ELEC 2543 Object-Oriented Programming and Data Structures

Programming Exercise 4

Topics: Data Visibility, Libraries

Released: Feb 14, 2025

Due: 9:30 am, Feb 24, 2025

Overview: In this exercise, you are going to develop a simple random walk game. You have to apply the appropriate visibility modifiers for all variables and methods.

The Random Walk Game

*Overview*

There is a linear path with *length* units long. A player starts in Position 1. In each round, he moves forward a few units. If he lands on the last position of the path, the game ends. If when he is close to the end of the path but the number of units he has to move is more than needed, he “bounds” back and continues to walk towards the end of the path.

*Details*

The path spans from 1 to *length*. In the beginning, the player is at position 1. In each round:

1. A map showing the current positions of the player is printed
2. the distance to move is selected from [1, 3]
3. a message showing the move details is printed out
4. update the new position of the player
5. game ends if the player lands at exactly the last position on the path
6. A message telling how many rounds have gone through is printed out.

*Sample Output*

P......... [Player moves 2 unit(s)]

..P....... [Player moves 3 unit(s)]

.....P.... [Player moves 3 unit(s)]

........P. [Player moves 3 unit(s)]

.......P.. [Player moves 1 unit(s)]

........P. [Player moves 1 unit(s)]

The game ends after 6 rounds.

The above is a sample output when *length* = 10. ‘P’ tells the position of the player. In Line 4 (highlighted in red), player is supposed to move forward by 3 units. However, there are not enough rooms, and the player was push backward. The game ends only when the player lands exactly at the last position as shown in the last line.

Implementation

In Object-Oriented programming, each type of entities is modeled through a class. We are going to develop this game using four classes: a class for generating random numbers, a class for the die needed in the game, a class for the game logic, and a class for the driver program.

Class MyRandom: This class is provided to generate random numbers. You should not modify this class.

Method nextInt returns a random integer between *[1, max]*.

Class MyDie: This class represents the die to be used in the game to determine the distance the player moves in each round.

Note that in this game, the distance is between 1 to 3. Therefore, the sample class Die provided cannot be used directly. We now develop a class of dies that allows both the minimum face value and the maximum face value to be configured. The skeleton has been provided.

Instance variables min and max represent the minimum and maximum face values this die can have, respectively. Instance variable rnd points to a MyRandom object for generating random numbers. The constructor and method roll() have been defined for you. You should follow the instructions in the comments to implement method genFaceValue(). No other methods/instance variables can be defined.

You have to put in appropriate visibility modifiers (private or public) for each instance variable and method, even the method/variable has been defined for you.

Class RandomWalkGame: This class represents one complete game that starts with the player is at position 1.

Two instance variables have been declared for the length of the path and current position of the player (playerPos). You cannot define other instance variables. Method void printMap() has been provided to print a map showing the current location of the player. Work on the following:

* Define the constructor such that the length of the path is passed as parameter. Both length and playerPos should be initialized.
* Use the appropriate visibility modifier for all the instance variables and methods
* Implement method void play() that simulates the game and produces the output. You have to use an appropriate MyDie object to generate the distance in each move. When you print the messages about the moves (those in square brackets in the sample output), use the following:

System.out.println("\t[Player moves " + <your variable for distance> + " unit(s)]");

Class RandomWalkGameDriver: This class is the driver program and is the starting point of the whole application. You should not modify this class. The current driver assumes *length*=10.

Handin

Submit your MyDie.java and RandomWalkGame.java to Moodle before the deadline. We will use our own MyRandom and RandomWalkGameDriver to test your submission. You can assume we will not use an integer smaller than 2 as *length*.