

**SCHOOL OF DIGITAL MEDIA AND INFOCOMM TECHNOLOGY (DMIT)
DIPLOMA IN INFORMATION TECHNOLOGY**

ST0113 OBJECT-ORIENTED PROGRAMMING

**2009/2010 SEMESTER 1
PRACTICAL ASSIGNMENT**

Instructions and Guidelines:

1. The assignment should be submitted by **11 August 2009 (Tue), 9:00 am**. You are required to submit a hardcopy program listing to your Lecturer and a softcopy of source codes in BlackBoard. Provide your Class Group, Admission Number(s) and Name(s) on the softcopy and hardcopy.
2. Students are encouraged to work in pairs.
3. Marks will be awarded separately for each student in a pair, depending on his contribution to the assignment. The assignment will account for **30%** of your final grade.
4. The development platform will be in Java using NetBeans IDE.
5. The interview will be conducted during the practical lessons from **11 August 2009 to 14 August 2009**. You are expected to explain the program logic and modify the program during the interview. **If you are absent from the interview, you will be awarded zero mark for the assignment.**
6. **No marks will be awarded**, if the work is copied or you have allowed others to copy your work.
7. Marks will be deducted for late submission.

Paper Scissors Stone Game

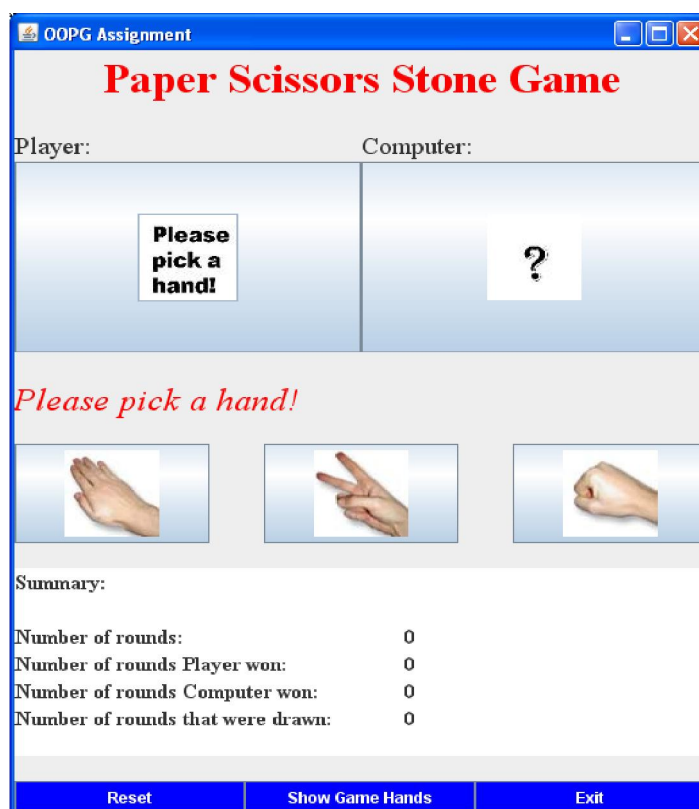
1. Objective:

The objective of this assignment is to create an application that simulates “Paper Scissors Stone” game which encompasses GUI concepts, event handling and object-oriented programming techniques.

2. Overview of the System

You are tasked to develop a “Paper Scissors Stone” game that allows a player to play the game against the computer.

The figure below shows a sample of what the game may look like.



3. Basic features of the system

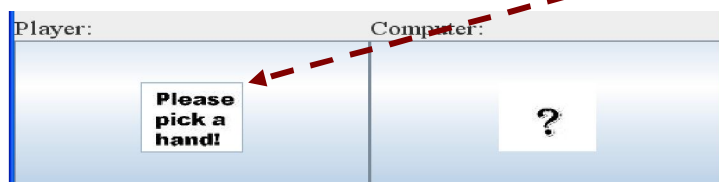
The system should provide the following features:

- (i) The title of the system.

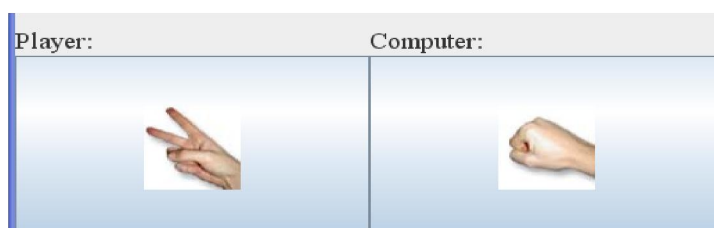
Paper Scissors Stone Game

- (ii) A panel that displays player's hand and computer's hand.

Before the player picks a hand, it displays the instruction:



After the player picks a hand, it displays the hands for both player and computer:



- (iii) A panel that displays the three hands that player can pick.



- (iv) A text area that displays the summary of the game.

Summary:	
Number of rounds:	0
Number of rounds Player won:	0
Number of rounds Computer won:	0
Number of rounds that were drawn:	0

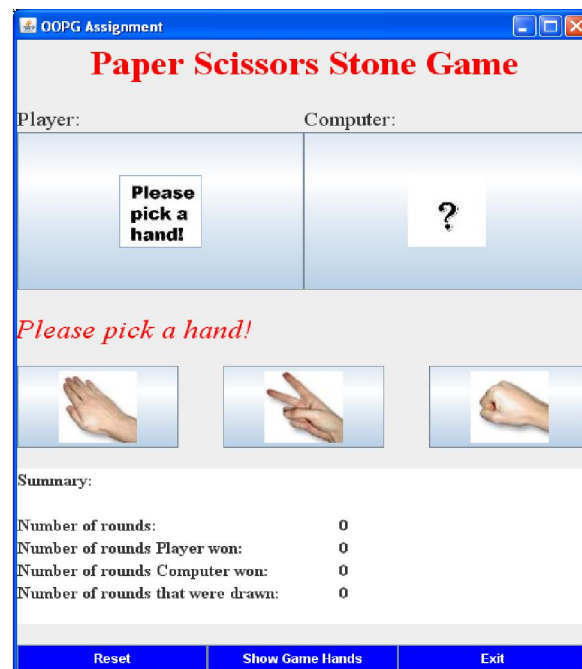
- (v) Three buttons: **Reset** → Reset the Summary statistics to 0 for all (as stated above)

Exit → Exit the game and close the application

Show Game Hands → Display the hands of player and computer of the game.



- (vi) When the application is launched, it displays the frame as shown:



- (vii) When the player picks a hand (e.g. picks stone), it randomly generates a hand for computer and displays the hands for both player and computer.



Reports the winner for this round based on Chart 3.1.

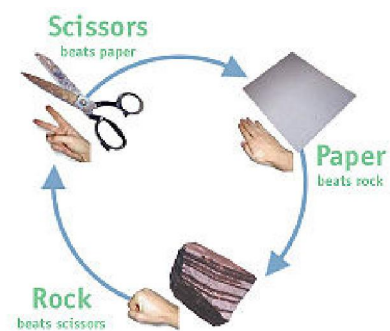
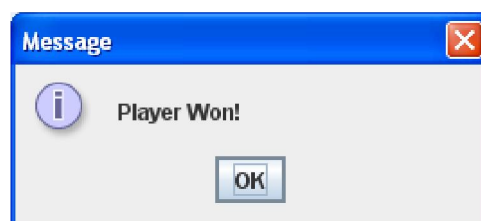


Chart 3.1

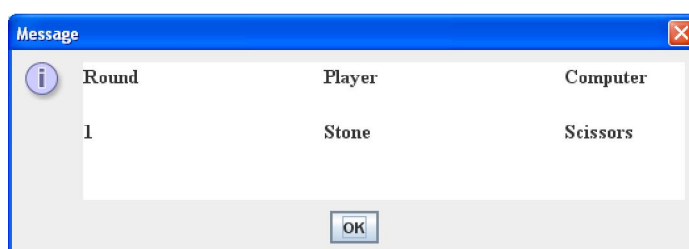
Update the Summary statics.

Summary:	
Number of rounds:	1
Number of rounds Player won:	1
Number of rounds Computer won:	0
Number of rounds that were drawn:	0

- (viii) When the **Reset** button is clicked. It resets the summary statistics to 0 for all.

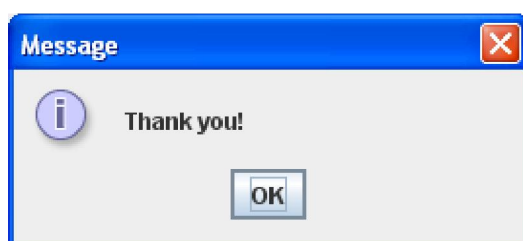
Summary:	
Number of rounds:	0
Number of rounds Player won:	0
Number of rounds Computer won:	0
Number of rounds that were drawn:	0

- (ix) When the **Show Game Hands** button is clicked, it displays the hands of player and computer for all rounds. (The following screen shot shows the hands for the game when user only plays one round).



Round	Player	Computer
1	Stone	Scissors

- (x) When the **Exit Game** button is clicked, it should display “Thank you” message and terminate the application



4. **Basic Requirements:**

- a) Write a Java class ***Game*** to represent the game. e.g. Instances variables that stores no. of rounds game played, no. of rounds player/computer wins...etc. The class should have methods that will be invoked by the ***GameFrame*** class to run the game. e.g Generate hand for computer, save player's and computer's hands, generate game statistics, etc.

Add a constructor to initialize the game and appropriate ***get*** methods to the class.

- b) Write a Java class ***GameFrame*** for the GUI in the application. You are free to create your own layout for the GUI.
- c) Write a Java class ***GameFrameUser*** that contains the ***main()*** method for the entire application.
- d) Classes designed should be modular and have appropriate methods for code clarity. e.g, you may have methods to create the GUI's different panels, or methods to perform each different operation. Do not squeeze all code into just one method.
- e) You should use arrays for coding efficiency.

5. **Advanced Features**

Bonus marks will be awarded for advanced features, such as, but not limited to the following:

- Allow customization of the GUI components such as changing the text color, background color of the panels and buttons, etc.
- Additional GUI features such as JMenuBar, JComboBox, JScrollBar etc.
- Allow selection of different scissors, stone, paper pattern.
- Sound effects.
- Store the game statistics in the file.
- Track the time when the player starts and exits the game.
- Other features that enhances the game.

Please keep in mind that all these are just bonus features. The main bulk of marks are allocated to the completion of a workable program that meets the minimum requirements. You should try to fulfil the minimum requirements before you attempt to include any advance features.

6. **Assessment Guidelines**

The assignment will be assessed based on the following criteria:

- Ability to demonstrate the minimum requirements stated above
- User friendliness:
 - Ease of use
 - Attractive User Interface
- Program design:
 - Correct and efficient usage of classes and programming constructs
 - Appropriate method decomposition
 - Appropriate validations
 - Code efficiency
- Program readability:
 - Meaningful identifiers
 - Meaningful comments and indentation in source code
- Innovation and creativity, and/or any advanced features
- Amount of individual contribution to the project
- Question & Answer during interview

-- End --