Project#2: Monty Hall Game

Due Dates:

- **Algorithm: February 15, 2009 @ 11:55 p.m.**
- **Java Program: February 20, 2009 @ 11:55 p.m.**

This project is based on the famous game show hosted by Monty Hall. For this project, you will design this game where the user will choose the door that he wants to open, and if the user chooses the door with the prize (s)he wins.

This is how the game should be designed:

1. Your application should randomly choose one of the three doors to have the prize behind it. Assume that the other two doors are empty.
2. Ask the user to choose a door to open.
3. If the user chooses to open the door behind which contains the prize, then open any one of the other doors **at random** and ask if the user wants to switch the door. If the user says yes then change the user's choice to the closed empty door.
4. If the user chooses to open a door which is empty, then open the other empty door and ask if the user wants to switch the door. If the user says yes then change the user's choice to the door containing the prize.
5. If the user's choice of door is same as the door containing the prize, then display appropriate message and say You Win!! Otherwise, print that You Loose!!
6. Your program should only run a single game.

Take note that if the user wants to change his or her guess, you must change the state of the program accordingly. For example, if door 2 contains the prize and the user initially chooses door 3, then you should open door 1 and ask the user whether (s)he wants to switch her/his choice. If the user chooses to switch the door, the user's choice changes from door 3 (initial choice) to door 2 (the other closed door).

Your input checking should be somewhat robust. When asked if the user wants to switch the door, it should accept "y", "Y", "yes", "YES", "n", "N", "no", or "NO" as an answer.

Part I

- For this project, you must write an algorithm, i.e., sequence of steps to solve the project according to the rules indicated above.
- Please check the attached algorithm as an example of what you should do for this part.
- Submit your algorithm via WebCT.

Part II

- Write a java program that implements the algorithm to play the Monty Hall game. The name of the class in your java program must be *MontyHallGame*. Therefore, the java source code file must be called **MontyHallGame.java**

Sample Output

In the following examples, the text in green displays program output, the text in purple corresponds to the answers entered by the user.

Example #1:

```
Which door do you want to open? (1,2 or 3) : 1
The door 2 is EMPTY, do you want to switch your choice?(yes/no)yes
You choose to open door 3
The door 3 contains the prize!!
You Win!!
```

Example #2:

```
Which door do you want to open? (1,2 or 3) : 1
The door 2 is EMPTY, do you want to switch your choice?(yes/no)y
You choose to open door 3
The door 3 is Empty!! The door that has the prize is door 1
You Loose!!
```

Example #3:

```
Which door do you want to open? (1,2 or 3) : 3
The door 2 is EMPTY, do you want to switch your choice?(yes/no)NO
You choose to open door 3
The door 3 is Empty!! The door that has the prize is door 1
You Loose!!
```

Example #4:

```
Which door do you want to open? (1,2 or 3) : 1
The door 2 is EMPTY do you want to switch your choice?(yes/no)n
You choose to open door 1
The door 1 contains the prize!!
You Win!!
```

Hints

- Use the method random of the class Math to compute a random number between 1 and 3 in the following manner:

```
int aRandomNumber;
```

```
aRandomNumber =(int) (Math.random()*3) + 1;
```

The method random of the class Math returns a double random number greater than or equal to zero and less than 1. Multiplying the random number returned by the random method by 3 and casting its result as an integer will generate an integer between 0 and 2. After adding 1 to this result, we will get a random integer between 1 and 3.

Requirements

- The written algorithm should be a Word document describing the steps that should be followed to implement the logic of the Monty Hall game. Submit your algorithm to WebCT before **Sunday, February 15, 2009 @ 11:55 p.m.**
- You should try to make your program output look just like the examples above.

Other Requirements

Your code must include a comment header like the following in every project you submit.

```
/*
* MontyHallGame.java
* Author: Amy Smith
* Last edited: 02/19/2009
*
* Purpose: A brief one or two paragraph description of the
* program. What does it do? How does it do it?
*
* Statement of Academic Honesty:
*
* The following code represents my own work. I have neither
* received nor given inappropriate assistance. I have not copied
* or modified code from any source other than the course webpage
* or the course textbook. I recognize that any unauthorized
* assistance or plagiarism will be handled in accordance with
* the University of Georgia's Academic Honesty Policy and the
* policies of this course.
*/
```

Project Submission

- Submit your algorithm on WebCT.
- Submit the file MontyHallGame.java on WebCT

Project Grading

All projects are graded out of possible 100 points. In this project, 10 will be for your algorithm, 70 points will be for program correctness and 20 points for programming style. The following specific criteria will be applied to grade this project:

If your code does not compile, you will get a 0. Make absolutely certain your code compiles before you submit it via WebCT.

Algorithm (10 points)

Program Style (20 points):

- 1 point for program header.
- 2 points for helpful comments.
- 2 points for using mnemonic well-named variables.
- 2 points for correct code indentation and readability.
- 10 points for using the appropriate decision statements to implement the game.
- 3 points for using the appropriate assignment and output statements.

Program Correctness (70 points):

Your program will be tested thoroughly to verify if it determines the outcome of the game correctly. Furthermore, your program should display all the required information:

- prompts the user to input his/her guess.
- displays the door opened after the user has selected a door.
- asks the user if he/she wants to change his/her guess.
- displays the final user's choice.
- if the user does not pick the correct door, the program will display the door that has the prize.
- The program displays whether the user "wins" or "losses" correctly.
- Additionally, your program should accept and handle any of the following answers to the question that prompts the user if (s)he wants to change his/her choice.