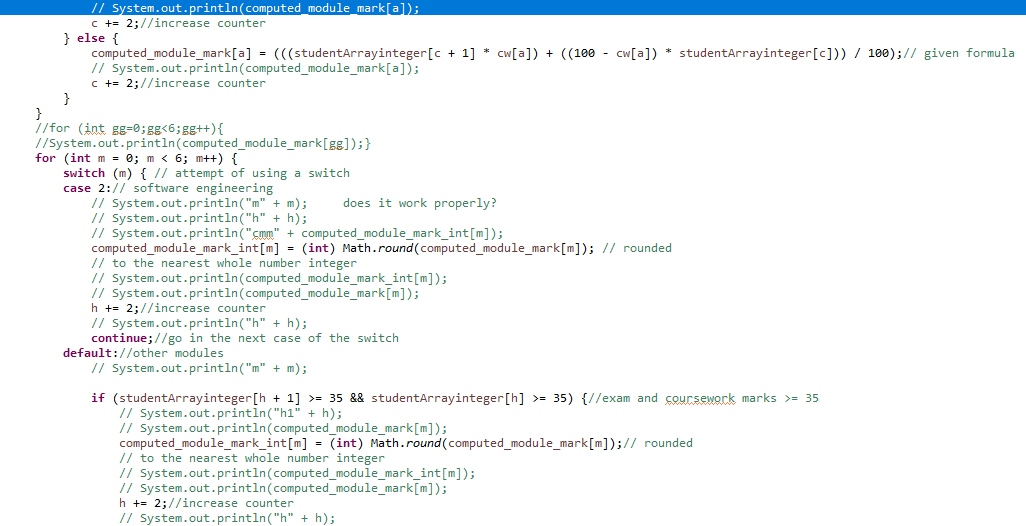
As written in the program specification, each module mark is calculated as follows:

((coursework mark \* coursework weighting) + (examination mark \* (100 - coursework weighting))) / 100.

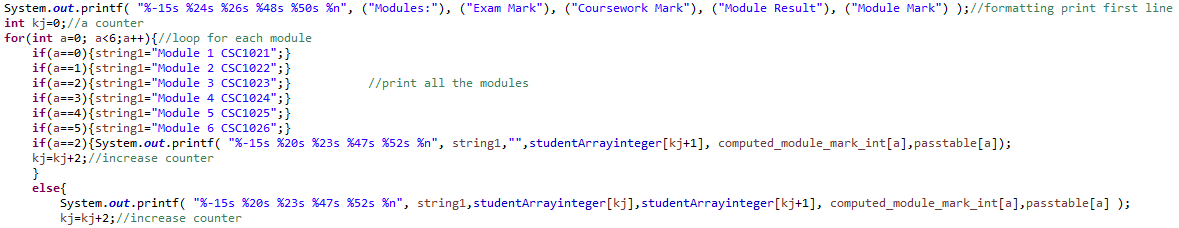
As long as I needed to use a considerable number of ‘for loops’, I tested it to verify there was no error in calculations.



A good way of testing is to put a ‘print statement’ after the code and follow the printed data until you find where the error is if the result is not the one expected.

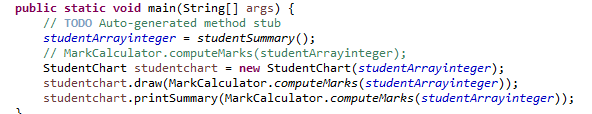


After a few attempts to know if the compiled program was correct, the result was a formatted table with all the exact values.



Function and usability

The java software asks users to insert for each module exam and coursework mark to use these inputs in other methods.



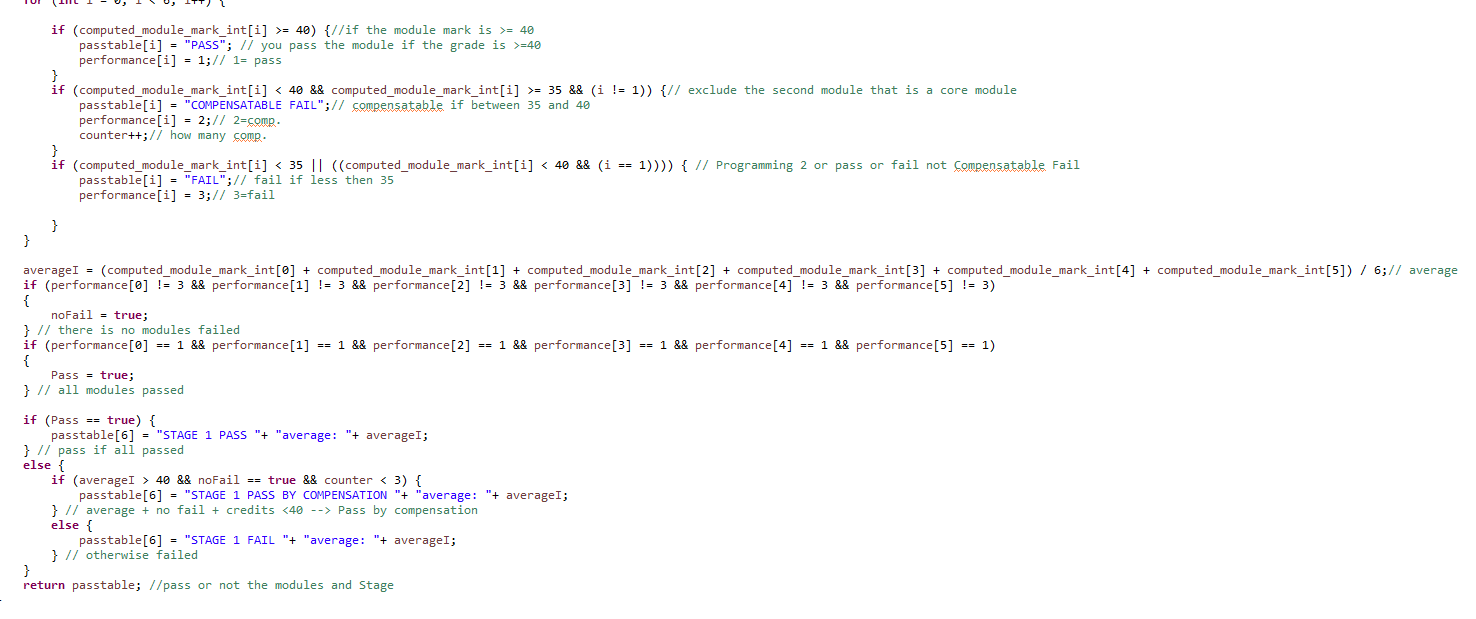
Inserting random values and calculating the module mark with the given formula, we can compare the table with our result.



The student passes the module if the module mark is at least 40; the student passes by compensation if the module is non-core, (the only core module is Programming II) and module mark is less than 40 but at least 35 or otherwise the module is computed as fail.

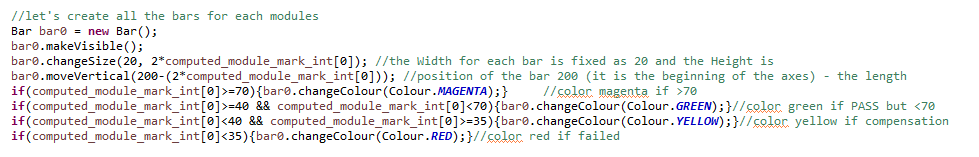
Once all the module marks have been determined, the user can know if Stage one is passed, if and only if all modules are recorded as a pass.

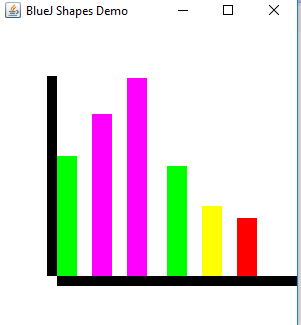
The stage is passed by compensation, if the Stage average is at least 40 and no module is recorded as a Fail, and there are no more than three modules recorded as a compensatable fail or failed otherwise.



The stage result affects the second part of the program in which then, using the Bar class, it draws coloured bars with the height depending on the module mark.

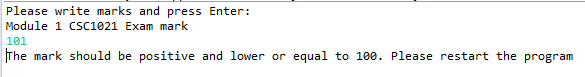
Failing marks should be coloured red, Compensating marks yellow, Pass marks green and First class marks (70 or over) magenta.

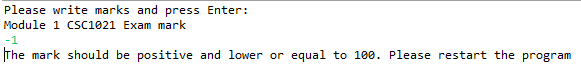




Verify the output doing some maths

First, make sure that the numbers allowed are positive and lower or equal than 100.





|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | CSC1021 | | CSC1022 | | CSC1023 | CSC1024 | | CSC1025 | | CSC1026 | |
|  | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark | Cours.  Mark | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark |
| 1° | 10 | 30 | 10 | 30 | 10 | 10 | 30 | 10 | 30 | 10 | 30 |
| 2° | 10 | 80 | 10 | 80 | 30 | 10 | 80 | 10 | 80 | 10 | 80 |
| 3° | 80 | 10 | 80 | 10 | 80 | 80 | 10 | 80 | 10 | 80 | 10 |
| 4° | 36 | 36 | 78 | 96 | 48 | 37 | 35 | 99 | 54 | 67 | 84 |
| 5° | 35 | 70 | 35 | 70 | 70 | 35 | 70 | 35 | 70 | 35 | 70 |
| 6° | 90 | 80 | 36 | 40 | 70 | 90 | 80 | 70 | 90 | 80 | 70 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MODULE MARK | | | | | | | | | | |
|  | CSC1021 | | CSC1022 | | CSC1023 | CSC1024 | | CSC1025 | | CSC1026 |
| 1° | 20 | | 18 | | 10 | 20 | | 14 | | 17 |
| 2° | 45 | 35 | 38 | 35 | 30 | 45 | 35 | 24 | | 34.5 |
| 3° | 45 | 35 | 52 | 35 | 80 | 45 | 35 | 66 | 35 | 55.5 |
| 4° | 36 | | 85.2 | | 48 | 36 | | 90 | | 72.95 |
| 5° | 52.5 | | 49 | | 70 | 52.5 | | 42 | | 47.25 |
| 6° | 85 | | 37.6 | | 70 | 85 | | 74 | | 76.50 |

First, create a table in order to test every situation of fail, pass by compensation and pass.

Then consider different scenarios:

* All the modules are recorded as Fail and the module mark is less than 35 so the computed mark is the minimum between 35 and the computed mark:

You have 10 for the exam and 30 for CW.

The computed mark based on the different weights (50/50 – 60/40 – 100 – 50/50 – 80/20 – 65/35), as we can read in the table above, are:

((10\*50)+(30\*50))/100=20;

(10\*0.6)+(30\*0.4)=18;

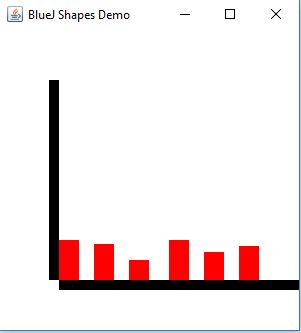
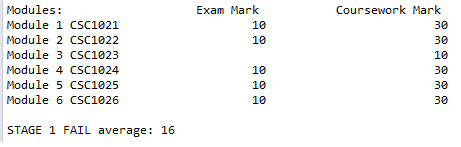
CSC1023 we have coursework only so the mark is the given coursework mark;

(10\*0.8)+(30\*0.2)=14;

(10\*0.65)+(30\*0.35)=17.

Since both the marks are less than 35, the rule minimum (35, computed mark) applies.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | CSC1021 | | CSC1022 | | CSC1023 | CSC1024 | | CSC1025 | | CSC1026 | |
|  | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark | Cours.  Mark | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark |
| 1° | 10 | 30 | 10 | 30 | 10 | 10 | 30 | 10 | 30 | 10 | 30 |
| MODULE MARK | | | | | | | | | | | |
| 1° | 20 | | 18 | | 10 | 20 | | 14 | | 17 | |





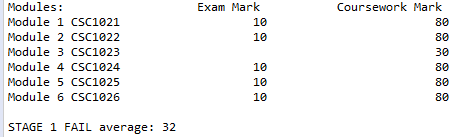
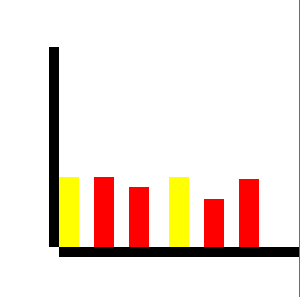
* Two modules are recorded as a compensatable fail but the module mark is greater than 35 so the computed mark is 35 :

If only one of two marks is less than 35 (Exam mark = 10, Coursework mark=80). The computed mark (based on 50/50) is (10\*.5+80\*.5)=45.

Since the Exam mark is less than 35, the rule minimum (35, computed mark) applies, therefore your mark would be minimum(35,45)=35. This module would be a Compensatable fail if it is non-core.

In the table, you can read both of the value, the one computed with the formula (marked with double strikethrough) and the one after the filter (minimum).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | CSC1021 | | | CSC1022 | | | CSC1023 | CSC1024 | | | CSC1025 | | CSC1026 | |
|  | Exam Mark | | Cours. Mark | Exam Mark | | Cours. Mark | Cours.  Mark | Exam Mark | | Cours. Mark | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark |
| 2° | 10 | | 80 | 10 | | 80 | 30 | 10 | | 80 | 10 | 80 | 10 | 80 |
| MODULE MARK | | | | | | | | | | | | | | |
| 2° | 45 | 35 | | 38 | 35 | | 30 | 45 | 35 | | 24 | | 34.5 | |



x

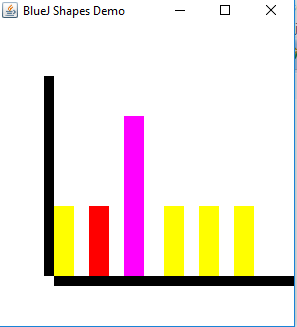
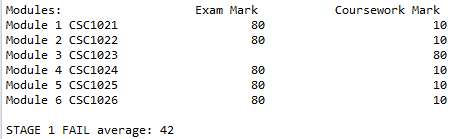
* Compare the second test with this one in order to understand how different weights work. The marks are the same (10 and 80 but now the one which was for the exam is for coursework and vice versa:

In the third test the results are really similar to the previous marks and for the last module the computed result (highlighted) is a floating-point number and it is stored in a more accurate double array to avoid each possible error (double 64 bits maximum value 10^308 vs integer 32 bits 2\*10^9.)

This double then is rounded up into an integer.

The stage result is computed as Fail because there are more than two modules recorded as compensatable fail and the core module, the only module that can only be passed of failed is 35.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | CSC1021 | | | CSC1022 | | | CSC1023 | CSC1024 | | | CSC1025 | | | CSC1026 | | |
|  | Exam Mark | | Cours. Mark | Exam Mark | | Cours. Mark | Cours.  Mark | Exam Mark | | Cours. Mark | Exam Mark | | Cours. Mark | Exam Mark | | Cours. Mark |
| 3° | 80 | | 10 | 80 | | 10 | 80 | 80 | | 10 | 80 | | 10 | 80 | | 10 |
| MODULE MARK | | | | | | | | | | | | | | | | |
| 3° | 45 | 35 | | 52 | 35 | | 80 | 45 | 35 | | 66 | 35 | | 55.5 | 35 | |



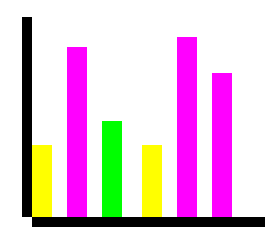


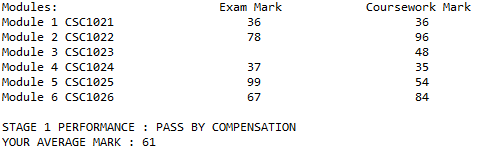
* The Stage is Pass By Compensation because one or two modules are recorded as Compensatable Fail.

If the mark is less than 40 but at least 35, the module (non-core) is recorded as Compensatable Fail.

Then the Stage result can be computed as Pass By Compensation, if the Stage average is at least 40, and there are no more than two modules recorded as a Compensatable Fail.

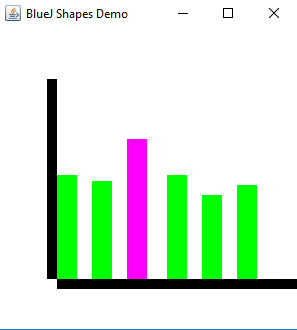
|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | CSC1021 | | CSC1022 | | CSC1023 | CSC1024 | | CSC1025 | | CSC1026 | |
|  | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark | Cours.  Mark | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark |
| 4° | 36 | 36 | 78 | 96 | 48 | 37 | 35 | 99 | 54 | 67 | 84 |
| MODULE MARK | | | | | | | | | | | |
| 4° | 36 | | 85.2 | | 48 | 36 | | 90 | | 72.95 | |

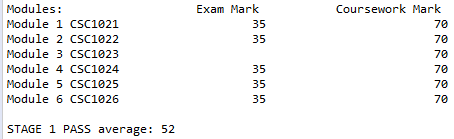




* In this example all the modules are recorded as Pass with one greater or equal to 70:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | CSC1021 | | CSC1022 | | CSC1023 | CSC1024 | | CSC1025 | | CSC1026 | |
|  | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark | Cours.  Mark | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark |
| 5° | 35 | 70 | 35 | 70 | 70 | 35 | 70 | 35 | 70 | 35 | 70 |
| MODULE MARK | | | | | | | | | | | |
| 5° | 52.5 | | 49 | | 70 | 52.5 | | 42 | | 47.25 | |

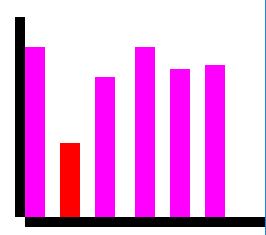


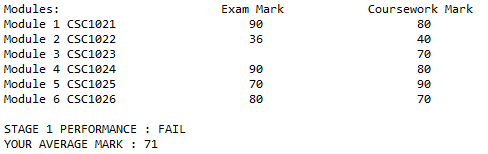




* In this case, the modules greater than 70 are recorded as First class marks but Stage is recorded as Fail because CSC1022 cannot be recorded as a Compensatable fail :

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | CSC1021 | | CSC1022 | | CSC1023 | CSC1024 | | CSC1025 | | CSC1026 | |
|  | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark | Cours.  Mark | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark | Exam Mark | Cours. Mark |
| 6° | 90 | 80 | 36 | 40 | 70 | 90 | 80 | 70 | 90 | 80 | 70 |
| MODULE MARK | | | | | | | | | | | |
| 6° | 85 | | 37.6 | | 70 | 85 | | 74 | | 76.5 | |







Conclusion

Upon providing mathematical tests with the relative calculations and testing all possible outcome and scenarios that might impact the capability of the program, we can with certainty agree that the program is indeed working as intended.

Developing it was not easy because I had to integrate my own knowledge and how much I learned during the lectures with an in-depth research in books and on the internet.

To successfully create this software, I have followed step-by-step all the requirements even though I have often found some difficulties since English is not my first language.

Another problem was working with arrays, creating a single array to store the input of the user, then to process all the marks in the various classes using the ‘return statement’ correctly for each method because it was often necessary to return more than one variable used simultaneously by multiple methods.

One of the main difficulties I came across was calculating the module mark correctly with the formula and using it to check the requirement to determine the outcome of the Stage.

During all the practical exercises I learned how to do an accurate problem-solving work by looking for errors in saving data while working with arrays using ‘for loops’ and ‘if statements’ printing on the console all the values to find out each possible error.

On the other hand, I succeed in using the ‘switch statement’ and creating a constructor learning the bases of object-oriented development, assigning objects like bars the right attributes (e.g. colour and size.)