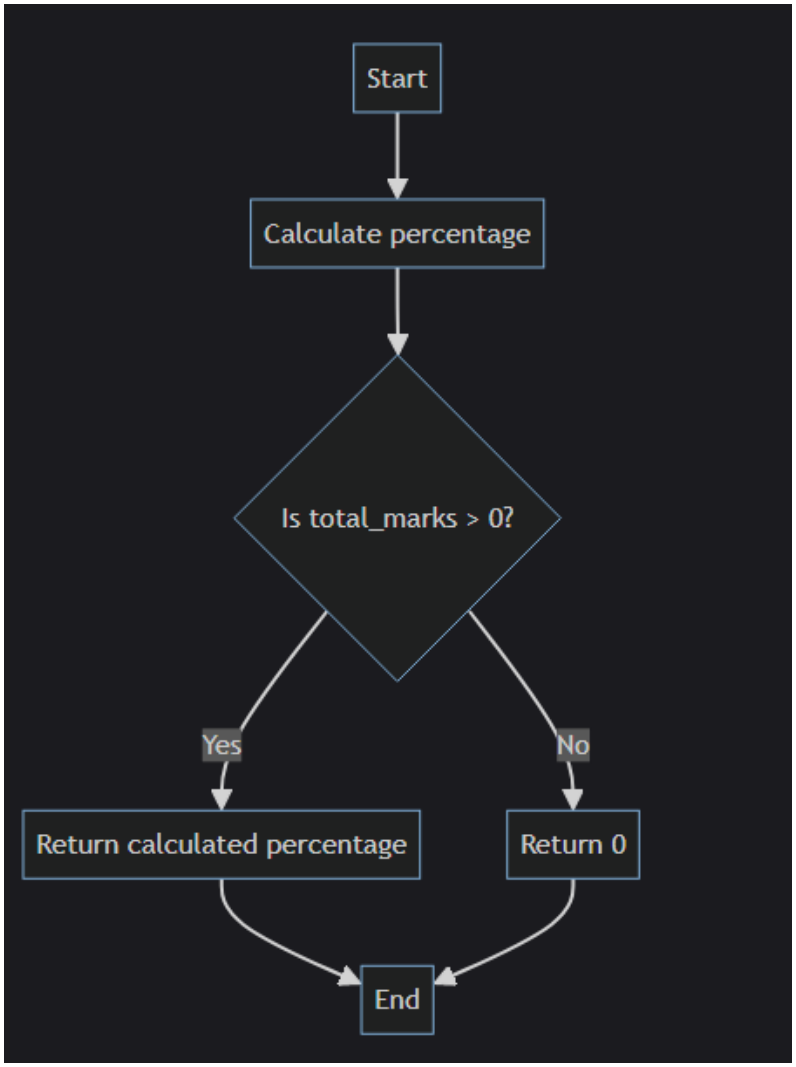
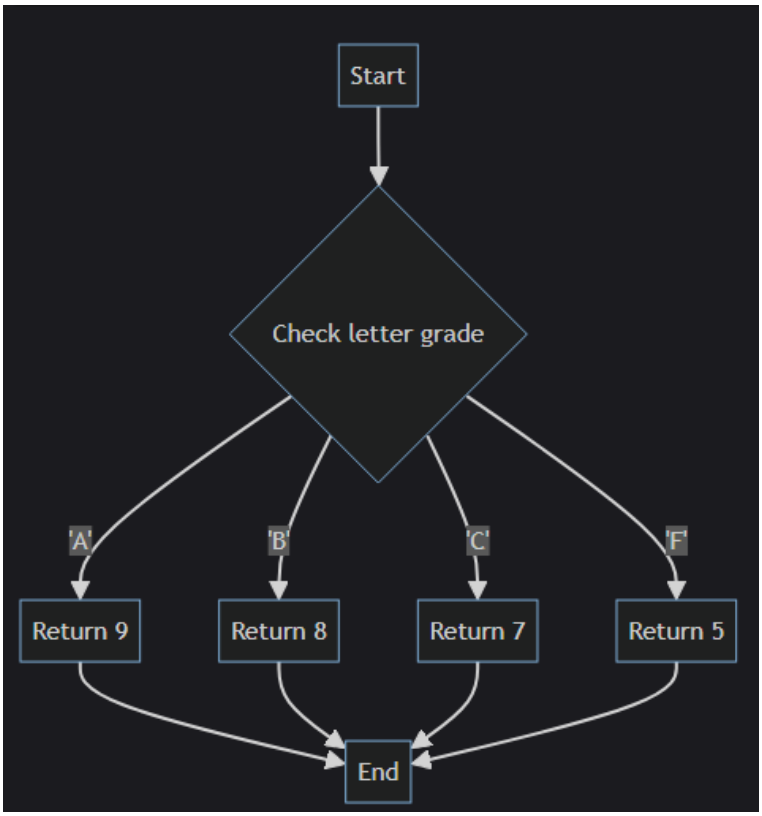
1.1.

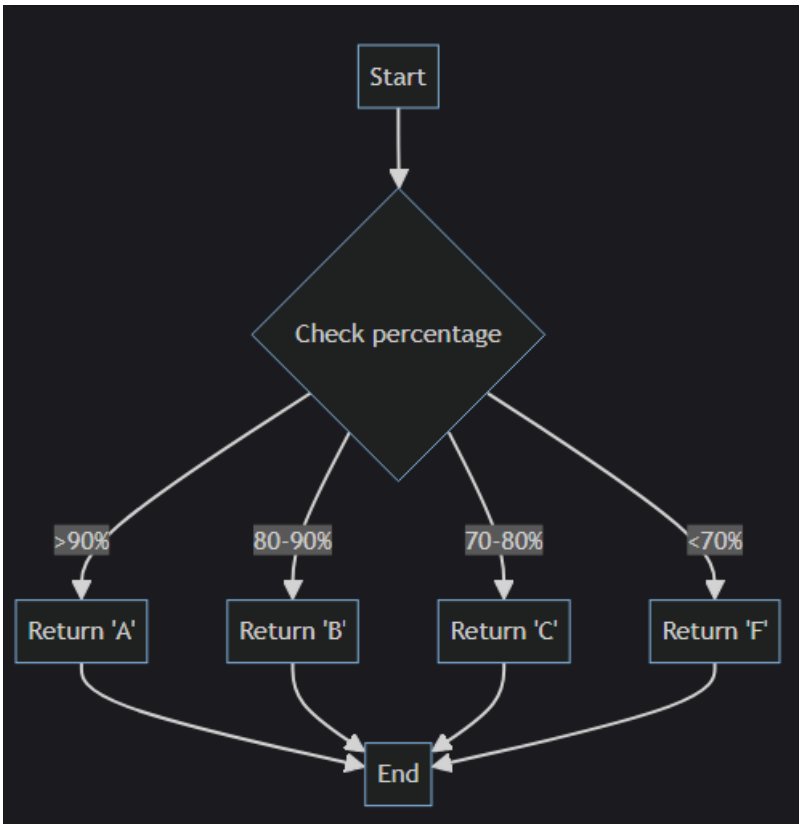
percentage_grade Graph



Letter_grade Graph



Numeric_grade Graph



1.2.

Test Cases for percentage_grade

Test Case Number	Test Data	Expected Results	Conditions Covered	Branches Covered
1	total_marks = 50, max_possible_marks = 100	50%	max_possible_marks > 0	Path A-B-C-D-F
2	total_marks = 0, max_possible_marks = 100	0%	max_possible_marks > 0, total_marks = 0	Path A-B-C-D-F
3	total_marks = 50, max_possible_marks = 0	Return "Error: Division by zero"	max_possible_marks = 0	Path A-B-C-E-F

Test Cases for numeric_grade

Test Case Number	Test Data	Expected Results	Conditions Covered	Branches Covered
1	letter_grade = 'A'	9	'A'	Path A-B-C-G
2	letter_grade = 'B'	8	'B'	Path A-B-D-G
3	letter_grade = 'C'	7	'C'	Path A-B-E-G
4	letter_grade = 'F'	5	'F'	Path A-B-F-G

Test Cases for letter_grade

Test Case Number	Test Data	Expected Results	Conditions Covered	Branches Covered
1	percentage = 95	'A'	percentage > 90	Path A-B-C-G
2	percentage = 85	'B'	80 < percentage ≤ 90	Path A-B-D-G
3	percentage = 75	'C'	70 < percentage ≤ 80	Path A-B-E-G
4	percentage = 65	'F'	percentage ≤ 70	Path A-B-F-G

1.3. Please see file called *calculator_test.exs* in the GitHub Repository under folder Q1. Please see file called *Branch Coverage* for Proof of 100% branch coverage

1.4.

Achieved Coverage:

Our test suite for the Grades.Calculator module achieved 100% statement coverage. This means that every line of code in the methods percentage_grade/2, letter_grade/1, and numeric_grade/1 was executed at least once during our tests. Please see Question 1.3 for the location of proof of 100% coverage.

Addressing Limitations of Statement-Level Coverage Tools

Statement-level coverage measures whether each line of code has been executed during tests. However, it does not ensure that all logical paths or combinations of conditions have been tested. This can leave potential logical errors undetected, especially in complex conditional structures or where multiple conditions combine to influence outcomes. While statement coverage ensures each line of code runs, branch coverage goes deeper by ensuring every possible route through control structures (like if, else, switch, etc.) is executed.