**Project Report**

**for**

**Application Pragramming Technique**

**Kathmandu University**

**-**

**Slide Puzzle Game**

**Submitted By:**

Ramesh Maharjan

Binod Sharma

Shanta Acharya

**Submitted To:**

Sagun Dhakhwa

**Contents**

**Chapter 1: Introduction 1**

**Chapter 2: Background theory 2**

**Chapter 3: Internal Structures 4**

**Chapter 4: Graphical User Interfaces (GUIs) 7**

**Chapter 5: Conclusion 10**

**List of figures**

**fig 2.1 general MVC diagram 3**

**fig 3.1 UML diagram of slide game puzzle 5**

**fig 3.2control flow in slide puzzle game 6**

**fig 4.1 level choose options 7**

**fig 4.2 mode choose option 7**

**fig 4.3 beginner level 8**

**fig 4.4 intermediate level 8**

**fig 4.5 advanced level 8**

**fig 4.6 name entering option for finishing the game in minimum moves 8**

**fig 4.7 score board of slide puzzle game 9**

**fig 4.8 thank you message 9**

**Chapter 1**

**Introduction**

Java is a vast and influential programming language which is widely used to develop software, applications and games for different purposes. Lack of proper frameworks or design patterns would increase confusions and complexities in Java as it is a wide collection of utilities, interfaces, internal classes and other frameworks. Use of design patterns simplifies the complexities that arise in the development process using java. The motto of this project and the course itself is to be acquainted with different design patterns and their implementations.

**Slide puzzle game** is one of the brain testing games played (especially by kids). The game can be of varying size with numbered blocks to be arranged in different orders or it may contain a pictured blocks to be arranged to produce original picture. The game can be shuffled and played as many times as possible. The shuffled blocks are slide to occupy the empty space and to reach the desired arrangement if numbers are used in the slides else to match the original picture if image is used in the slides.

The game becomes more interesting when players tries to minimize the number of moves needed to rearrange the shuffled slides and break previous records. Specification of time in the game would increase the difficulty level. Inclusion of different levels would make the game challenging for the players.

The slide puzzle game project tries to accumulate all the functionalities and qualities explained above to meet the aim of the course.

The following chapters tries to explain in more detail about the project and the design patterns utilized in the project where chapter 2 tries to explain the general definition of design pattern used, chapter three explains the actual pattern used in the project and chapter 4 explains the UIs of the project.

**Chapter 2**

**Background Theory**

This chapter tries to explain the background theory of the project: Slide Puzzle Game, which use Model, view and controller (MVC) as the design pattern. Each part is clearly explained.

**Java MVC (Model View Controller) Design Pattern**

Model View controller is a classical design pattern used in applications who needs a clean separation between their business logic and view who represents data. Model View Controller (MVC) pattern creates applications that separate the different aspects of the application (input logic, business logic, and UI logic), while providing a loose coupling between these elements.

MVC design pattern isolates the application logic from the user interface and permitted the individual development, testing and maintenance for each component. This design pattern is divided into three parts.

**Model**

This component manages the information and notify the observers when the information changes. It represents the data on which the application operates. The model provides the persistent storage of data, which manipulated by the controller.

**View**

 The view displays the data, and also takes input from user. It renders the model data into a form to display to the user. There can be several view associated with a single model. It is actually representation of model data.

**Controller**

The controller handles all requests coming from the view or user interface. The data flow to whole application is controlled by controller. It forwards the request to the appropriate handlers. Only the controller is responsible for accessing model and rendering it into various User Interfaces.

The MVC abstraction can be graphically represented as follows.

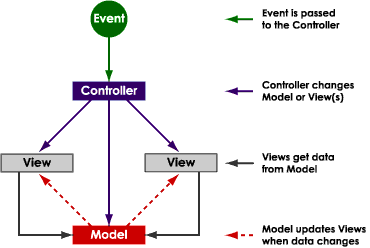


fig 2.1 general MVC diagram

Events typically cause a controller to change a model, or view, or both. Whenever a controller changes a model’s data or properties, all dependent views are automatically updated. Similarly, whenever a controller changes a view, for example, by revealing areas that were previously hidden, the view gets data from the underlying model to refresh itself.

**Chapter 3**

**Internal Structure**

This chapter tries to explain the internal structure i.e. the functions of the classes used. As it is evident that the slide game puzzle used MVC as a design pattern, we try to explain the internal structure based on the pattern used.

**View**

The view part of the slide game puzzle includes four classes: Main, GUI, GUI2 and GUI3. All the displays are controlled by these classes.

**Main**

It is the main class to start the application. The options for choosing the game levels and the game modes are all defined in this class.

**GUI**

The selection of “drag and drop” mode starts this class which defines all the functions necessary for drag and drop mode. The functions to start a new game and the functions to choose a new image are also defined in this class. Functions to paint the cropped images and the full image in the display are also done in this class. Finally, functions to display the winning score after the completion of a game session and if broken the previous record, storing the name of the player with the score are also defined in this class.

**GUI2**

The selection of “mouse move” mode starts this class which defines all the necessary functions required for moving slides by just moving mouse over the slides to be slide in the empty slide. As in the GUI class, new game and new image functions are present in this class for starting new game and for selecting image of choice to be selected for the game. The painting functions and the criteria for winning score are also same as in GUI class.

**GUI3**

The selection of “mouse press” mode starts this class which is almost similar to GUI2 class except some functions that enable users to move slides adjacent to empty slide by pressing mouse on the slide to be moved.

**Model**

This class is responsible for acting as an interface between control and view of the design pattern we followed for this project. Every actions updated by the view part is notified to the control and every information from control part are transferred to the view by the Model part. All the modifications needed and asked by the view part are also done in this class.

**Control**

This part contains two classes: Tile, which is used to update the rows, columns and images of each slide, and CropImage class, which is responsible for cropping image for each slide.

The following UML diagram tries to illustrate the details explained above where all the important functions of each class are included in MVC design form.

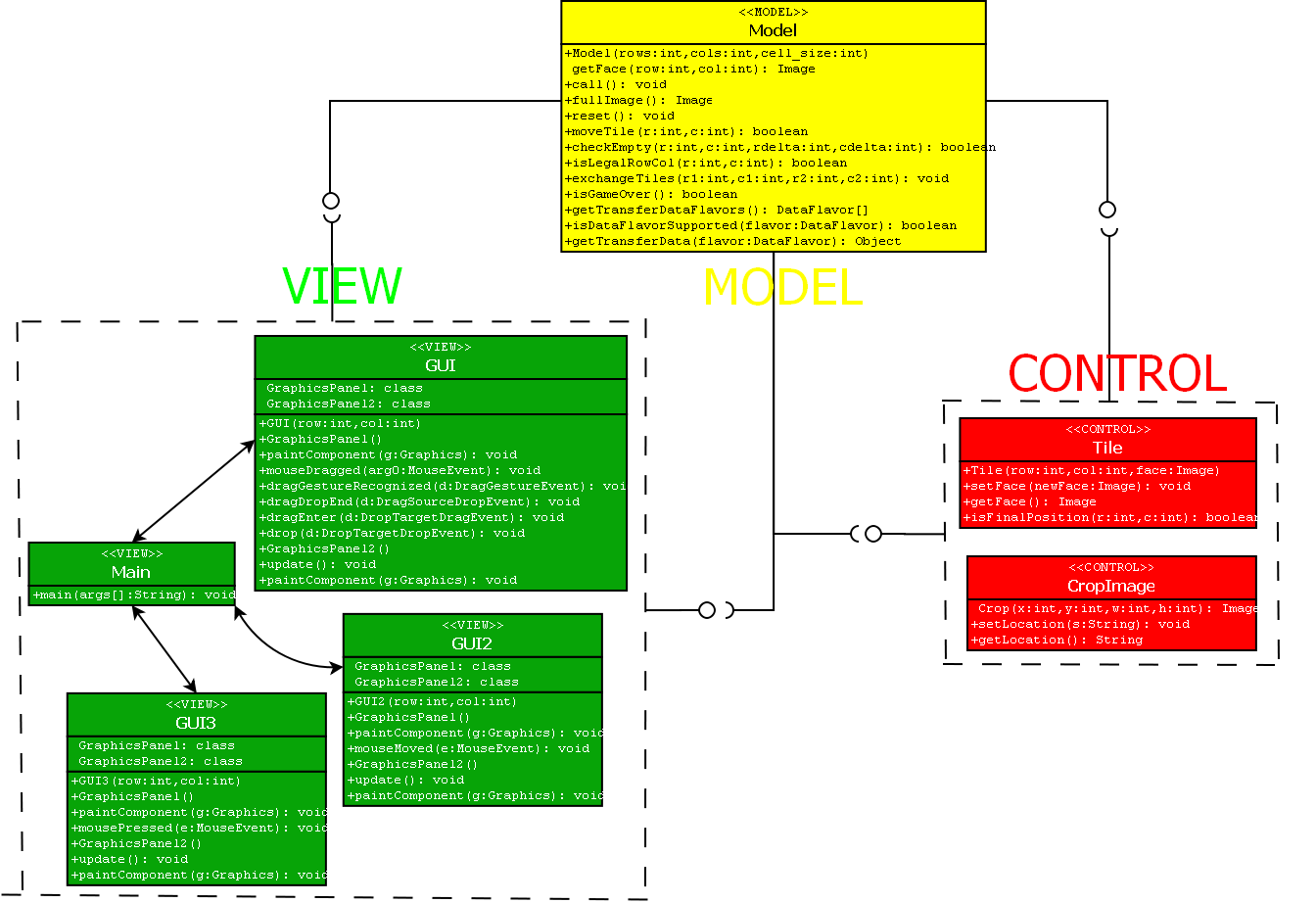


fig 3.1 UML diagram of slide game puzzle

The following flow control diagram explains the actions flow in the slide game puzzle application.

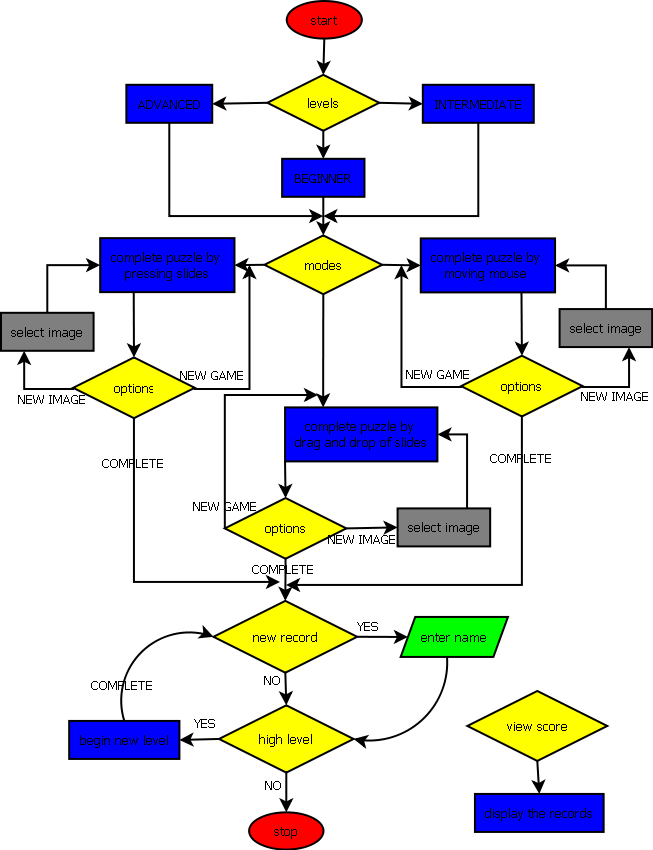


fig 3.2 control flow in slide puzzle game

**Chapter 4**

**Graphical User Interfaces (GUIs)**

This chapter tries to explain the GUI of slide puzzle game step by step.

When the application is run we see a GUI as in fig 4.1 which allows user to begin with an appropriate level. After a level is chosen, level options are disabled, and the mode option is enabled to choose from as shown in fig 4.2. Choosing an appropriate mode starts a new game.

Enabled levels to choose

disabled levels

Enabled modes to choose

disabled modes

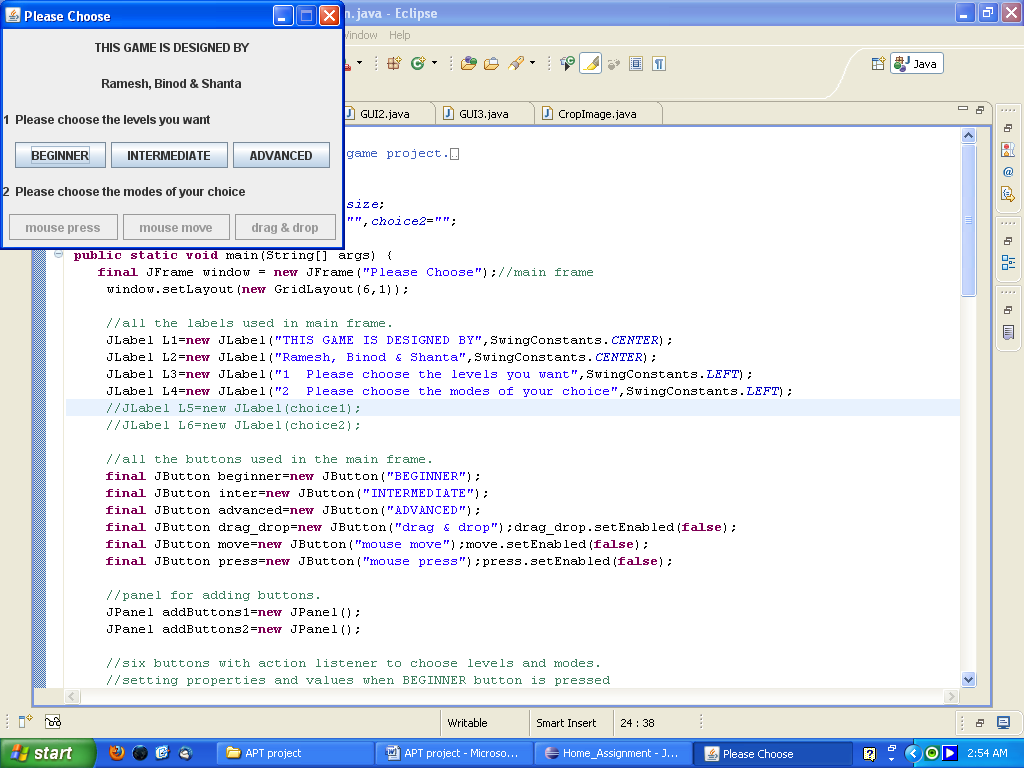
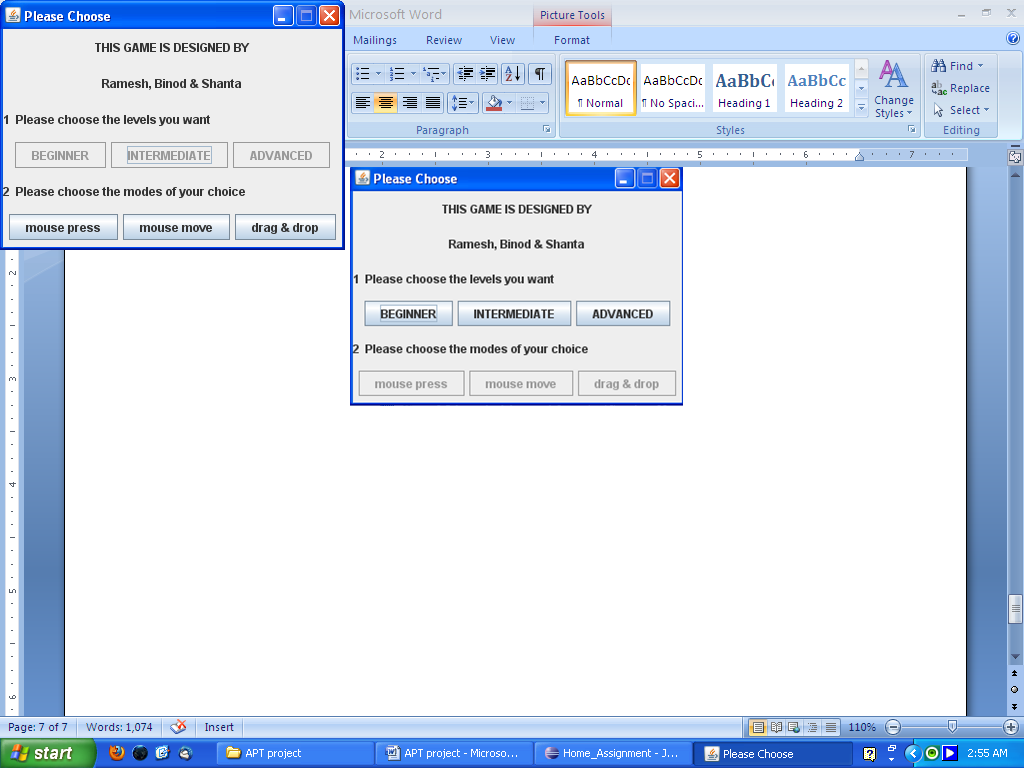
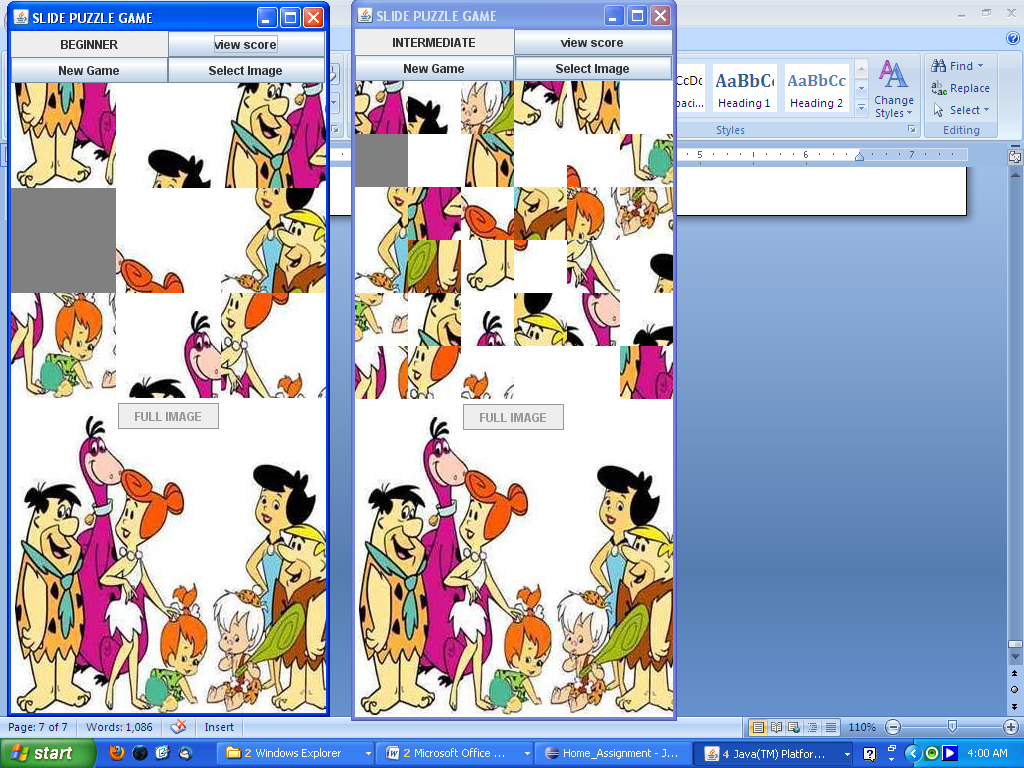
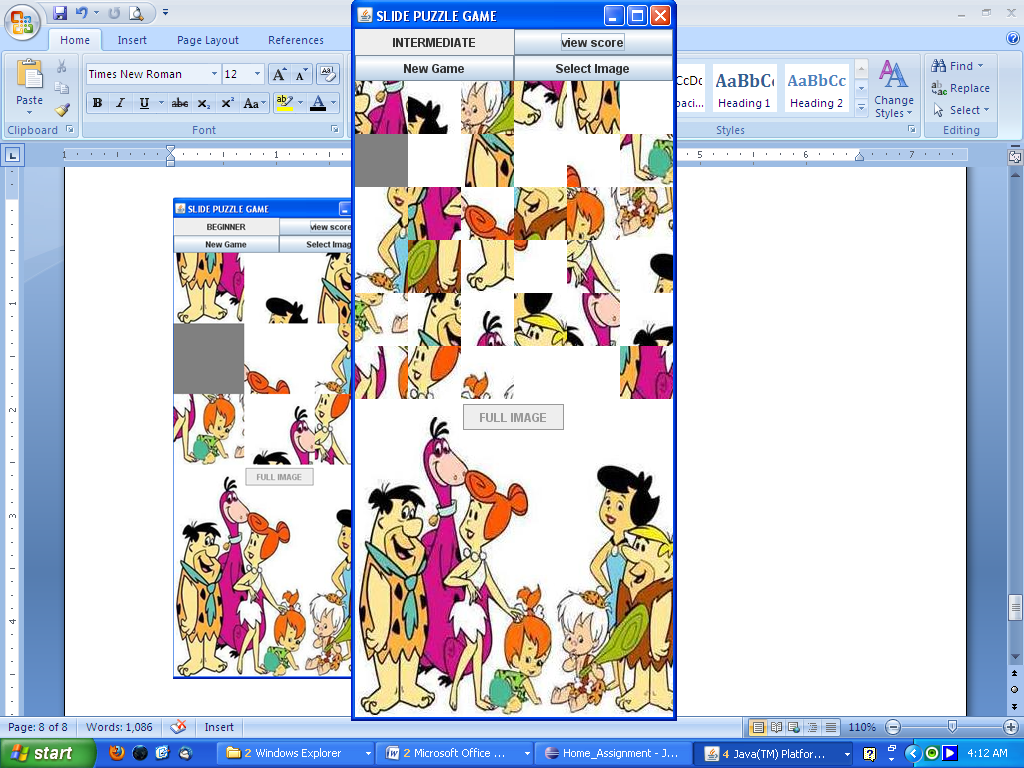
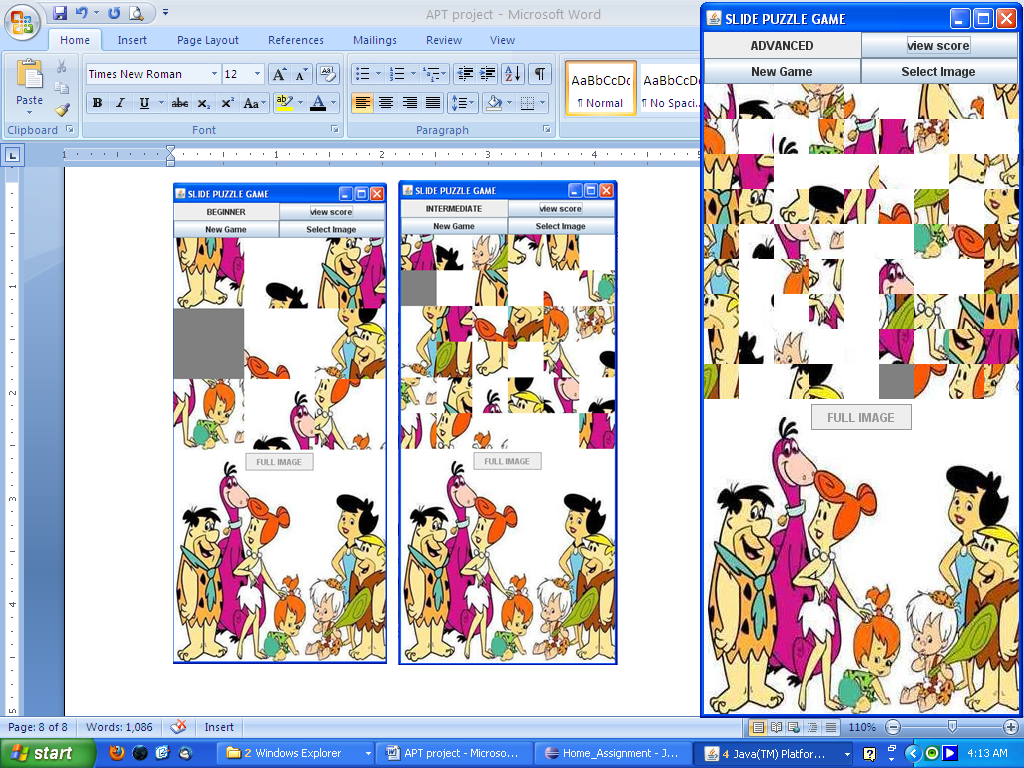
 

fig 4.1 level choose options fig 4.2 mode choose option

The new game consists three buttons, one shuffled area to be sorted, a full image area for reference and the level name to tell the user of the level being played as seen fig 4.3, fig 4.4 and fig 4.5. The beginner mode consists of 3Χ3 matrix of shuffled slides to be sorted to match with the full image while the intermediate mode consists of 6Χ6 matrix of shuffled slides and advanced mode of 9Χ9 matrix of shuffled slides. All of these levels can be played with different chosen modes.

New Game button shuffles the slides and starts a new game and Select Image button allows users to select a new image of their choice.

Level of the game

Three buttons

Shuffled slides to be sorted

Full image for reference

fig 4.3 beginner level fig 4.4 intermediate level fig 4.5 advanced level

Players who manage to finish a level in minimum moves than the previous one then the player is allowed to enter their names as shown in fig 4.6 where the user finished beginner mode with minimum moves than the previous game.

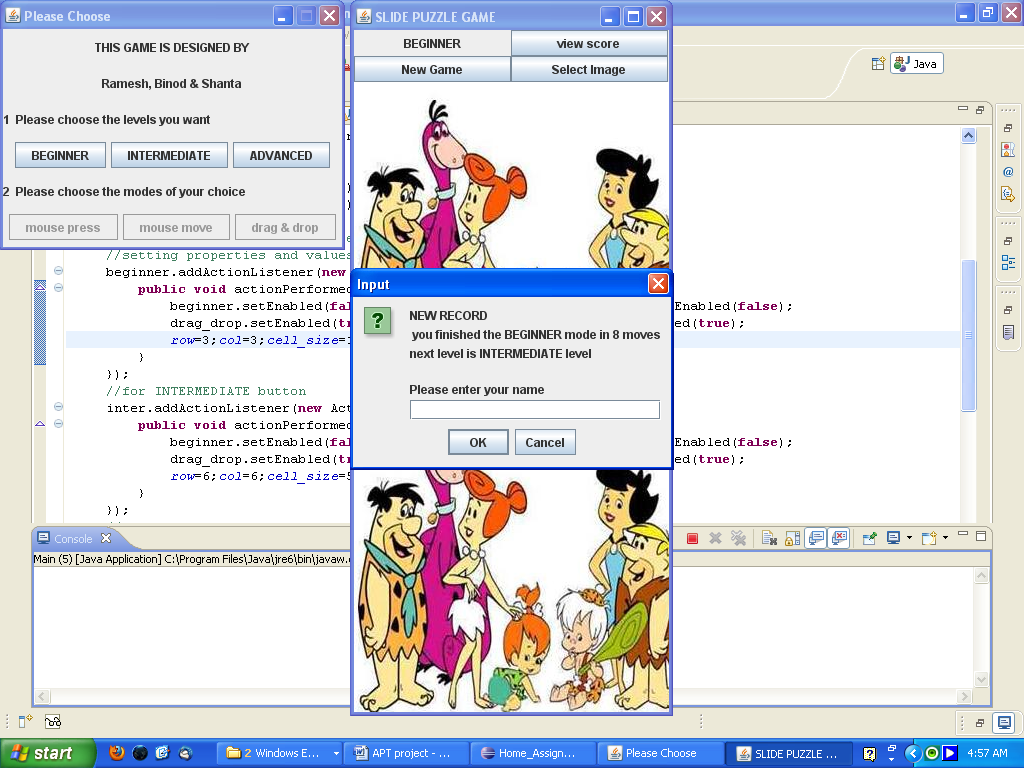


fig 4.6 name entering option for finishing the game in minimum moves

These records are stored for each level and for each mode which can be viewed by pressing view score button which displays the score as shown in fig 4.7.

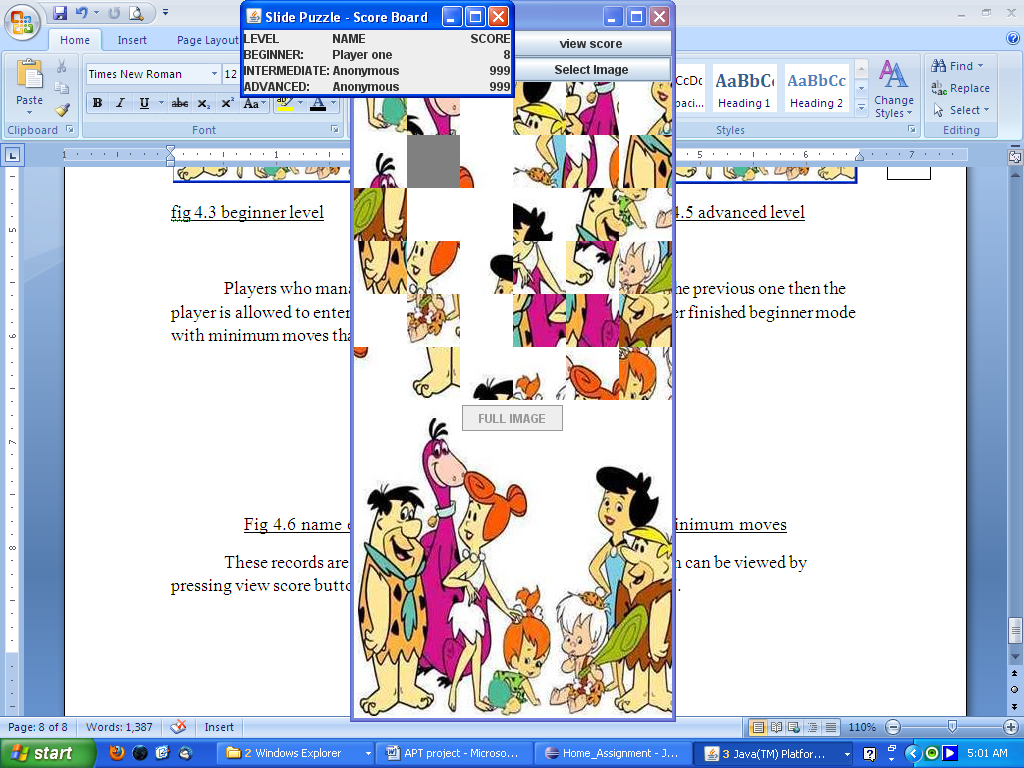
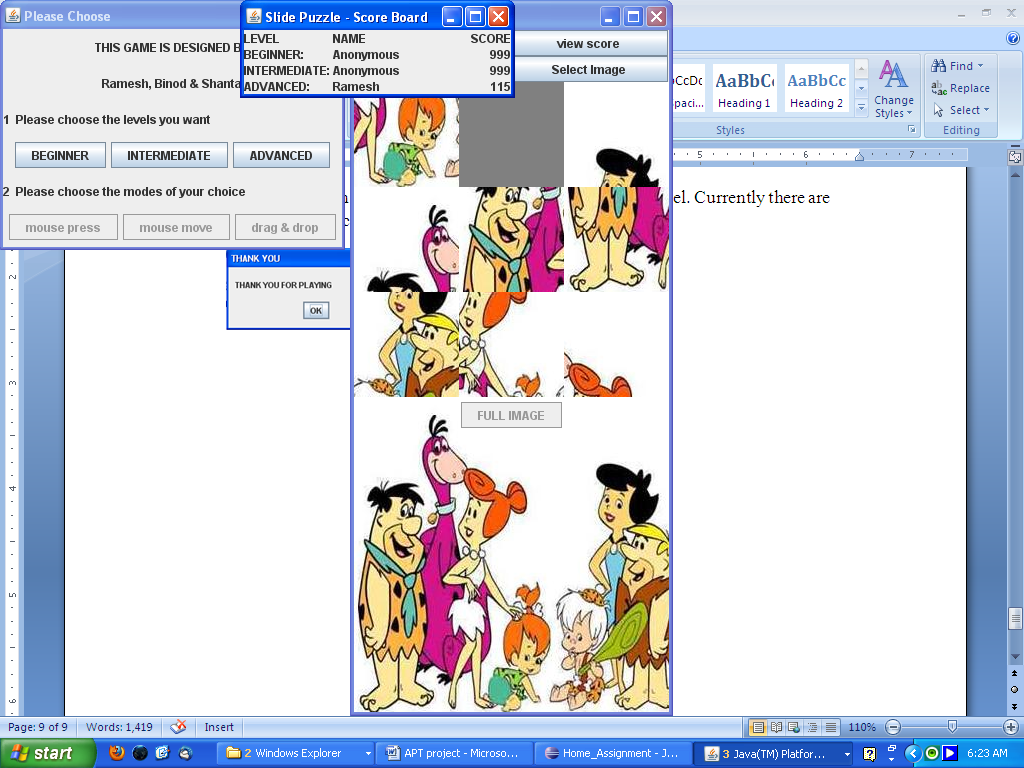
 

fig 4.7 score board of slide puzzle game

After completion of each level, the game advances to a new level. Currently there are only three levels, advanced level being the most difficult. After the completion of advanced level we get a thank you message as shown below in fig 4.8.

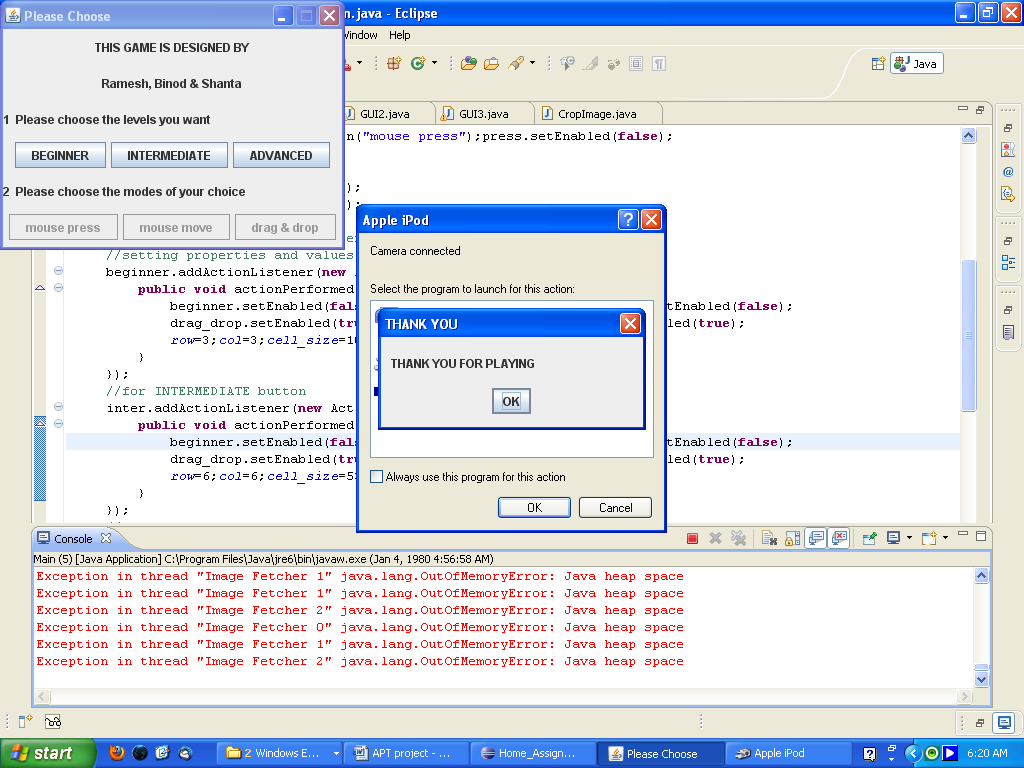


fig 4.8 thank you message

**Chapter 5**

**Conclusion**

The slide puzzle game is an interesting working game where players can enjoy by placing the slides in right place to match the original picture. Players can compete to beat each others records by completing each level with minimum moves than the previous record. Players can even choose to switch between levels otherwise the game advances with new high levels by default.

There are many areas for the improvement in the current version such as to include more advanced levels, includes timing limits, functions and classes to store the records permanently, more supporting functions for other image formats, and many more.

**References**

1. http://www.javaonthebrain.com/java/puzz15/
2. http://leepoint.net/notes-java/examples/games/slidepuzzle/slidepuzzle.html
3. http://www.boutell.com/baklava/slidePuzzle/source.html