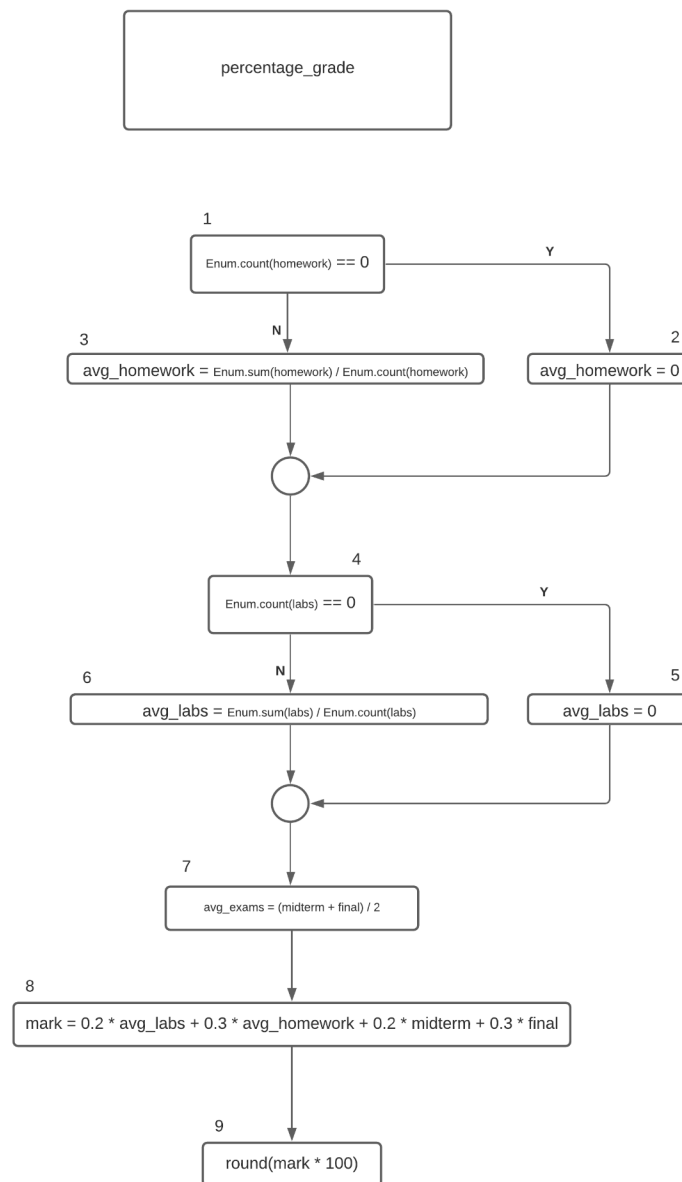


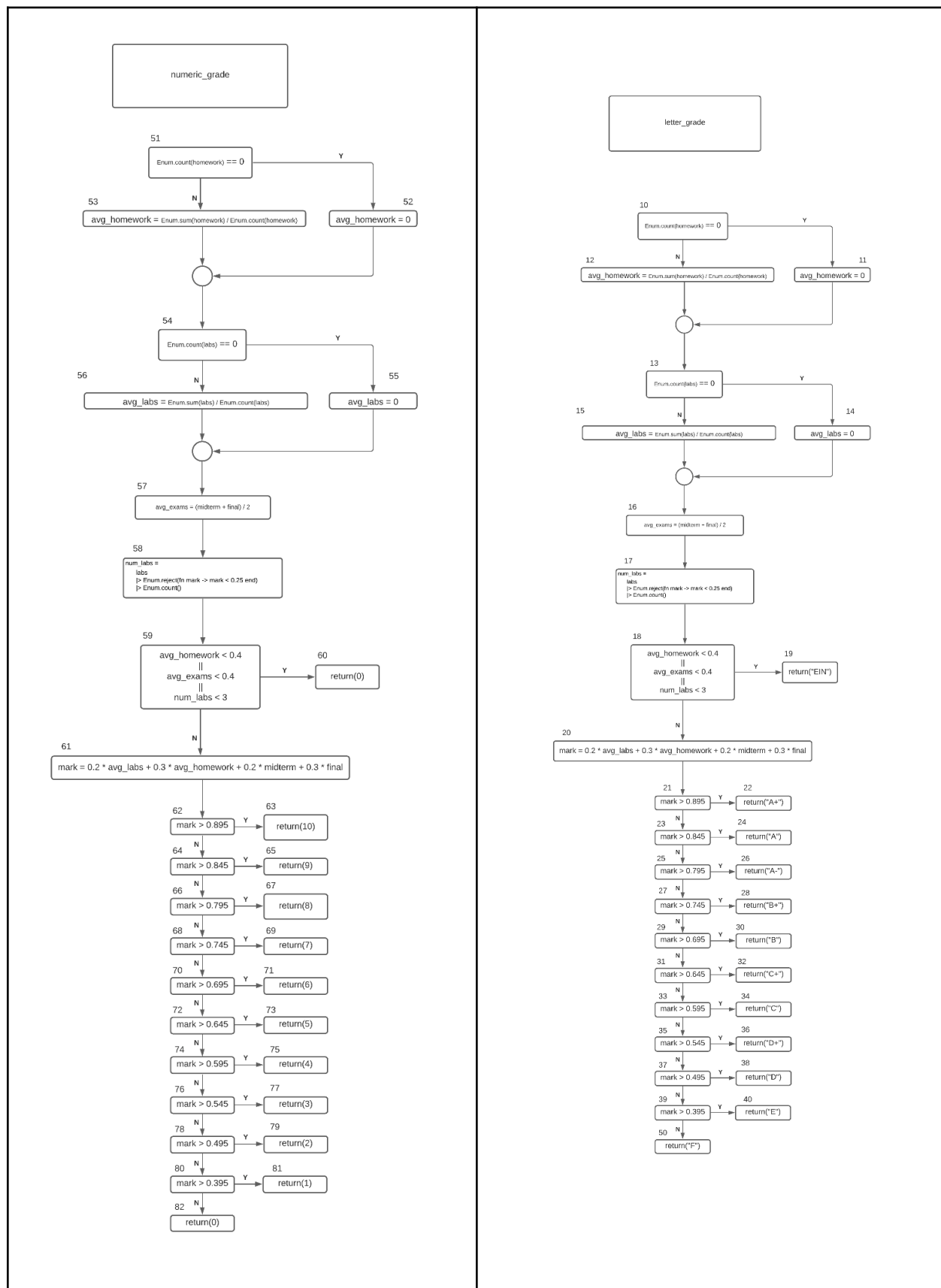
# Assignment 2

By Gaberiel Cordovado (300110852)  
And Michael Kagnev (300113347)

## Question 1.1

Draw the simplified control flow graph corresponding to each of the methods `percentage_grade`, `letter_grade`, and `numeric_grade`.





## Question 1.2

Provide a white box test design for 100% branch coverage of the methods `percentage_grade`, `letter_grade`, and `numeric_grade`. Your test suite will be evaluated on the number of its test cases (try to have the smallest possible number of test cases that will allow 100% branch coverage). Use the following template for your test case design:

Tests **1-12** input data into the **`letter_grade()`** function.

Tests **13-24** input data into the **`numeric_grade()`** function.

Tests **25-26** input data into the **`percentage_grade()`** function.

\* The branch-cover associated with each test case corresponds to the edge between two statements that has NOT been covered by any test-case before. Hence, after each test, all branches should be mentioned ONCE in the table under the “branches covered” column.

=> ex. Test case 2 will not contain branch edges 10 to 20 because they have been covered by test case 1.

\* some conditions are grouped together for visual purposes, definitions are under the table.

Test Case Number	Test Data	Expected Result For Test Case	Conditions Covered	Branches covered
1, 13, 25	Homework = [0.99] Labs = [0.99,0.99,0.99] Midterm = 0.99 Final = 0.99	(1) "A+" (13) 10 (25) 99	Common Conditions <sup>1</sup> mark > 0.895	(1) 10-12-13-15-16-17-18-20-21-22 (13) 51-53-54-56-57-58-59-61-62-63 (25) 1-3-4-6-7-8-9
2, 14	Homework = [0.85] Labs = [0.85, 0.85, 0.85] Midterm = 0.85 Final = 0.85	(2) "A" (14) 9	Common Conditions <sup>1</sup> mark < 0.895 mark > 0.845	(2) 23-24 (14) 64-65
3, 15	Homework = [0.80] Labs = [0.80, 0.80, 0.80] Midterm = 0.80 Final = 0.80	(3) "A-" (15) 8	Common Conditions <sup>1</sup> mark < 0.845 mark > 0.795	(3) 25-26 (15) 66-67
4, 16	Homework = ]0.75] Labs = [0.75,0.75,0.75] Midterm = 0.75	(4) "B+" (16) 7	Common Conditions <sup>1</sup> mark < 0.795 mark > 0.745	(4) 27-28 (16) 68-69

	Final = 0.75			
5, 17	Homework = [0.70] Labs = [0.70,0.70,0.70] Midterm = 0.70 Final = 0.70	<b>(5)</b> "B" <b>(17)</b> 6	Common Conditions <sup>1</sup> mark < 0.745 mark > 0.695	<b>(5)</b> 29-30 <b>(17)</b> 70-71
6, 18	Homework = 0.65 Labs = [0.65,0.65,0.65] Midterm = 0.65 Final = 0.65	<b>(6)</b> "C+" <b>(18)</b> 5	Common Conditions <sup>1</sup> mark < 0.695 mark > 0.645	<b>(6)</b> 31-32 <b>(18)</b> 72-73
7, 19	Homework = [0.60] Labs = [0.60,0.60,0.60] Midterm = 0.60 Final = 0.60	<b>(7)</b> "C" <b>(19)</b> 4	Common Conditions <sup>1</sup> mark < 0.645 mark > 0.595	<b>(7)</b> 33-34 <b>(19)</b> 74-75
8, 20	Homework = [0.55] Labs = [0.55,0.55,0.55] Midterm = 0.55 Final = 0.55	<b>(8)</b> "D+" <b>(20)</b> 3	Common Conditions <sup>1</sup> mark < 0.595 mark > 0.545	<b>(8)</b> 35-36 <b>(20)</b> 76-77
9, 21	Homework = [0.50] Labs = [0.50,0.50,0.50] Midterm = 0.50 Final = 0.50	<b>(9)</b> "D" <b>(21)</b> 2	Common Conditions <sup>1</sup> mark < 0.545 mark > 0.495	<b>(9)</b> 37-38 <b>(21)</b> 78-79
10, 22	Homework = [0.40] Labs = [0.40,0.40,0.40] Midterm = 0.40 Final = 0.40	<b>(10)</b> "E" <b>(22)</b> 1	Common Conditions <sup>1</sup> mark < 0.495 mark > 0.395	<b>(10)</b> 39-40 <b>(22)</b> 80-81
11, 23	Homework = [0.40] Labs = [0,0,0,0.25,0.25,0.25,0.25,0.25,0,0,0,0,0,0] Midterm = 0.4 Final = 0.4	<b>(11)</b> "F" <b>(23)</b> 0	Common Conditions <sup>1</sup> mark < 0.395	<b>(11)</b> 39-50 <b>(23)</b> 80-82
12, 24, 26	Homework = [] Labs = [] Midterm = 0.3 Final = 0.3	<b>(12)</b> "EIN" <b>(24)</b> 0 <b>(26)</b> 15	Uncommon Conditions <sup>2</sup>  Enum.count(homework) == 0  Enum.count(labs) == 0	<b>(12)</b> 10-11-13-14, 18-19 <b>(24)</b> 51-52-54-55-57, 59-60 <b>(26)</b> 1-2-4-5-7

[1] Common Conditions ( Enum.count(homework) != 0, Enum.count(labs) != 0, Avg\_homework > 0.4, Avg\_exams > 0.4, Num\_labs > 3 )

[2] Uncommon Conditions ( Avg\_homework < 0.4, Avg\_exams < 0.4, Num\_labs < 3 )

### Question 1.3

Provide an implementation of your test suite using ExUnit.

\* code embedded within the zipped-repository

### Question 1.4

What is the degree of statement coverage obtained? If you weren't able to obtain 100% coverage, explain why. Please be sure to attach screenshots of coverage results. Elixir's coverage tool is primitive, as it only provides statement level accuracy. How might you address the limitations of a testing tool that only provides statement level coverage?

```
Finished in 0.2 seconds (0.1s async, 0.1s sync)
29 tests, 0 failures

Randomized with seed 163000

Generating cover results ...

Percentage | Module
-----|-----
0.00% | GradesWeb
0.00% | GradesWeb.ChannelCase
0.00% | GradesWeb.ErrorHelpers
9.09% | GradesWeb.Pagelive
50.00% | GradesWeb.LayoutView
50.00% | GradesWeb.UserSocket
66.67% | GradesWeb.ErrorView
75.00% | Grades.Application
75.00% | GradesWeb.Router
75.00% | GradesWeb.Telemetry
100.00% | Grades
100.00% | Grades.Calculator
100.00% | GradesWeb.ConnCase
100.00% | GradesWeb.Endpoint
100.00% | GradesWeb.Router.Helpers
-----|-----
74.16% | Total

Generated HTML coverage results in "cover" directory
```

*Coverage results from `mix test --cover` command*

The degree of statement coverage for the calculator module comes out to 100%. The limitations of the testing coverage only to statements will miss potential logic issues that are only identifiable in other coverage types, such as branch and/or condition coverage. Statement coverage only indicates that each statement has been executed once, but does not test if the true/false logic in conditional statements are achievable, which is not done by statement coverage. To address this issue, one could either use languages that provide other types of coverage, or manually test for other coverages in flow charts, such as what we did in question 1.1.

## Question 2.1

Extract a helper method **avg** to cleanup the duplicate code like...

```
avg_homework =  
  if Enum.count(homework) == 0 do  
    0  
  else  
    Enum.sum(homework) / Enum.count(homework)  
  end
```

**Commit:** [75fd539f00b01fef9fe2d187a1c0b632f2c4f9bc](#)

**Commit-Title:** refactor calculation for the average of a list of floats

**Commit-Message:** replaced repetitive code regarding the avg grade of a list of float values, this was replaced by an avg function that takes one parameter, the list of floats, and returns a float representing the avg grade of those values. Lines 19, 21, 28, 30, 61, 63 reflect these changes. All test cases pass

\* image of commit on next page of the document \*

```

... 50 grades/lib/grades/calculator.ex
... @@ -1,37 +1,33 @@
1 1 defmodule Grades.Calculator do
2 +
3 + # Question 2.1 (Gabriel Cordovado)
4 + # This function will take in the homework and lab lists separately to reduce the amount of code
5 + # that is being duplicated within the percentage_grade, letter_grade, and numeric_grade classes
6 + #
7 + # @param list_of_n_items: list of floats
8 + # @returns an float representing the average of N items within the list of floats
9 + #
10 + def avg(list_of_n_items) do
11 +   if Enum.count(list_of_n_items) == 0 do
12 +     0
13 +   else
14 +     Enum.sum(list_of_n_items) / Enum.count(list_of_n_items)
15 +   end
16 + end
17 +
18 2 def percentage_grade(%{homework: homework, labs: labs, midterm: midterm, final: final}) do
19 +   avg_homework =
20 -     if Enum.count(homework) == 0 do
21 -       0
22 -     else
23 -       Enum.sum(homework) / Enum.count(homework)
24 -     end
25 +   avg_homework = avg(homework) # refactor 2.1
26
27 9 20
28 10 -   avg_labs =
29 -     if Enum.count(labs) == 0 do
30 -       0
31 -     else
32 -       Enum.sum(labs) / Enum.count(labs)
33 -     end
34 +   avg_labs = avg(labs) # refactor 2.1
35
36 16 22
37 17 23   mark = 0.2 * avg_labs + 0.3 * avg_homework + 0.2 * midterm + 0.3 * final
38 18 24   round(mark * 100)
39 19 25   end
40 20 26
41 21 27   def letter_grade(%{homework: homework, labs: labs, midterm: midterm, final: final}) do
42 -     avg_homework =
43 -       if Enum.count(homework) == 0 do
44 -         0
45 -       else
46 -         Enum.sum(homework) / Enum.count(homework)
47 -       end
48 +     avg_homework = avg(homework) # refactor 2.1
49
50 28 29
51 29 -     avg_labs =
52 -       if Enum.count(labs) == 0 do
53 -         0
54 -       else
55 -         Enum.sum(labs) / Enum.count(labs)
56 -       end
57 +     avg_labs = avg(labs) # refactor 2.1
58
59 35 31
60 36 32   avg_exams = (midterm + final) / 2
61 37 33
62 + @@ -62,19 +58,9 @@ defmodule Grades.Calculator do
63 58   end
64 59
65 60   def numeric_grade(%{homework: homework, labs: labs, midterm: midterm, final: final}) do
66 -     avg_homework =
67 -       if Enum.count(homework) == 0 do
68 -         0
69 -       else
70 -         Enum.sum(homework) / Enum.count(homework)
71 -       end
72 +     avg_homework = avg(homework) # refactor 2.1
73
74 71 62
75 72 -     avg_labs =
76 -       if Enum.count(labs) == 0 do
77 -         0
78 -       else
79 -         Enum.sum(labs) / Enum.count(labs)
80 -       end
81 +     avg_labs = avg(labs) # refactor 2.1
82
83 78 64
84 79 65   avg_exams = (midterm + final) / 2
85 80 66

```

## Question 2.2

Extract a helper method **failed\_to\_participate** to clean up duplicate code like...

```
avg_homework < 0.4 || avg_exams < 0.4 || num_labs(labs) < 3
```

Commit: [2d1bd4f8b481d440db698266b94a86df5d8a9df3](#)

Commit-Title: Refactor 2.2

Commit-Message: wrapped the calculation if the student was able to participate in marking in a single function to return true or false. Affects lines 60, and 93

```

16 grades/lib/grades/calculator.ex
@@ -26,6 +26,15 @@ defmodule Grades.Calculator do
26   def calculate_grade(avg_labs, avg_homework, midterm, final) do
27     0.2 * avg_labs + 0.3 * avg_homework + 0.2 * midterm + 0.3 * final
28   end
29 +
30 +   # Question 2.2 (Michael Kagnew)
31 +   # The formula used to determine if the student is able to participate in the marking scheme.
32 +   # @param avg_homework, avg_exams, num_labs : boolean
33 +   # @returns a boolean determining if the student is able to participate in the grade calculation.
34 +   #
35 +   def failed_to_participate(avg_homework, avg_exams, num_labs) do
36 +     avg_homework < 0.4 || avg_exams < 0.4 || num_labs < 3
37 +   end
38
39   def percentage_grade(%{homework: homework, labs: labs, midterm: midterm, final: final}) do
40     avg_homework = avg(homework) # refactor 2.1
41
42   @@ -48,7 +57,7 @@ defmodule Grades.Calculator do
48     |> Enum.reject(fn mark -> mark < 0.25 end)
49     |> Enum.count()
50
51 -   if avg_homework < 0.4 || avg_exams < 0.4 || num_labs < 3 do
52 +   if failed_to_participate(avg_homework, avg_exams, num_labs) do #refactor 2.2
53     "EIN"
54   else
55     mark = calculate_grade(avg_labs, avg_homework, midterm, final) # refactor 2.3
56
57   @@ -81,7 +90,7 @@ defmodule Grades.Calculator do
81     |> Enum.reject(fn mark -> mark < 0.25 end)
82     |> Enum.count()
83
84 -   if avg_homework < 0.4 || avg_exams < 0.4 || num_labs < 3 do
85 +   if failed_to_participate(avg_homework, avg_exams, num_labs) do #refactor 2.2
86     0
87   else
88     mark = calculate_grade(avg_labs, avg_homework, midterm, final) # refactor 2.3
89
90   @@ -101,4 +110,7 @@ defmodule Grades.Calculator do
101     end
102   end
103   end
104
105 +
106 +
107 +
108   end

```



## Question 2.3

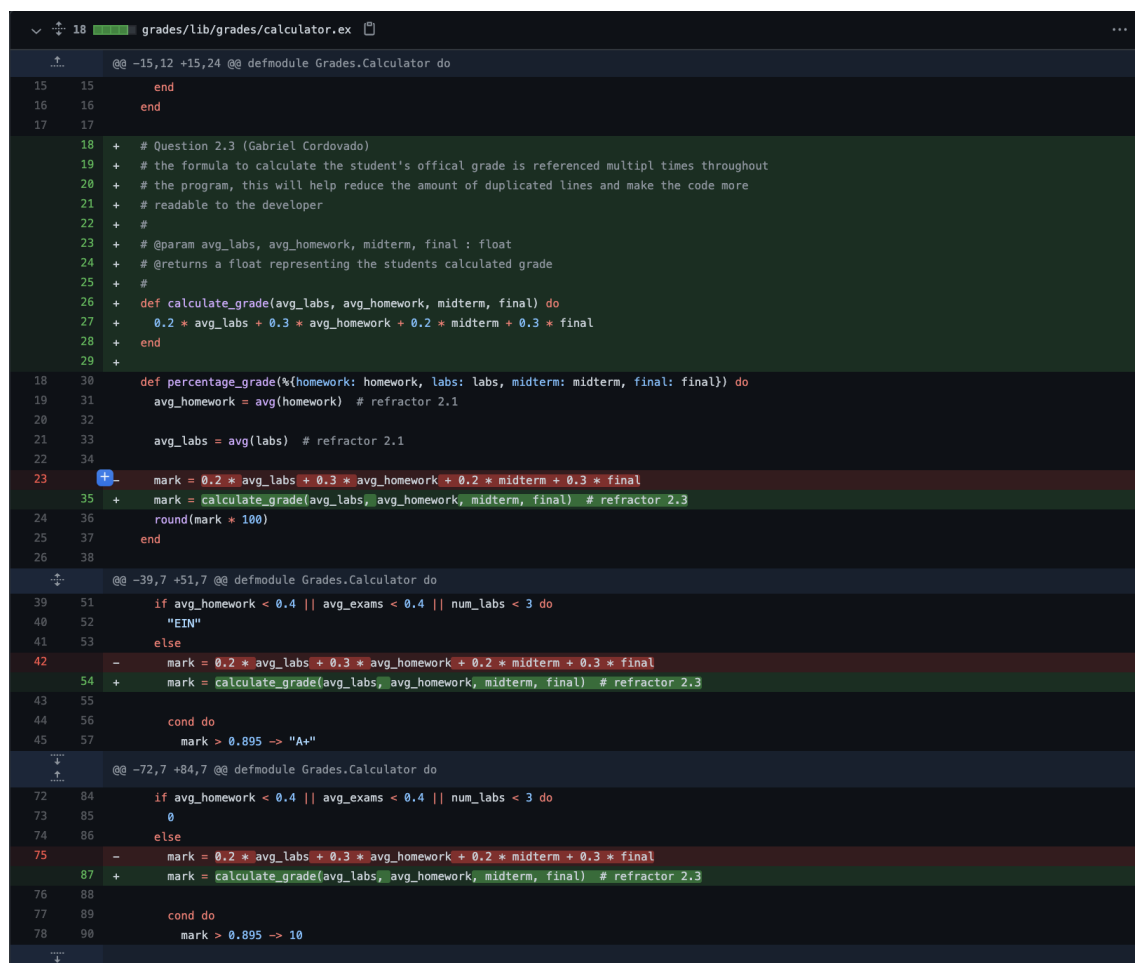
Extract a helper method **calculate\_grade** to clean up duplicate code like...

```
0.2 * avg_labs + 0.3 * avg_homework + 0.2 * midterm + 0.3 * final
```

**Commit:** [3cc0560ac15b411f583cfd6b488284cc85e6251d](#)

**Commit-Title:** refactor 2.3 abstracted to formula to calculate the student final grade into calculate\_grade

**Commit-Message:** the formula was placed in multiple functions within the Grade.calculator module, this should ease code readability and maintainability. Affects lines 35, 54, and 87. All coverage tests pass



```
15 15 end
16 16 end
17 17
18 + # Question 2.3 (Gabriel Cordovado)
19 + # the formula to calculate the student's official grade is referenced multiple times throughout
20 + # the program, this will help reduce the amount of duplicated lines and make the code more
21 + # readable to the developer
22 + #
23 + # @param avg_labs, avg_homework, midterm, final : float
24 + # @returns a float representing the student's calculated grade
25 + #
26 + def calculate_grade(avg_labs, avg_homework, midterm, final) do
27 +   0.2 * avg_labs + 0.3 * avg_homework + 0.2 * midterm + 0.3 * final
28 + end
29 +
30 def percentage_grade(%{homework: homework, labs: labs, midterm: midterm, final: final}) do
31   avg_homework = avg(homework) # refactor 2.1
32
33   avg_labs = avg(labs) # refactor 2.1
34
35 + mark = 0.2 * avg_labs + 0.3 * avg_homework + 0.2 * midterm + 0.3 * final
36 + mark = calculate_grade(avg_labs, avg_homework, midterm, final) # refactor 2.3
37   round(mark * 100)
38 end
39
40 @@ -39,7 +51,7 @@ defmodule Grades.Calculator do
41 51 if avg_homework < 0.4 || avg_exams < 0.4 || num_labs < 3 do
42 52 "EIN"
43 53 else
44 54 mark = 0.2 * avg_labs + 0.3 * avg_homework + 0.2 * midterm + 0.3 * final
45 55 mark = calculate_grade(avg_labs, avg_homework, midterm, final) # refactor 2.3
46 56
47 57 cond do
48 58 mark > 0.895 -> "A+"
49
50 @@ -72,7 +84,7 @@ defmodule Grades.Calculator do
51 84 if avg_homework < 0.4 || avg_exams < 0.4 || num_labs < 3 do
52 85 0
53 86 else
54 87 mark = 0.2 * avg_labs + 0.3 * avg_homework + 0.2 * midterm + 0.3 * final
55 88 mark = calculate_grade(avg_labs, avg_homework, midterm, final) # refactor 2.3
56 89
57 90 cond do
58 91 mark > 0.895 -> 10
```

## Question 2.4

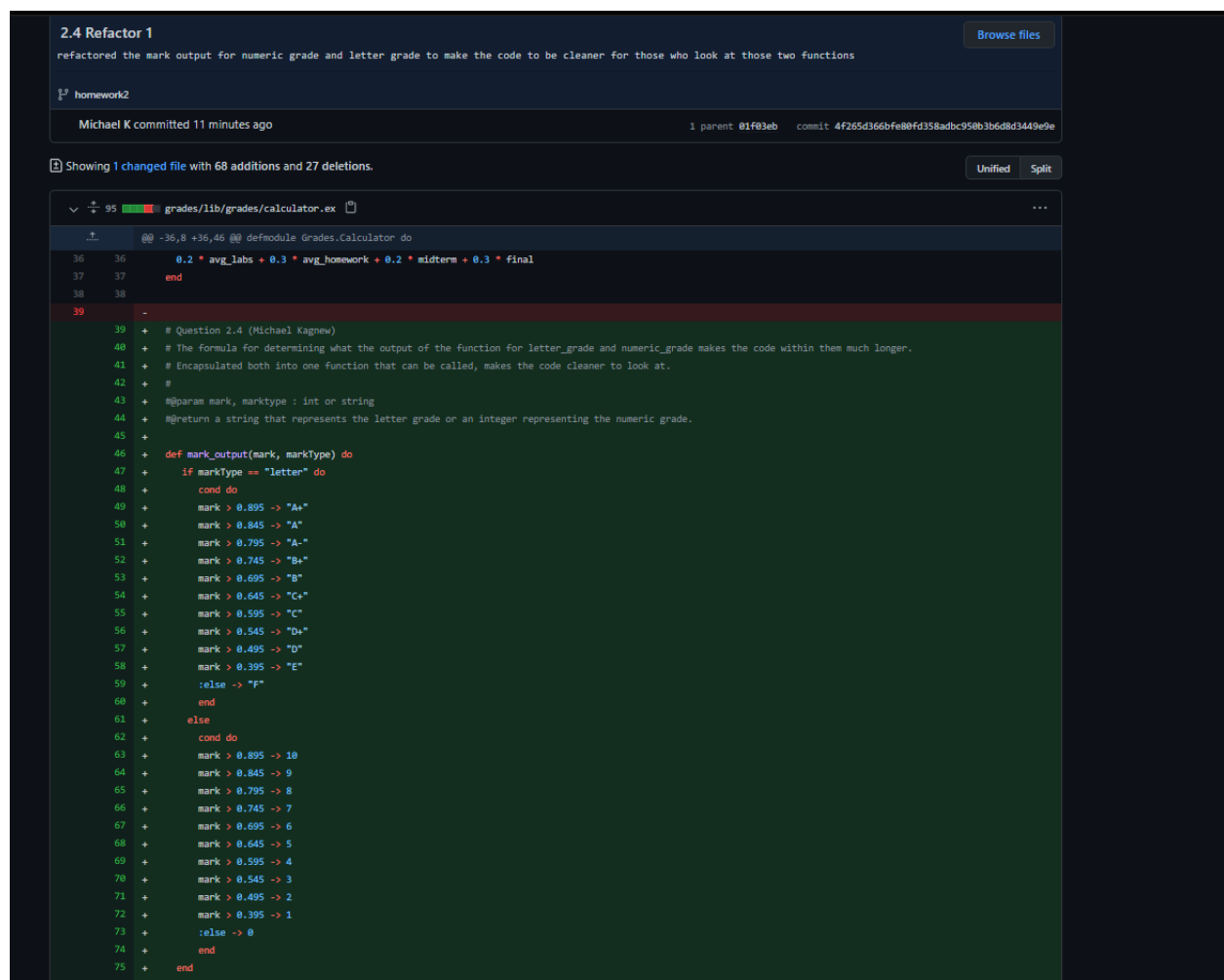
Provide at least 2 additional refactoring to the code. Your refactoring should not require additional testing, however if you encounter any bugs in the original code then please fix them separately (ensuring your tests continue to pass) before continuing to refactor.

(1)

**Commit:** [4f265d366bfe80fd358adbc950b3b6d8d3449e9e](#)

**Commit-Title:** 2.4 Refactor 1

**Commit-Message:** refactored the mark output for numeric grade and letter grade to make the code to be cleaner for those who look at those two functions



The screenshot shows a commit interface for a repository named 'homework2'. The commit is titled '2.4 Refactor 1' and has a message: 'refactored the mark output for numeric grade and letter grade to make the code to be cleaner for those who look at those two functions'. The commit was made by Michael K 11 minutes ago. The diff shows changes to the file 'grades/lib/grades/calculator.ex'. The code is in Elixir and defines a function 'mark\_output' that takes 'mark' and 'markType' as arguments. It uses a case statement to return either a letter grade or a numeric grade based on the 'mark' value. The diff highlights the refactored code with green lines for additions and red lines for deletions.

```
2.4 Refactor 1
refactored the mark output for numeric grade and letter grade to make the code to be cleaner for those who look at those two functions

homework2

Michael K committed 11 minutes ago
1 parent 01f03eb commit 4f265d366bfe80fd358adbc950b3b6d8d3449e9e

Showing 1 changed file with 68 additions and 27 deletions.

grades/lib/grades/calculator.ex
@@ -36,8 +36,46 @@ defmodule Grades.Calculator do
 36   0.2 * avg_labs + 0.3 * avg_homework + 0.2 * midterm + 0.3 * final
 37   end
 38
 39
 39 + # Question 2.4 (Michael Kagnev)
 40 + # The formula for determining what the output of the function for letter_grade and numeric_grade makes the code within them much longer.
 41 + # Encapsulated both into one function that can be called, makes the code cleaner to look at.
 42 + #
 43 + @param mark, markType : int or string
 44 + @return a string that represents the letter grade or an integer representing the numeric grade.
 45 +
 46 + def mark_output(mark, markType) do
 47 +   if markType == "letter" do
 48 +     cond do
 49 +       mark > 0.895 -> "A+"
 50 +       mark > 0.845 -> "A"
 51 +       mark > 0.795 -> "A-"
 52 +       mark > 0.745 -> "B+"
 53 +       mark > 0.695 -> "B"
 54 +       mark > 0.645 -> "C+"
 55 +       mark > 0.595 -> "C"
 56 +       mark > 0.545 -> "D+"
 57 +       mark > 0.495 -> "D"
 58 +       mark > 0.395 -> "E"
 59 +       :else -> "F"
 60 +     end
 61 +   else
 62 +     cond do
 63 +       mark > 0.895 -> 10
 64 +       mark > 0.845 -> 9
 65 +       mark > 0.795 -> 8
 66 +       mark > 0.745 -> 7
 67 +       mark > 0.695 -> 6
 68 +       mark > 0.645 -> 5
 69 +       mark > 0.595 -> 4
 70 +       mark > 0.545 -> 3
 71 +       mark > 0.495 -> 2
 72 +       mark > 0.395 -> 1
 73 +       :else -> 0
 74 +     end
 75 +   end
 76 + end
```

Refactor 2.4 (1/2): half of the commit screenshot

```

64 182     else
65 183         mark = calculate_grade(avg_labs, avg_homework, midterm, final) # refactor 2.3
66 184
67
68     -     cond do
69     -         mark > 0.895 -> "A+"
70     -         mark > 0.845 -> "A"
71     -         mark > 0.795 -> "A-"
72     -         mark > 0.745 -> "B+"
73     -         mark > 0.695 -> "B"
74     -         mark > 0.645 -> "C+"
75     -         mark > 0.595 -> "C"
76     -         mark > 0.545 -> "D+"
77     -         mark > 0.495 -> "D"
78     -         mark > 0.395 -> "E"
79     -         :else -> "F"
80     -     end
81
82     +     # cond do
83     +     +     mark > 0.895 -> "A+"
84     +     +     mark > 0.845 -> "A"
85     +     +     mark > 0.795 -> "A-"
86     +     +     mark > 0.745 -> "B+"
87     +     +     mark > 0.695 -> "B"
88     +     +     mark > 0.645 -> "C+"
89     +     +     mark > 0.595 -> "C"
90     +     +     mark > 0.545 -> "D+"
91     +     +     mark > 0.495 -> "D"
92     +     +     mark > 0.395 -> "E"
93     +     +     :else -> "F"
94     +     +     end
95     +     mark_output(mark, "letter")
96
97 119     end
98 120
99 121     end
100
101 @@ -97,19 +136,21 @@ defmodule Grades.Calculator do
102
103     97 136     else
104     98 137         mark = calculate_grade(avg_labs, avg_homework, midterm, final) # refactor 2.3
105     99 138
106
107     -     cond do
108     -         mark > 0.895 -> 10
109     -         mark > 0.845 -> 9
110     -         mark > 0.795 -> 8
111     -         mark > 0.745 -> 7
112     -         mark > 0.695 -> 6
113     -         mark > 0.645 -> 5
114     -         mark > 0.595 -> 4
115     -         mark > 0.545 -> 3
116     -         mark > 0.495 -> 2
117     -         mark > 0.395 -> 1
118     -         :else -> 0
119     -     end
120
121     +     # cond do
122     +     +     mark > 0.895 -> 10
123     +     +     mark > 0.845 -> 9
124     +     +     mark > 0.795 -> 8
125     +     +     mark > 0.745 -> 7
126     +     +     mark > 0.695 -> 6
127     +     +     mark > 0.645 -> 5
128     +     +     mark > 0.595 -> 4
129     +     +     mark > 0.545 -> 3
130     +     +     mark > 0.495 -> 2
131     +     +     mark > 0.395 -> 1
132     +     +     :else -> 0
133     +     +     end
134     +     mark_output(mark, "number")
135
136 113 154     end

```

Refactor 2.4 (2/2): Other half of the commit screenshot

(2)

**Commit:** [4777ce16a34bcededd1250a19612191273527e89](#)**Commit-Title:** refractor 2.4 - part 3**Commit-Message:** placed the formula for calculating the number of labs into an isolated function to avoid the need for 3 repeated lines in each case of percentage\_grade and numeric\_grade. The changes take place on lines 106 and 121. All tests pass.

```

68 grades/lib/grades/calculator.ex
@@ -14,13 +14,13 @@ defmodule Grades.Calculator do
14 14     Enum.sum(list_of_n_items) / Enum.count(list_of_n_items)
15 15     end
16 16     end
17 17 -
17 17 +
18 18     # Question 2.2 (Michael Kagnew)
19 19     # The formula used to determine if the student is able to participate in the marking scheme.
20 20     # @param avg_homework, avg_exams, num_labs : boolean
21 21     # @returns a boolean determining if the student is able to participate in the grade calculation.
22 22     #
23 23 - def failed_to_participate(avg_homework, avg_exams, num_labs) do
23 23 + def failed_to_participate(avg_homework, avg_exams, num_labs) do
24 24     avg_homework < 0.4 || avg_exams < 0.4 || num_labs < 3
25 25     end
26 26
@@ -35,14 +35,14 @@ defmodule Grades.Calculator do
35 35     def calculate_grade(avg_labs, avg_homework, midterm, final) do
36 36         0.2 * avg_labs + 0.3 * avg_homework + 0.2 * midterm + 0.3 * final
37 37     end
38 38 -
38 38 +
39 39     # Question 2.4 (Michael Kagnew)
40 40     # The formula for determining what the output of the function for letter_grade and numeric_grade makes the code within them much longer.
41 41     # Encapsulated both into one function that can be called, makes the code cleaner to look at.
42 42     #
43 43     # @param mark, marktype : int or string
44 44     # @return a string that represents the letter grade or an integer representing the numeric grade.
45 45 -
45 45 +
46 46     def mark_output(mark, markType) do
47 47         if markType == "letter" do
48 48             cond do
@@ -59,7 +59,7 @@ defmodule Grades.Calculator do
59 59         :else -> "F"
60 60     end
61 61     else
62 62 -         cond do
62 62 +         cond do
63 63             mark > 0.895 -> 10
64 64             mark > 0.845 -> 9
65 65             mark > 0.795 -> 8
@@ -75,7 +75,7 @@ defmodule Grades.Calculator do
75 75     end
76 76     end
77 77
78 78 -
78 78 + # Question 2.4 (Gabriel Cordovado)
79 79 + # The formula for calculating the number of labs is the same for each instance of percentage_grade
80 80 + # and letter grade so it would be better if we just encapsulated this data and swapped it in for

```

\*continuous on next page - screenshot too large\*

```

78 + # Question 2.4 (Gabriel Cordovado)
79 + # The formula for calculating the number of labs is the same for each instance of percentage_grade
80 + # and letter_grade so it would be better if we just encapsulated this data and swapped it in for
81 + # where the 3 repetative lines need to me for each area
82 + #
83 + #param list_lab : list of floats
84 + #return an integer representing the number of lab grades (floats) in the passed list
85 + #
86 + def count_labs(list_lab) do
87 +   list_lab
88 +   |> Enum.reject(fn mark -> mark < 0.25 end)
89 +   |> Enum.count()
90 + end
91 +
92
93 def percentage_grade(%{homework: homework, labs: labs, midterm: midterm, final: final}) do
94   avg_homework = avg(homework) # refractor 2.1
95
96
97 @-87,70 +100,31 @@ defmodule Grades.Calculator do
98
99 100
100 101 def letter_grade(%{homework: homework, labs: labs, midterm: midterm, final: final}) do
101 102   avg_homework = avg(homework) # refractor 2.1
102 103   -
103 104   avg_labs = avg(labs) # refractor 2.1
104 105   -
105 106   avg_exams = (midterm + final) / 2
106 107   -
107 108   num_labs =
108 109     labs
109 110     |> Enum.reject(fn mark -> mark < 0.25 end)
110 111     |> Enum.count()
111 112   num_labs = count_labs(labs) # refractor 2.4 - 3
112 113
113 114   if failed_to_participate(avg_homework, avg_exams, num_labs) do #refractor 2.2
114 115     "EIN"
115 116   else
116 117     mark = calculate_grade(avg_labs, avg_homework, midterm, final) # refractor 2.3
117 118
118 119     # cond do
119 120     #   mark > 0.895 -> "A+"
120 121     #   mark > 0.845 -> "A"
121 122     #   mark > 0.795 -> "A-"
122 123     #   mark > 0.745 -> "B+"
123 124     #   mark > 0.695 -> "B"
124 125     #   mark > 0.645 -> "C+"
125 126     #   mark > 0.595 -> "C"
126 127     #   mark > 0.545 -> "D+"
127 128     #   mark > 0.495 -> "D"
128 129     #   mark > 0.395 -> "E"
129 130     #   :else -> "F"
130 131   end
131 132   mark_output(mark, "letter")
132 133 end
133 134 end
134 135
135 136 def numeric_grade(%{homework: homework, labs: labs, midterm: midterm, final: final}) do
136 137   avg_homework = avg(homework) # refractor 2.1
137 138   -
138 139   avg_labs = avg(labs) # refractor 2.1
139 140   -
140 141   avg_exams = (midterm + final) / 2
141 142   -
142 143   num_labs =
143 144     labs
144 145     |> Enum.reject(fn mark -> mark < 0.25 end)
145 146     |> Enum.count()
146 147   num_labs = count_labs(labs) # refractor 2.4 - 3
147 148
148 149   if failed_to_participate(avg_homework, avg_exams, num_labs) do #refractor 2.2
149 150     0
150 151   else
151 152     mark = calculate_grade(avg_labs, avg_homework, midterm, final) # refractor 2.3
152 153
153 154     # cond do
154 155     #   mark > 0.895 -> 10
155 156     #   mark > 0.845 -> 9
156 157     #   mark > 0.795 -> 8
157 158     #   mark > 0.745 -> 7
158 159     #   mark > 0.695 -> 6
159 160     #   mark > 0.645 -> 5
160 161     #   mark > 0.595 -> 4
161 162     #   mark > 0.545 -> 3
162 163     #   mark > 0.495 -> 2
163 164     #   mark > 0.395 -> 1
164 165     #   :else -> 0
165 166   end
166 167   mark_output(mark, "number")
167 168 end
168 169 end
169 170 end
170 171 end
171 172 end
172 173 end
173 174 end
174 175 end
175 176 end
176 177 end
177 178 end
178 179 end
179 180 end
180 181 end
181 182 end
182 183 end
183 184 end
184 185 end
185 186 end
186 187 end
187 188 end
188 189 end
189 190 end
190 191 end
191 192 end
192 193 end
193 194 end
194 195 end
195 196 end
196 197 end
197 198 end
198 199 end
199 200 end
200 201 end
201 202 end
202 203 end
203 204 end
204 205 end
205 206 end
206 207 end
207 208 end
208 209 end
209 210 end
210 211 end
211 212 end
212 213 end
213 214 end
214 215 end
215 216 end
216 217 end
217 218 end
218 219 end
219 220 end
220 221 end
221 222 end
222 223 end
223 224 end
224 225 end
225 226 end
226 227 end
227 228 end
228 229 end
229 230 end
230 231 end
231 232 end
232 233 end
233 234 end
234 235 end
235 236 end
236 237 end
237 238 end
238 239 end
239 240 end
240 241 end
241 242 end
242 243 end
243 244 end
244 245 end
245 246 end
246 247 end
247 248 end
248 249 end
249 250 end
250 251 end
251 252 end
252 253 end
253 254 end
254 255 end
255 256 end
256 257 end
257 258 end
258 259 end
259 260 end
260 261 end
261 262 end
262 263 end
263 264 end
264 265 end
265 266 end
266 267 end
267 268 end
268 269 end
269 270 end
270 271 end
271 272 end
272 273 end
273 274 end
274 275 end
275 276 end
276 277 end
277 278 end
278 279 end
279 280 end
280 281 end
281 282 end
282 283 end
283 284 end
284 285 end
285 286 end
286 287 end
287 288 end
288 289 end
289 290 end
290 291 end
291 292 end
292 293 end
293 294 end
294 295 end
295 296 end
296 297 end
297 298 end
298 299 end
299 300 end
300 301 end
301 302 end
302 303 end
303 304 end
304 305 end
305 306 end
306 307 end
307 308 end
308 309 end
309 310 end
310 311 end
311 312 end
312 313 end
313 314 end
314 315 end
315 316 end
316 317 end
317 318 end
318 319 end
319 320 end
320 321 end
321 322 end
322 323 end
323 324 end
324 325 end
325 326 end
326 327 end
327 328 end
328 329 end
329 330 end
330 331 end
331 332 end
332 333 end
333 334 end
334 335 end
335 336 end
336 337 end
337 338 end
338 339 end
339 340 end
340 341 end
341 342 end
342 343 end
343 344 end
344 345 end
345 346 end
346 347 end
347 348 end
348 349 end
349 350 end
350 351 end
351 352 end
352 353 end
353 354 end
354 355 end
355 356 end
356 357 end
357 358 end
358 359 end
359 360 end
360 361 end
361 362 end
362 363 end
363 364 end
364 365 end
365 366 end
366 367 end
367 368 end
368 369 end
369 370 end
370 371 end
371 372 end
372 373 end
373 374 end
374 375 end
375 376 end
376 377 end
377 378 end
378 379 end
379 380 end
380 381 end
381 382 end
382 383 end
383 384 end
384 385 end
385 386 end
386 387 end
387 388 end
388 389 end
389 390 end
390 391 end
391 392 end
392 393 end
393 394 end
394 395 end
395 396 end
396 397 end
397 398 end
398 399 end
399 400 end
400 401 end
401 402 end
402 403 end
403 404 end
404 405 end
405 406 end
406 407 end
407 408 end
408 409 end
409 410 end
410 411 end
411 412 end
412 413 end
413 414 end
414 415 end
415 416 end
416 417 end
417 418 end
418 419 end
419 420 end
420 421 end
421 422 end
422 423 end
423 424 end
424 425 end
425 426 end
426 427 end
427 428 end
428 429 end
429 430 end
430 431 end
431 432 end
432 433 end
433 434 end
434 435 end
435 436 end
436 437 end
437 438 end
438 439 end
439 440 end
440 441 end
441 442 end
442 443 end
443 444 end
444 445 end
445 446 end
446 447 end
447 448 end
448 449 end
449 450 end
450 451 end
451 452 end
452 453 end
453 454 end
454 455 end
455 456 end
456 457 end
457 458 end
458 459 end
459 460 end
460 461 end
461 462 end
462 463 end
463 464 end
464 465 end
465 466 end
466 467 end
467 468 end
468 469 end
469 470 end
470 471 end
471 472 end
472 473 end
473 474 end
474 475 end
475 476 end
476 477 end
477 478 end
478 479 end
479 480 end
480 481 end
481 482 end
482 483 end
483 484 end
484 485 end
485 486 end
486 487 end
487 488 end
488 489 end
489 490 end
490 491 end
491 492 end
492 493 end
493 494 end
494 495 end
495 496 end
496 497 end
497 498 end
498 499 end
499 500 end
500 501 end
501 502 end
502 503 end
503 504 end
504 505 end
505 506 end
506 507 end
507 508 end
508 509 end
509 510 end
510 511 end
511 512 end
512 513 end
513 514 end
514 515 end
515 516 end
516 517 end
517 518 end
518 519 end
519 520 end
520 521 end
521 522 end
522 523 end
523 524 end
524 525 end
525 526 end
526 527 end
527 528 end
528 529 end
529 530 end
530 531 end
531 532 end
532 533 end
533 534 end
534 535 end
535 536 end
536 537 end
537 538 end
538 539 end
539 540 end
540 541 end
541 542 end
542 543 end
543 544 end
544 545 end
545 546 end
546 547 end
547 548 end
548 549 end
549 550 end
550 551 end
551 552 end
552 553 end
553 554 end
554 555 end
555 556 end
556 557 end
557 558 end
558 559 end
559 560 end
560 561 end
561 562 end
562 563 end
563 564 end
564 565 end
565 566 end
566 567 end
567 568 end
568 569 end
569 570 end
570 571 end
571 572 end
572 573 end
573 574 end
574 575 end
575 576 end
576 577 end
577 578 end
578 579 end
579 580 end
580 581 end
581 582 end
582 583 end
583 584 end
584 585 end
585 586 end
586 587 end
587 588 end
588 589 end
589 590 end
590 591 end
591 592 end
592 593 end
593 594 end
594 595 end
595 596 end
596 597 end
597 598 end
598 599 end
599 600 end
600 601 end
601 602 end
602 603 end
603 604 end
604 605 end
605 606 end
606 607 end
607 608 end
608 609 end
609 610 end
610 611 end
611 612 end
612 613 end
613 614 end
614 615 end
615 616 end
616 617 end
617 618 end
618 619 end
619 620 end
620 621 end
621 622 end
622 623 end
623 624 end
624 625 end
625 626 end
626 627 end
627 628 end
628 629 end
629 630 end
630 631 end
631 632 end
632 633 end
633 634 end
634 635 end
635 636 end
636 637 end
637 638 end
638 639 end
639 640 end
640 641 end
641 642 end
642 643 end
643 644 end
644 645 end
645 646 end
646 647 end
647 648 end
648 649 end
649 650 end
650 651 end
651 652 end
652 653 end
653 654 end
654 655 end
655 656 end
656 657 end
657 658 end
658 659 end
659 660 end
660 661 end
661 662 end
662 663 end
663 664 end
664 665 end
665 666 end
666 667 end
667 668 end
668 669 end
669 670 end
670 671 end
671 672 end
672 673 end
673 674 end
674 675 end
675 676 end
676 677 end
677 678 end
678 679 end
679 680 end
680 681 end
681 682 end
682 683 end
683 684 end
684 685 end
685 686 end
686 687 end
687 688 end
688 689 end
689 690 end
690 691 end
691 692 end
692 693 end
693 694 end
694 695 end
695 696 end
696 697 end
697 698 end
698 699 end
699 700 end
700 701 end
701 702 end
702 703 end
703 704 end
704 705 end
705 706 end
706 707 end
707 708 end
708 709 end
709 710 end
710 711 end
711 712 end
712 713 end
713 714 end
714 715 end
715 716 end
716 717 end
717 718 end
718 719 end
719 720 end
720 721 end
721 722 end
722 723 end
723 724 end
724 725 end
725 726 end
726 727 end
727 728 end
728 729 end
729 730 end
730 731 end
731 732 end
732 733 end
733 734 end
734 735 end
735 736 end
736 737 end
737 738 end
738 739 end
739 740 end
740 741 end
741 742 end
742 743 end
743 744 end
744 745 end
745 746 end
746 747 end
747 748 end
748 749 end
749 750 end
750 751 end
751 752 end
752 753 end
753 754 end
754 755 end
755 756 end
756 757 end
757 758 end
758 759 end
759 760 end
760 761 end
761 762 end
762 763 end
763 764 end
764 765 end
765 766 end
766 767 end
767 768 end
768 769 end
769 770 end
770 771 end
771 772 end
772 773 end
773 774 end
774 775 end
775 776 end
776 777 end
777 778 end
778 779 end
779 780 end
780 781 end
781 782 end
782 783 end
783 784 end
784 785 end
785 786 end
786 787 end
787 788 end
788 789 end
789 790 end
790 791 end
791 792 end
792 793 end
793 794 end
794 795 end
795 796 end
796 797 end
797 798 end
798 799 end
799 800 end
800 801 end
801 802 end
802 803 end
803 804 end
804 805 end
805 806 end
806 807 end
807 808 end
808 809 end
809 810 end
810 811 end
811 812 end
812 813 end
813 814 end
814 815 end
815 816 end
816 817 end
817 818 end
818 819 end
819 820 end
820 821 end
821 822 end
822 823 end
823 824 end
824 825 end
825 826 end
826 827 end
827 828 end
828 829 end
829 830 end
830 831 end
831 832 end
832 833 end
833 834 end
834 835 end
835 836 end
836 837 end
837 838 end
838 839 end
839 840 end
840 841 end
841 842 end
842 843 end
843 844 end
844 845 end
845 846 end
846 847 end
847 848 end
848 849 end
849 850 end
850 851 end
851 852 end
852 853 end
853 854 end
854 855 end
855 856 end
856 857 end
857 858 end
858 859 end
859 860 end
860 861 end
861 862 end
862 863 end
863 864 end
864 865 end
865 866 end
866 867 end
867 868 end
868 869 end
869 870 end
870 871 end
871 872 end
872 873 end
873 874 end
874 875 end
875 876 end
876 877 end
877 878 end
878 879 end
879 880 end
880 881 end
881 882 end
882 883 end
883 884 end
884 885 end
885 886 end
886 887 end
887 888 end
888 889 end
889 890 end
890 891 end
891 892 end
892 893 end
893 894 end
894 895 end
895 896 end
896 897 end
897 898 end
898 899 end
899 900 end
900 901 end
901 902 end
902 903 end
903 904 end
904 905 end
905 906 end
906 907 end
907 908 end
908 909 end
909 910 end
910 911 end
911 912 end
912 913 end
913 914 end
914 915 end
915 916 end
916 917 end
917 918 end
918 919 end
919 920 end
920 921 end
921 922 end

```