BİL 113/012 Computer Programming I

HOMEWORK 3 (40 Points)

To be demonstrated by November 9th

1 [10 POINTS] GAMBLER'S RUIN

In this question, you are going to simulate the *Gambler's Ruin*, a statistical concept that is most commonly expressed as the fact that a gambler playing a negative expected value game will eventually go broke, regardless of their betting system.

We have a gambler that came to casino with $\$n$ in his pocket. He wants to make his money $\$m$ for some $m > n$, by playing as many rounds as needed of a particular game. In each round of the game, the gambler bets \$1, and wins \$1 with some probability p or loses \$1 with probability $1-p$. Notice that at the end of a round, the gambler has one more dollar than the end of the previous round with probability p, and he has one less dollar than the end of the previous round with probability $1-p$. The gambler plays this game until one of the following two things happens: (i) his money reaches the goal he set before coming to the casino, i.e., his money becomes $\$m$, which we refer to as a *win* for him, (ii) his money reaches \$0 and he is not permitted to play any more rounds, and we refer to this situation as a *lose* for him. As you will see, she has a very slim chance of winning if m is large for $p < 0.5$.

As simulations involve randomness, you will simulate this game several times and count how many times the gambler won and lost. Your program must take the following as input and output how many times the gambler won and lost.

- n : the initial money of the gambler,
- m : the goal of the gambler,
- p : the probability of winning at each round, and

- the number of times the game is to be simulated. Note that one simulation of the game consists of several rounds.

You can find example program inputs and outputs below. Green texts are inputs. **Since randomness is involved in the computation, the output of your program may not match the given output for these examples, however, the results should be similar.**

EXAMPLE 1
Initial money:100
Goal:200
Probability of winning in a round:0.49
Number of times the game will be simulated:1000
Win:18
Lose:982

EXAMPLE 2
Initial money:100
Goal:200
Probability of winning in a round:0.495
Number of times the game will be simulated:1000
Win:127
Lose:873

EXAMPLE 3
Initial money:500
Goal:1000
Probability of winning in a round:0.490
Number of times the game will be simulated:1000
Win:0
Lose:1000

EXAMPLE 4
Initial money:500
Goal:1000
Probability of winning in a round:0.495
Number of times the game will be simulated:1000
Win:0
Lose:1000

EXAMPLE 5
Initial money:500
Goal:1000
Probability of winning in a round:0.499

Number of times the game will be simulated:1000
Win:125
Lose:875

EXAMPLE 6
Initial money:100
Goal:200
Probability of winning in a round:0.51
Number of times the game will be simulated:1000
Win:979
Lose:21

2 [10 POINTS] DEPT PAYMENT

You have just purchased a new laptop for your Bil113 course. Laptop costs $1,500 on the following credit plan: no down payment, an interest rate of 18% per year (and hence 1.5% per month), and monthly payments of $50. The monthly payment of $50 is used to pay the interest, and whatever is left is used to pay part of the remaining debt. Hence, the first month you pay 1.5% of $1,500 in interest. That is $22.5 in interest. So, the remaining $27.50 is deducted from your debt, which leaves you with a debt of $1472.50. The next month, you pay interest of 1.5% of $1472.50, which is $22,0875. Hence, you can deduct $27.9125 (which is $50–$22,0875) from the amount you owe. This goes on until you have no debt.
In this task you will write a program that tells you how many months it will take you to pay off a loan with the given interest under equal payments each month. Also, you will calculate the total payment paid. Use a loop to calculate the amount of interest and the size of the debt after each month. (Your final program need not output the monthly amount of interest paid and remaining debt, but you may want to write a preliminary version of the program that does output these values.) Use a variable to count the number of loop iterations and hence the number of months until the debt is zero. You may want to use other variables as well. The last payment may be less than the given monthly payment if the debt is small, but do not forget the interest! If you owe $50, your monthly payment of $50 will not pay off your debt, although it will come close. One month's interest on $50 is only 75 cents.

You may assume that monthly payments are greater than the first month's interest.

EXAMPLE 1
Please enter the initial debt:1000
Please enter the yearly interest rate (in percentages):24
Please enter the monthly payment you want to make:50
It takes 26 months to pay the debt and the total payment equals to $1,289.87

EXAMPLE 2
Please enter the initial debt:1000
Please enter the yearly interest rate (in percentages):12
Please enter the monthly payment you want to make:50
It takes 23 months to pay the debt and the total payment equals to $1,121.35

EXAMPLE 3
Please enter the initial debt:1000
Please enter the yearly interest rate (in percentages):12
Please enter the monthly payment you want to make:25
It takes 52 months to pay the debt and the total payment equals to $1,283.47

EXAMPLE 4
Please enter the initial debt:1000
Please enter the yearly interest rate (in percentages):12
Please enter the monthly payment you want to make:12.5
It takes 162 months to pay the debt and the total payment equals to $2,021.85

EXAMPLE 5
Please enter the initial debt:1600
Please enter the yearly interest rate (in percentages):18
Please enter the monthly payment you want to make:25
It takes 217 months to pay the debt and the total payment equals to $5,404.96

3 [20 POINTS] FRACTIONAL NUMBERS

In this task, you are going to write a Java class named ***FractionalNumber*** which will support various operations on fractional numbers such as addition, subtraction, etc.
Fractional numbers consists of an integer *numerator* and *denominator*. So your class should have them as instance variables. You can find the methods that you need to implement below.

- *FractionalNumber* class should have two constructors. One of them should take a String as a parameter. The other one should take two integers (numerator and denominator) as a parameter.
- public double getDoubleValue(): This method should return the double value of this fractional number, e.g., if this fractional number is 4/5, then this method should return 0.8.
- public String toString(): This method should return string representation of this fractional number which is formatted as " *numerator*/*denominator*". For example if the *numerator* is 4 and *denominator* is 5, then method should return "4/5".
- public boolean equals(FractionalNumber fn2): This method should return true if this FractionalNumber is equivalent to fn2.

- public FractionalNumber simplify(): This method should return the simplified version of this fractional number. For example, if the fractional number is $4/8$, then this method should return the FractionalNumber $1/2$. If the FractionalNumber is negative, then the $-$ sign should always be located before the numerator. For example, if the FractionalNumber is $2/-4$, then this method should return the FractionalNumber $-1/2$. Hint: You need to find the greatest common divisor of numerator and denominator to implement this method. This method should be used inside the four methods below, so make sure that you write this method correctly, otherwise, you may receive a low score.

- public FractionalNumber add(FractionalNumber fn2): This method should return the simplified version of the summation of this FractionalNumber and fn2. For example, if this fractional number is $4/8$ and fn2 is $2/8$ then this method should return the FractionalNumber $3/4$.

- public FractionalNumber subtract(FractionalNumber fn2): This method should return the simplified version of the difference between this fractional number and fn2. For example, if this fractional number is $4/8$ and fn2 is $2/8$ then this method should return the FractionalNumber $1/4$.

- public FractionalNumber multiply(FractionalNumber fn2): This method should return the simplified version of the product of this fractional number and fn2. For example, if this fractional number is $4/8$ and fn2 is $2/8$ then this method should return the FractionalNumber $1/8$.

- public FractionalNumber divide(FractionalNumber fn2): This method should return the simplified version of the division of this fractional number with fn2. If fn2 is equal to 0 then this method should return null. For example, if this fractional number is $4/8$ and fn2 is $2/8$ then this method should return the FractionalNumber $2/1$. But if this fractional number is $4/8$ and fn2 is $0/8$ then this method should return null.

We shared a demo class named ***FractionalNumberDemo***. Put this file to the same folder with your FractionalNumber.java file. You can use this class to test correctness of your implementation. If you could not implement some methods before submitting your homework, write methods with same signature and make them return the default value of the return type. For example if you could not implement the subtract method, then write

```
public FractionalNumber subtract(FractionalNumber fn2){
        return null;
}
```

Otherwise, your code will not compile. Example outputs are given below. Green texts represent input.

EXAMPLE 1

```
Please enter the first fractional number in string representation
3/6
Please enter the second fractional number in string representation
7/9
Please enter the third fractional number in numerator and denominator form
2 4
First fractional number is 3/6 and its double value is 0.5
Second fractional number is 7/9 and its double value is 0.7777777777777778
Third fractional number is 2/4 and its double value is 0.5

First fractional number is 3/6 and its simplified version is 1/2
Second fractional number is 7/9 and simplified version is 7/9
Third fractional number is 2/4 and simplified version is 1/2

First and second fractional numbers are not equal
First and third fractional numbers are equal
Second and third fractional numbers are not equal

3/6 + 7/9 = 23/18
3/6 + 2/4 = 1/1
7/9 + 2/4 = 23/18

3/6 - 7/9 = -5/18
3/6 - 2/4 = 0/1
7/9 - 2/4 = 5/18

3/6 * 7/9 = 7/18
3/6 * 2/4 = 1/4
2/4 * 7/9 = 7/18

3/6 / 7/9 = 9/14
3/6 / 2/4 = 1/1
2/4 / 7/9 = 9/14
```

EXAMPLE 2

```
Please enter the first fractional number in string representation
2/4
Please enter the second fractional number in string representation
0/9
Please enter the third fractional number in numerator and denominator form
3 -6
First fractional number is 2/4 and its double value is 0.5
Second fractional number is 0/9 and its double value is 0.0
Third fractional number is 3/−6 and its double value is −0.5

First fractional number is 2/4 and its simplified version is 1/2
Second fractional number is 0/9 and simplified version is 0/1
Third fractional number is 3/−6 and simplified version is −1/2

First and second fractional numbers are not equal
First and third fractional numbers are not equal
Second and third fractional numbers are not equal

2/4 + 0/9 = 1/2
2/4 + 3/−6 = 0/1
0/9 + 3/−6 = −1/2

2/4 − 0/9 = 1/2
2/4 − 3/−6 = 1/1
0/9 − 3/−6 = 1/2

2/4 * 0/9 = 0/1
2/4 * 3/−6 = −1/4
3/−6 * 0/9 = 0/1

2/4 / 0/9 = null
2/4 / 3/−6 = −1/1
3/−6 / 0/9 = null
```