Devoir 2:

Auteur: Basma kaanane

Numero : 300142701

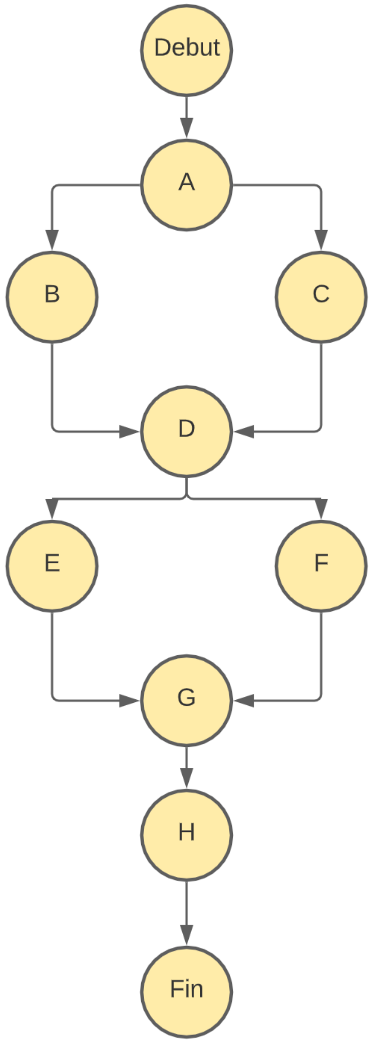
Professeur : Dr. Andrew Forward

TA: Aymen Mhamdi

Probleme1:

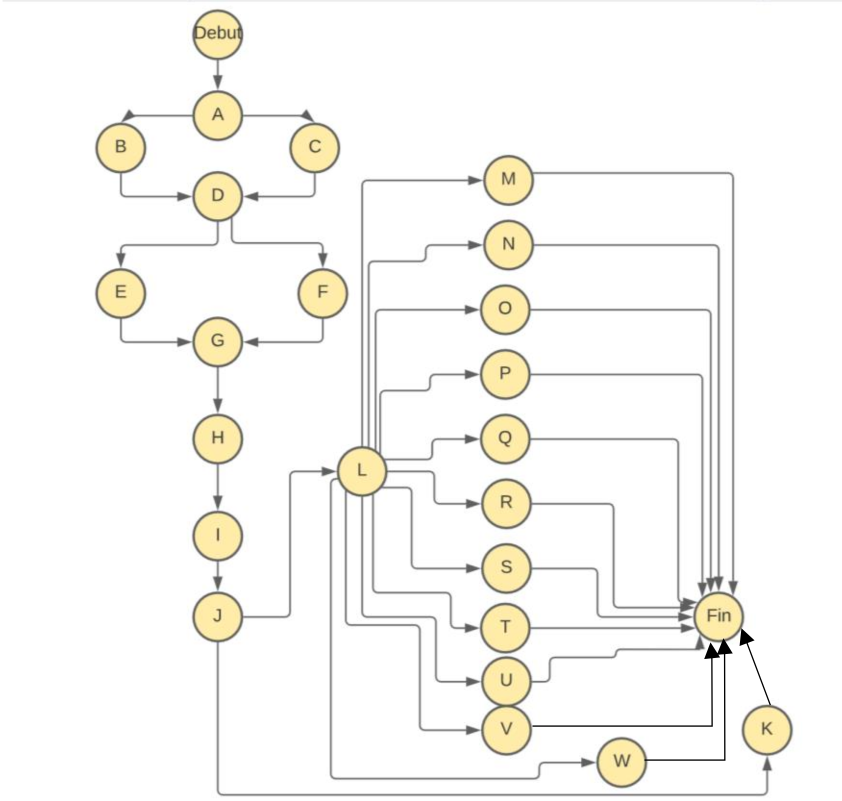
Question 1.1:

-
- Graphe de flot pour la méthode pourentage_grade



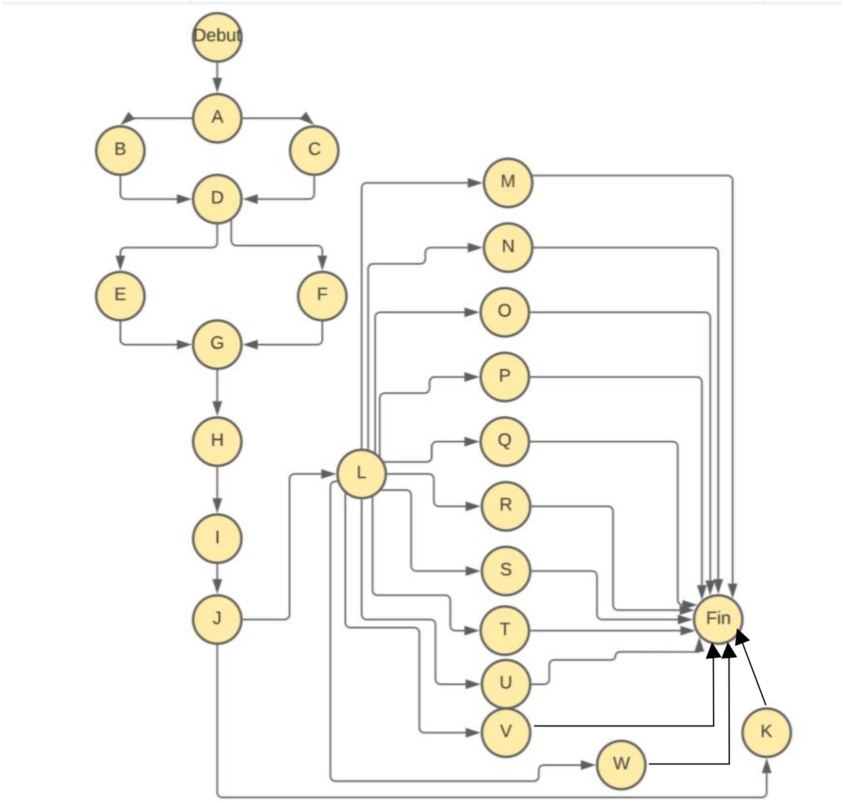
```
def percentage_grade(%{homework: homework,
labs: labs, midterm: midterm, final: final}) do
avg_homework =
   if Enum.count(homework) == 0 do A
     0 B
   else
     Enum.sum(homework) /
Enum.count(homework) C
   end

  avg_labs =
   if Enum.count(labs) == 0 do D
     0 E
   else
     Enum.sum(labs) / Enum.count(labs) F
   end

  mark = 0.2 * avg_labs + 0.3 * avg_homework + 0.2
* midterm + 0.3 * final G
  round(mark * 100) H
 end
```

- Graphe de flot pour la methode letter_grade



```
def letter_grade(%{homework: homework, labs: labs, midterm: midterm, final: final}) do
  avg_homework =
   if Enum.count(homework) == 0 do A
    0 B
   else
    Enum.sum(homework) / Enum.count(homework) C
   end

  avg_labs =
   if Enum.count(labs) == 0 do D
    0 E
   else
    Enum.sum(labs) / Enum.count(labs) F
   end
  avg_exams = (midterm + final) / 2 G

  num_labs =
    labs
    |> Enum.reject(fn mark -> mark < 0.25 end) H
    |> Enum.count() I

  if avg_homework < 0.4 || avg_exams < 0.4 || num_labs < 3 do J
    "EIN" K
  else
    mark = 0.2 * avg_labs + 0.3 * avg_homework + 0.2 * midterm + 0.3 * final L

    cond do
      mark > 0.895 -> "A+" M
      mark > 0.845 -> "A" N
      mark > 0.795 -> "A-" O
      mark > 0.745 -> "B+" P
      mark > 0.695 -> "B" Q
      mark > 0.645 -> "C+" R
      mark > 0.595 -> "C" S
      mark > 0.545 -> "D+" T
      mark > 0.495 -> "D" U
      mark > 0.395 -> "E" V
      :else -> "F" W
    end
  end
end
```

Graphe de flot pour la methode numeroc_grade



Question 1.2

Pourcentage_grade :

Cas de test	Entree pour le test	Résultats attedus	Conditions couvertes	Branches couivertes
1	Homework =[1] / lab =[0] / Mideterm=1 / final =1	80	AD	CEGH
2	Homework =[0] / lab =[1,1,1] / Mideterm=1 / final =1	70	AD	BFGH

Letter_grade :

Cas de test	Entree pour le test	Résultats attedus	Conditions couvertes	Branches couivertes
1	Homework =[0] / lab =[0] / Mideterm=0 / final =0	EIN	ADJW	BEGHIJKW
2	Homework =[1] / lab =[1,1,1] / Mideterm=1 / final =1	A+	ADJM	CFGHIJKLM
3	homework: [1], labs: [1, 1, 1], midterm: 1, final: 0.5	A	ADJN	CFGHIJLN
4	homework: [1], labs: [1, 1, 1], midterm: 1, final: 0.35	A-	ADJO	CFGHIJLO
5	homework: [1], labs: [1, 1, 1], midterm: 1, final: 0.2	B+	ADJP	CFGHIJLP
6	homework: [1], labs: [1, 1, 1], midterm: 1, final: 0.1	B	ADJP	CFGHIJLP
7	homework: [1], labs: [1, 1, 1], midterm: 0.9, final: 0	C+	ADJR	CFGHIJLR
8	homework: [0.7], labs: [1, 1, 1], midterm: 0.4, final: 0.4	C	ADJS	CFGHIJLS
9	homework: [0.6], labs: [1, 1, 1], midterm: 0.4, final: 0.4	D+	ADJT	CFGHIJLT
10	homework: [0.4], labs: [1, 1, 1], midterm: 0.4, final: 0.4	D	ADJU	CFGHIJLU
11	homework: [0.4], labs: [0.4, 0.4, 0.4], midterm: 0.4, final: 0.4	E	ADJV	CFGHIJLV

12	Cannot be reached	F	ADJW	CFGHIJLW

Numeric Grade :

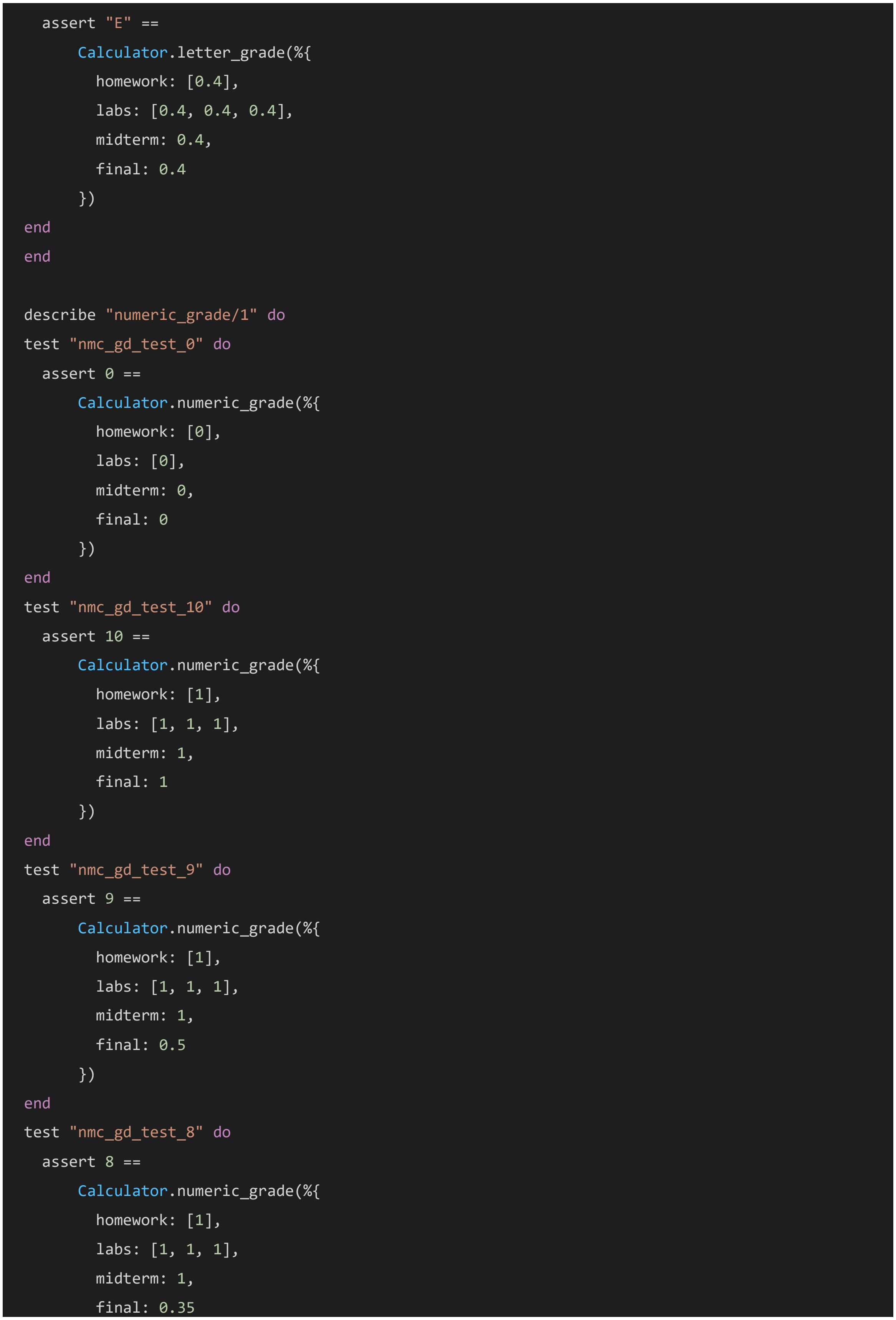
Cas de test	Entree pour le test	Résultats attedus	Conditions couvertes	Branches couivertes
1	Homework =[0] / lab =[0] / Mideterm=0 / final =0	0	ADJW	BEGHIJKW
2	Homework =[1] / lab =[1,1,1] / Mideterm=1 / final =1	10	ADJM	CFGHIJKLM
3	homework: [1], labs: [1, 1, 1], midterm: 1, final: 0.5	9	ADJN	CFGHIJLN
4	homework: [1], labs: [1, 1, 1], midterm: 1, final: 0.35	8	ADJO	CFGHIJLO
5	homework: [1], labs: [1, 1, 1], midterm: 1, final: 0.2	7	ADJP	CFGHIJLP
6	homework: [1], labs: [1, 1, 1], midterm: 1, final: 0.1	6	ADJP	CFGHIJLP
7	homework: [1], labs: [1, 1, 1], midterm: 0.9, final: 0	5	ADJR	CFGHIJLR
8	homework: [0.7], labs: [1, 1, 1], midterm: 0.4, final: 0.4	4	ADJS	CFGHIJLS
9	homework: [0.6], labs: [1, 1, 1], midterm: 0.4, final: 0.4	3	ADJT	CFGHIJLT
10	homework: [0.4], labs: [1, 1, 1], midterm: 0.4, final: 0.4	2	ADJU	CFGHIJLU

11	homework: [0.4], labs: [0.4, 0.4, 0.4], midterm: 0.4, final: 0.4	1	ADJV	CFGHIJLV

Question 1.3

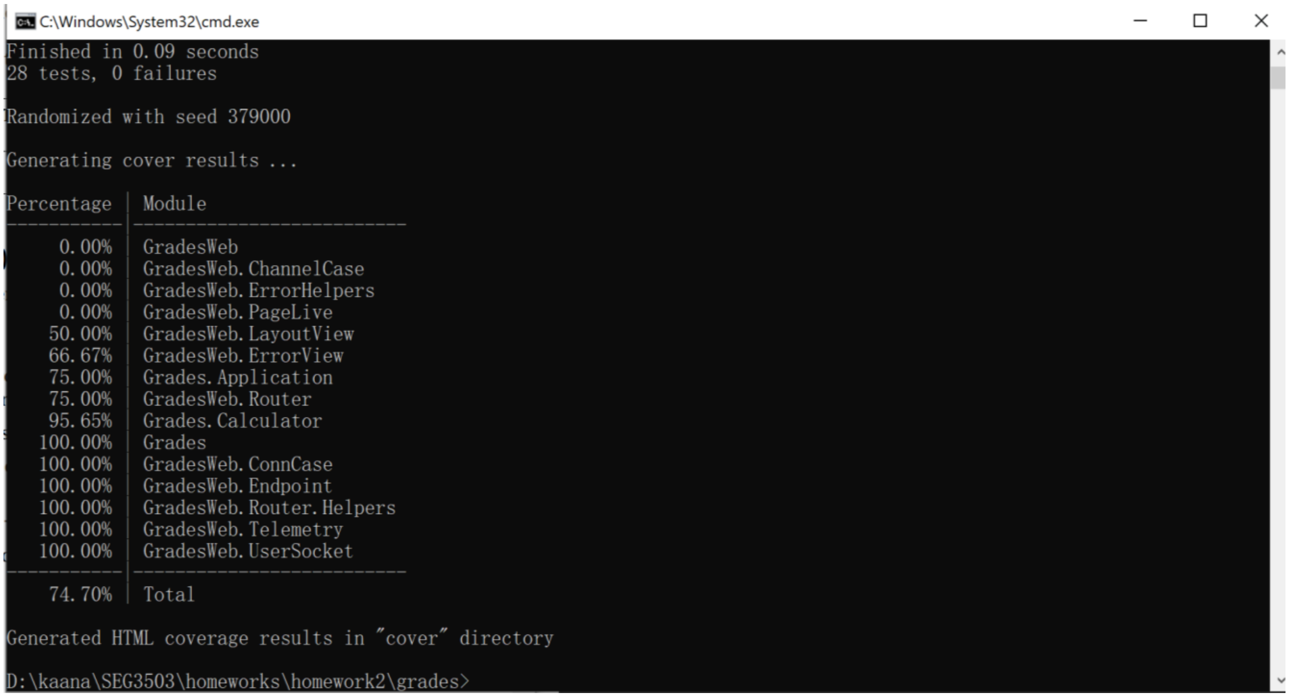
```
defmodule Grades.CalculatorTest do
  use ExUnit.Case
  alias Grades.Calculator

  describe "percentage_grade/1" do
    test "sample" do
      assert 85 ==
               Calculator.percentage_grade(%{
                 homework: [0.8],
                 labs: [1, 1, 1],
                 midterm: 0.70,
                 final: 0.9
               })
    end
    test "pctg_gd_test_80" do
    assert 80 ==
        Calculator.percentage_grade(%{
          homework: [1],
          labs: [0],
          midterm: 1,
          final: 1
        })
  end
  test "pctg_gd_test_70" do
    assert 70 ==
        Calculator.percentage_grade(%{
          homework: [0],
          labs: [1, 1, 1],
          midterm: 1,
          final: 1
        })
  end
  end

  describe "letter_grade/1" do
  test "ltr_gd_test_EIN" do
    assert "EIN" ==
        Calculator.letter_grade(%{
          homework: [0],
          labs: [0],
          midterm: 0,
          final: 0
        })
  end
  test "ltr_gd_test_A+" do
    assert "A+" ==
        Calculator.letter_grade(%{
          homework: [1],
          labs: [1, 1, 1],
          midterm: 1,
          final: 1
        })
  end
  test "ltr_gd_test_A" do
    assert "A" ==
        Calculator.letter_grade(%{
          homework: [1],
          labs: [1, 1, 1],
          midterm: 1,
          final: 0.5
        })
  end
  test "ltr_gd_test_A-" do
    assert "A-" ==
        Calculator.letter_grade(%{
          homework: [1],
          labs: [1, 1, 1],
          midterm: 1,
          final: 0.35
        })
  end
  test "ltr_gd_test_B+" do
    assert "B+" ==
        Calculator.letter_grade(%{
          homework: [1],
          labs: [1, 1, 1],
          midterm: 1,
          final: 0.2
        })
  end
  test "ltr_gd_test_B" do
```

```
  assert "B" ==
      Calculator.letter_grade(%{
        homework: [1],
        labs: [1, 1, 1],
        midterm: 1,
        final: 0.1
      })
end
test "ltr_gd_test_C+" do
  assert "C+" ==
      Calculator.letter_grade(%{
        homework: [1],
        labs: [1, 1, 1],
        midterm: 0.9,
        final: 0
      })
end
test "ltr_gd_test_C" do
  assert "C" ==
      Calculator.letter_grade(%{
        homework: [0.7],
        labs: [1, 1, 1],
        midterm: 0.4,
        final: 0.4
      })
end
test "ltr_gd_test_D+" do
  assert "D+" ==
      Calculator.letter_grade(%{
        homework: [0.6],
        labs: [1, 1, 1],
        midterm: 0.4,
        final: 0.4
      })
end
test "ltr_gd_test_D" do
  assert "D" ==
      Calculator.letter_grade(%{
        homework: [0.4],
        labs: [1, 1, 1],
        midterm: 0.4,
        final: 0.4
      })
end
test "ltr_gd_test_E" do
```



```
  assert "E" ==
      Calculator.letter_grade(%{
        homework: [0.4],
        labs: [0.4, 0.4, 0.4],
        midterm: 0.4,
        final: 0.4
      })
end
end

describe "numeric_grade/1" do
test "nmc_gd_test_0" do
  assert 0 ==
      Calculator.numeric_grade(%{
        homework: [0],
        labs: [0],
        midterm: 0,
        final: 0
      })
end
test "nmc_gd_test_10" do
  assert 10 ==
      Calculator.numeric_grade(%{
        homework: [1],
        labs: [1, 1, 1],
        midterm: 1,
        final: 1
      })
end
test "nmc_gd_test_9" do
  assert 9 ==
      Calculator.numeric_grade(%{
        homework: [1],
        labs: [1, 1, 1],
        midterm: 1,
        final: 0.5
      })
end
test "nmc_gd_test_8" do
  assert 8 ==
      Calculator.numeric_grade(%{
        homework: [1],
        labs: [1, 1, 1],
        midterm: 1,
        final: 0.35
      })
end
test "nmc_gd_test_7" do
  assert 7 ==
      Calculator.numeric_grade(%{
        homework: [1],
        labs: [1, 1, 1],
        midterm: 1,
        final: 0.2
      })
end
test "nmc_gd_test_6" do
  assert 6 ==
      Calculator.numeric_grade(%{
        homework: [1],
        labs: [1, 1, 1],
        midterm: 1,
        final: 0.1
      })
end
test "nmc_gd_test_5" do
  assert 5 ==
      Calculator.numeric_grade(%{
        homework: [1],
        labs: [1, 1, 1],
        midterm: 0.9,
        final: 0
      })
end
test "nmc_gd_test_4" do
  assert 4 ==
      Calculator.numeric_grade(%{
        homework: [0.7],
        labs: [1, 1, 1],
        midterm: 0.4,
        final: 0.4
      })
end
test "nmc_gd_test_3" do
  assert 3 ==
      Calculator.numeric_grade(%{
        homework: [0.6],
        labs: [1, 1, 1],
        midterm: 0.4,
        final: 0.4
        })
  end
  test "nmc_gd_test_2" do
    assert 2 ==
        Calculator.numeric_grade(%{
          homework: [0.4],
          labs: [1, 1, 1],
          midterm: 0.4,
          final: 0.4
        })
  end
  test "nmc_gd_test_1" do
    assert 1 ==
        Calculator.numeric_grade(%{
          homework: [0.4],
          labs: [0.4, 0.4, 0.4],
          midterm: 0.4,
          final: 0.4
        })
  end
  end
end
```

Question 1.4



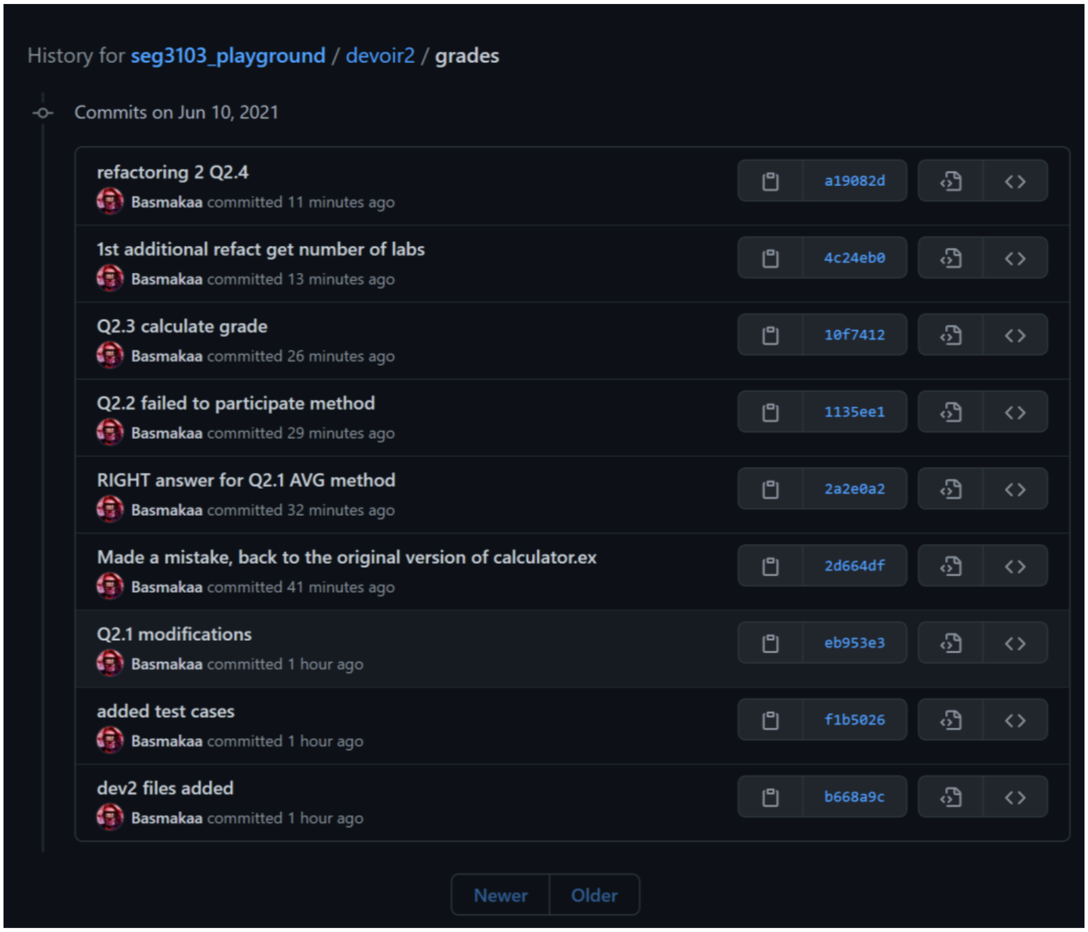
```
Finished in 0.09 seconds
28 tests, 0 failures

Randomized with seed 379000

Generating cover results ...

Percentage | Module
-----------|--------------------------
     0.00% | GradesWeb
     0.00% | GradesWeb.ChannelCase
     0.00% | GradesWeb.ErrorHelpers
     0.00% | GradesWeb.PageLive
    50.00% | GradesWeb.LayoutView
    66.67% | GradesWeb.ErrorView
    75.00% | Grades.Application
    75.00% | GradesWeb.Router
    95.65% | Grades.Calculator
   100.00% | Grades
   100.00% | GradesWeb.ConnCase
   100.00% | GradesWeb.Endpoint
   100.00% | GradesWeb.Router.Helpers
   100.00% | GradesWeb.Telemetry
   100.00% | GradesWeb.UserSocket
-----------|--------------------------
    74.70% | Total

Generated HTML coverage results in "cover" directory

D:\kaana\SEG3503\homeworks\homework2\grades>
```

La couverture est de 95,65%, je ne suis pas en mesure d'atteindre une couverture de 100%, parce que pour letter_grade et numeric_grade, si je veux vérifier les branches ":else -> "F" " et ":else -> 0", alors les devoirs, les laboratoires, les examens de mi-session et la finale, au moins l'un d'entre eux doit être inférieur à 0,395. être inférieur à 0,395, et les trois autres doivent être égaux à 0,395. Et cela va le programme, et ne vérifiera pas les conditions et les branches de M à W. le programme ira à la branche K et renverra directement le "EIN" ou 0, puis arrêtera le programme.

Question 2 :

Voir l'ensemble du code dans le sous repertoire devoir 2. Lien : https://github.com/Basmakaa/seg3103_playground.git

Vous pouvez checkez l'historique de commits avec toutes les modifications



History for seg3103_playground / devoir2 / grades

Commits on Jun 10, 2021

refactoring 2 Q2.4
Basmakaa committed 11 minutes ago
a19082d

1st additional refact get number of labs
Basmakaa committed 13 minutes ago
4c24eb0

Q2.3 calculate grade
Basmakaa committed 26 minutes ago
10f7412

Q2.2 failed to participate method
Basmakaa committed 29 minutes ago
1135ee1

RIGHT answer for Q2.1 AVG method
Basmakaa committed 32 minutes ago
2a2e0a2

Made a mistake, back to the original version of calculator.ex
Basmakaa committed 41 minutes ago
2d664df

Q2.1 modifications
Basmakaa committed 1 hour ago
eb953e3

added test cases
Basmakaa committed 1 hour ago
f1b5026

dev2 files added
Basmakaa committed 1 hour ago
b668a9c