PUZ-BUG

INTRODUCTION

An important aspect in programming is detection/finding the bug, its location and cause of its occurrence. Debugging is an integral part in software development. It refers to finding the errors in the code and fixing them. As a software developer, one must possess this skill. The goal of our project was to develop a game called '**PUZ-BUG**'. As the name suggests, it a combination of puzzle and debugging. Our aim was to make learning fun and at the same time teach the students the skill of debugging.

BRIEF DESCRIPTION

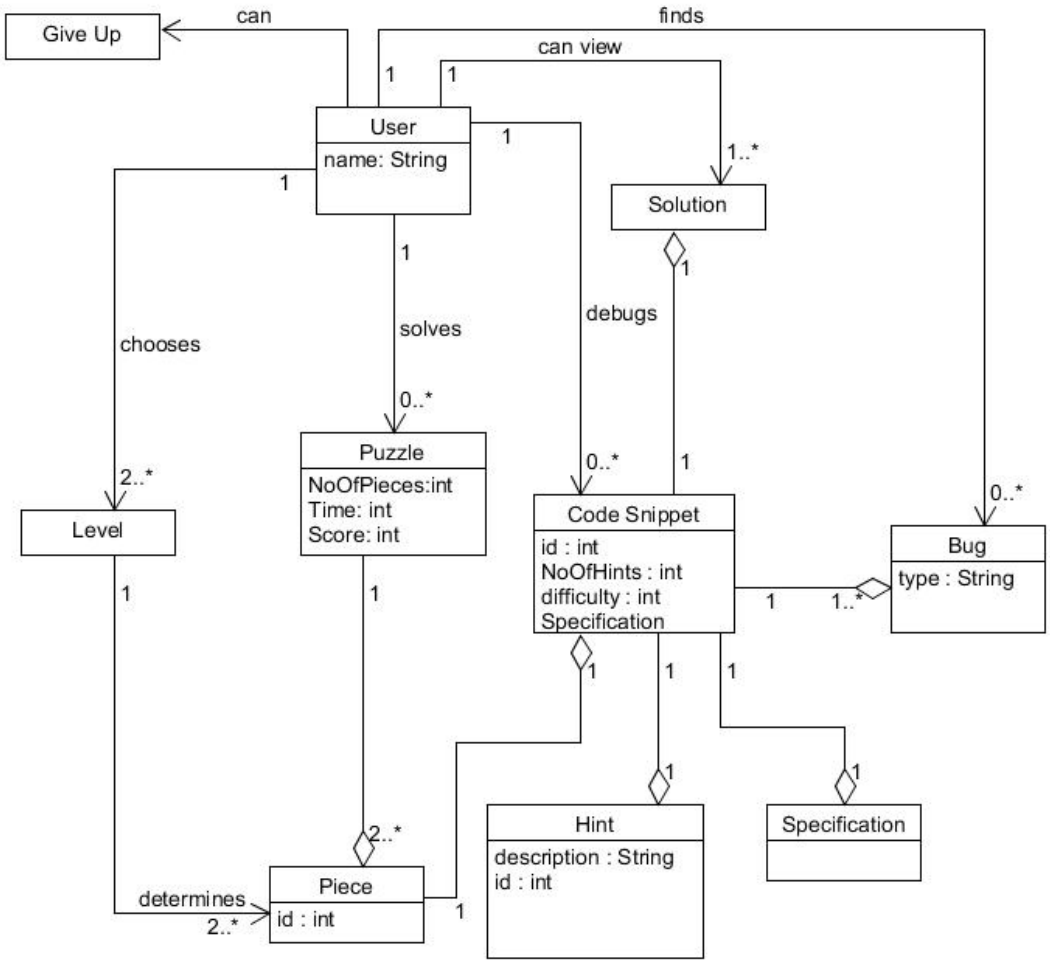
There are three levels for this game - Easy, Medium and Hard. Upon selecting the level, the user can choose the puzzle that he wishes to solve. The puzzle picture will be divided into multiple pieces, each of which will have a code snippet linked to it. The number of pieces will be determined by the level chosen. The puzzle pieces will be 2, 4 and 9 for easy, medium and hard level respectively. To solve the puzzle, the user must first debug the code snippets linked to the puzzle pieces. The code snippets are coded in JAVA.

DOMAIN MODEL

In our domain model, we have the following objects:

- User
- Puzzle
- Piece
- Code Snippet
- Hint
- Bug
- Solution
- Specification
- Give Up

The User object has an attribute called 'name' and the User can solve 0 or more puzzles which is split into 1 or more number of pieces. Each of the pieces have an Id linked to it. Also, each of the pieces has a code snippet linked to it. The code snippets have an Id, number of hints and the level of difficulty as its attributes.
Each snippet has 1 or more bugs in it. The code snippets have 1 or 2 hints linked to each one of them and each of the snippets has a specification that tells the purpose of the code. The User will find 0 or more bugs in the snippet and will debug 0 or more of them. Zero bugs mean that the user was unable to find the bugs and chose to give up.



DOMAIN MODEL

REQUIREMENT SPECIFICATION

At the start of our project we decided upon different functionalities and created user stories for the same. Below are the ones that we had created and their status of completion.

1. As a student I want to learn the types of bugs that might exist in a code so that I can identify them.
Status: Successfully implemented by planting different bugs in code snippets.

2. As a user, I want to be given some hints when I'm stuck with a particular code so that it helps me in debugging the code.
Status: Successfully implemented by giving the user ability to see a hint.

3. As a gamer, I want the game to be interesting and hence, I would like to have the puzzle divided into different number of pieces based upon my choice, so that there is some level of difficulty in the game.
Status: Successfully implemented by giving the user a choice of three levels of 2,4, and 9 pieces.

4. As a student, I want to view the correct code so that if I am not able to debug, I am aware of my mistakes and learn something from those.
Status: Successfully implemented by giving the user an option to give up thus displaying the solution.

5. As a gamer, I want to be able to select the difficulty level so that my interest in the game is retained.
Status: Successfully implemented by giving the user 3 levels, -> easy, medium and hard.

6. As a student, I want to get the ability to run my code so that I can figure out errors in it.
Status: Successfully implemented by giving a run button which will run the code.

7. As a user, I want to know the time taken by me to debug the code so that I can be competitive.
Status: Not implemented as time did not permit but it is possible to add an element to accomplish the same.

8. As a user, I want to be have the ability to switch to a different puzzle piece so that if I am stuck at fixing a particular error, I can solve the other puzzle and return to the unfixed one later on.
Status: Successfully implemented by giving the user the ability to click on any puzzle piece and view the code.

9. As a user, I want to be provided with option of selecting different pictures for the puzzle so that I don't lose interest in the game.
Status: Successfully implemented by giving the user an option to browse a predefined set of images that can be used for the puzzle.

10. As a user, I want to be able to have control over the puzzle pieces in the main puzzle frame so that I can set the positioning of the piece in the puzzle.
Status: Successfully implemented by assigning an active icon which can be used to move the other pieces around in the puzzle. Only pieces adjacent to the active icon can be moved and swapped with the active icon.

11. As a user, I would like to know about the specifications of the code snippet so that I can understand what I need to do.
Status: Successfully implemented by implementing a specification button which give the user the information as to what the code is supposed to be doing.

12. As a student, I need to have an option to debug any errors irrespective of the order.
Status: Successfully implemented by by giving the user the ability to choose any picture and any puzzle piece based on choice and interest.

DESIGN

Architectural Design
Rationale of the architecture: In our project, each component is a complete unit but we need to integrate them so that one component provides the services to the other one. Each layer(component) performs a function that lets the higher layer (other component) abstract away the complexities. Therefore, we chose Layered architecture for our project. Our architecture is divided into following layers:

1. GUI (Graphical User Interface): This is the layer that the users will interact with and it is equivalent to the presentation layer. The main element of the GUI will be the puzzle. The user will have the ability to select any puzzle picture from the given pool of pictures. Once the picture is chosen, the puzzle will be divided into pieces. Each piece will have a code snippet attached to it. On selection, the user will be able to debug the code using this interface.

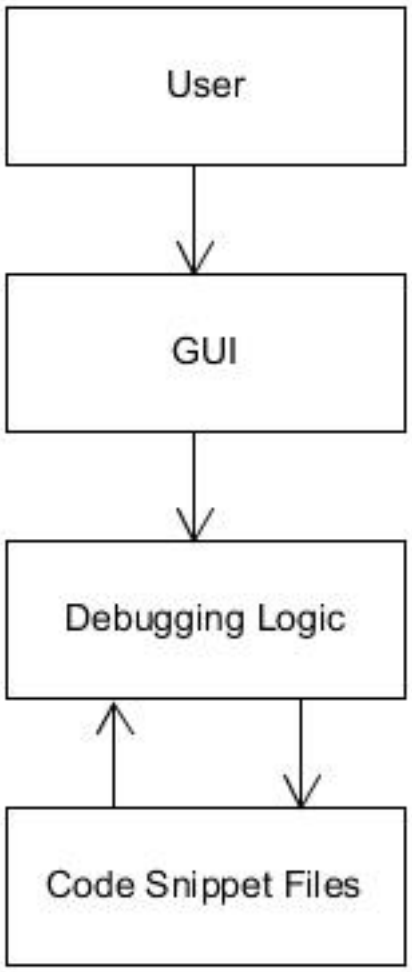
2. Debugging Logic: This layer is the critical component of our system. This layer coordinates with the UI, debugger and the code snippets.

3. Code Snippet Files: This layer will be responsible for handling the code snippets and is equivalent to Data layer. The code snippet that the user must debug will be stored and retrieved from this layer. The solutions will also be stored here so that if the user is unable to debug it, he can view the correct solution. Although, there can be many ways to solve a code snippet, we will provide just one correct solution.

Architectural Drivers
The application that we are building requires a clear separation between the user interface, the debugging functionality and how we plan to store our files. This led us to decide our main architectural drivers.
They are:

- Clean User Interface
- Debugging Platform
- Data Storage for code snippets

Conceptual Architecture Diagram



The important modules of our system are Puzzle Solving and Debugging. For implementing the same, we created the following classes-

1. GuiMain

This is the starting point of our system. This triggers the GUI and all the initial loading is done through this class.

2. Level

This class is used to decide the number of pieces that the picture will be split into. This will depend upon the number of pieces chosen by the user.

3. Split

This is the most important class amongst all the class. It is used for implementation of the movement of the puzzle pieces along with their enabling and disabling functionality. Besides

this, displaying the code snippet visibility and recording the changes made by the user to the same code snippet is managed by this class.

4. FileOperations
   This class has been used for running the code snippets and their testing.

CONCERNS

Testing

The main concern of our system is testing the code snippet that the user edits. Since there are multiple ways to solve one code problem, each code snippet can have more than one possible solution. For our system, we were not able to find a generic way that addresses this issue. We tested the code snippets corresponding to only one particular solution. This limits the usability of the system.

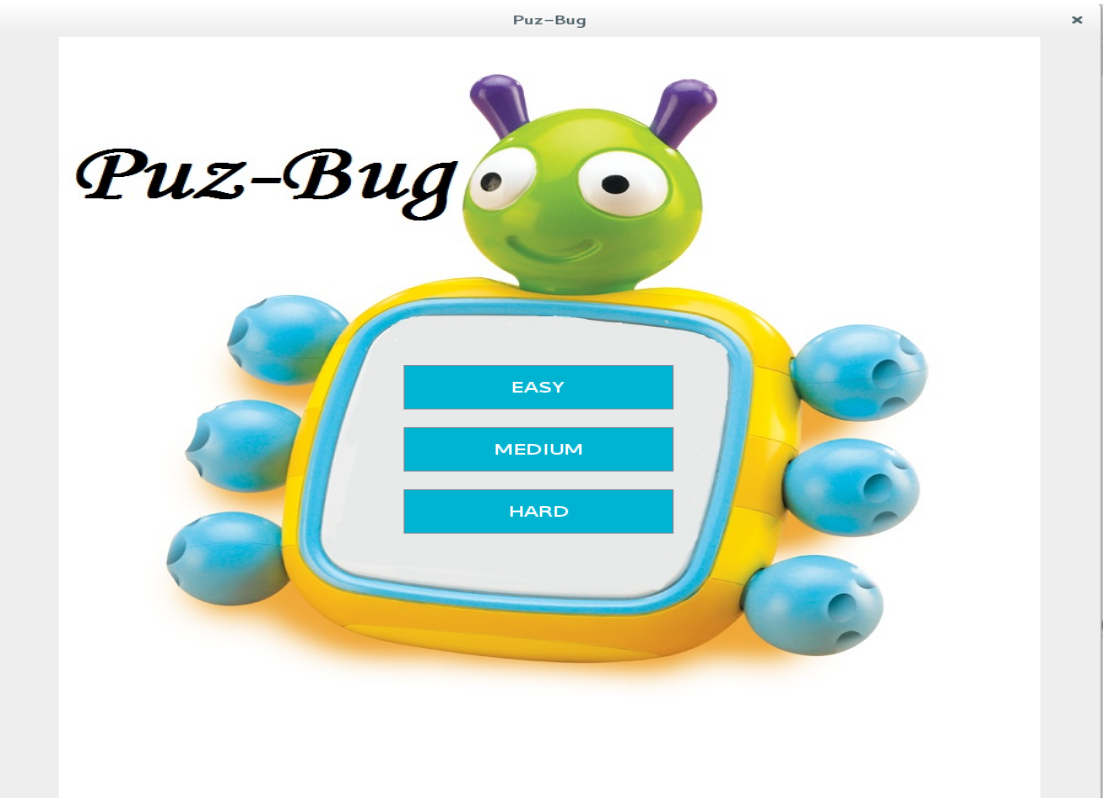
JAVA Required

Another concern of the system is that JAVA must be installed in the user's system.

ACCEPTANCE TESTING

Testing is an integral part of software development. After completing the system implementation, we performed acceptance testing and have captured their results as follows-

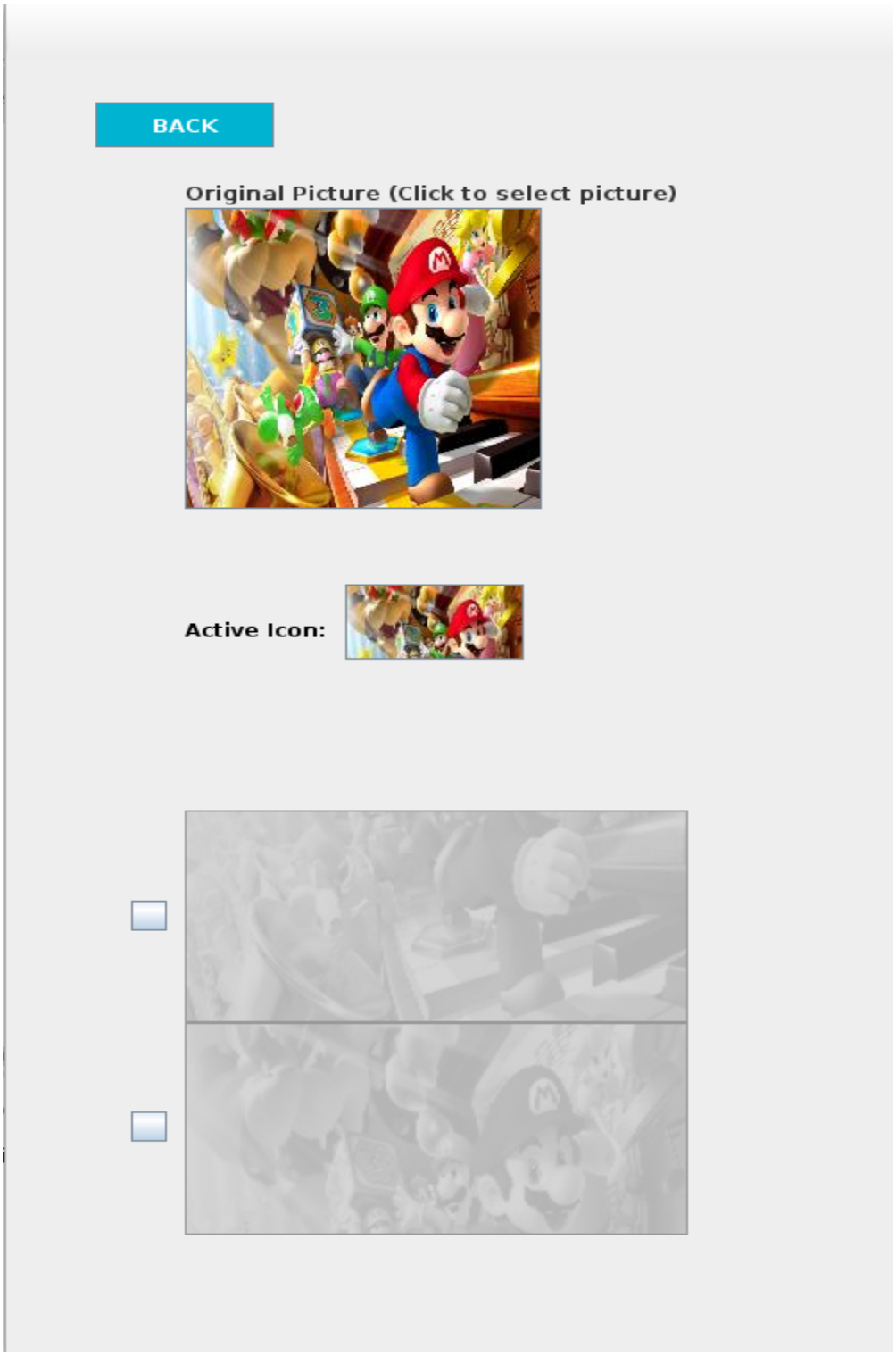
Level Selection



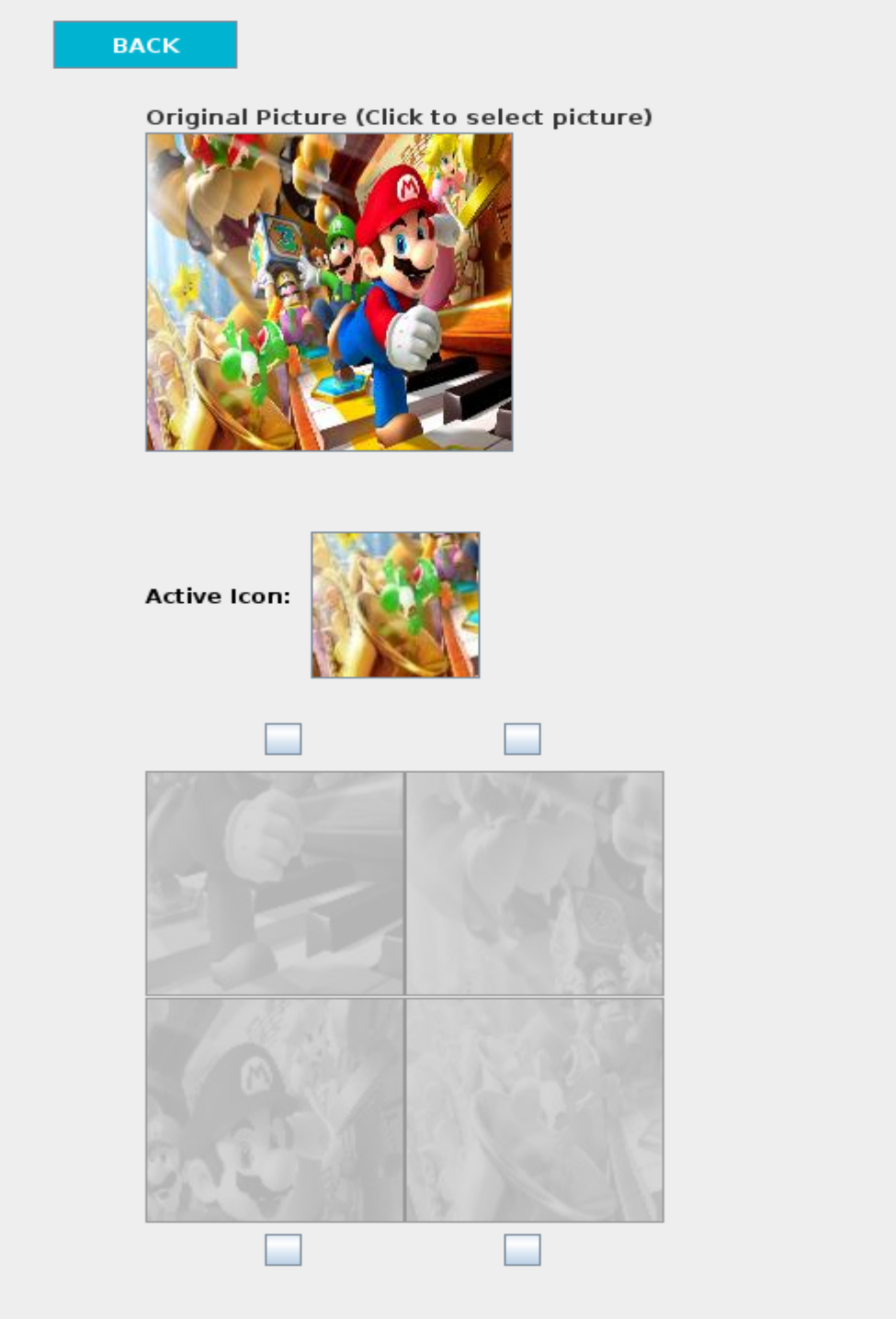
Picture Selection

Puzzle Pieces

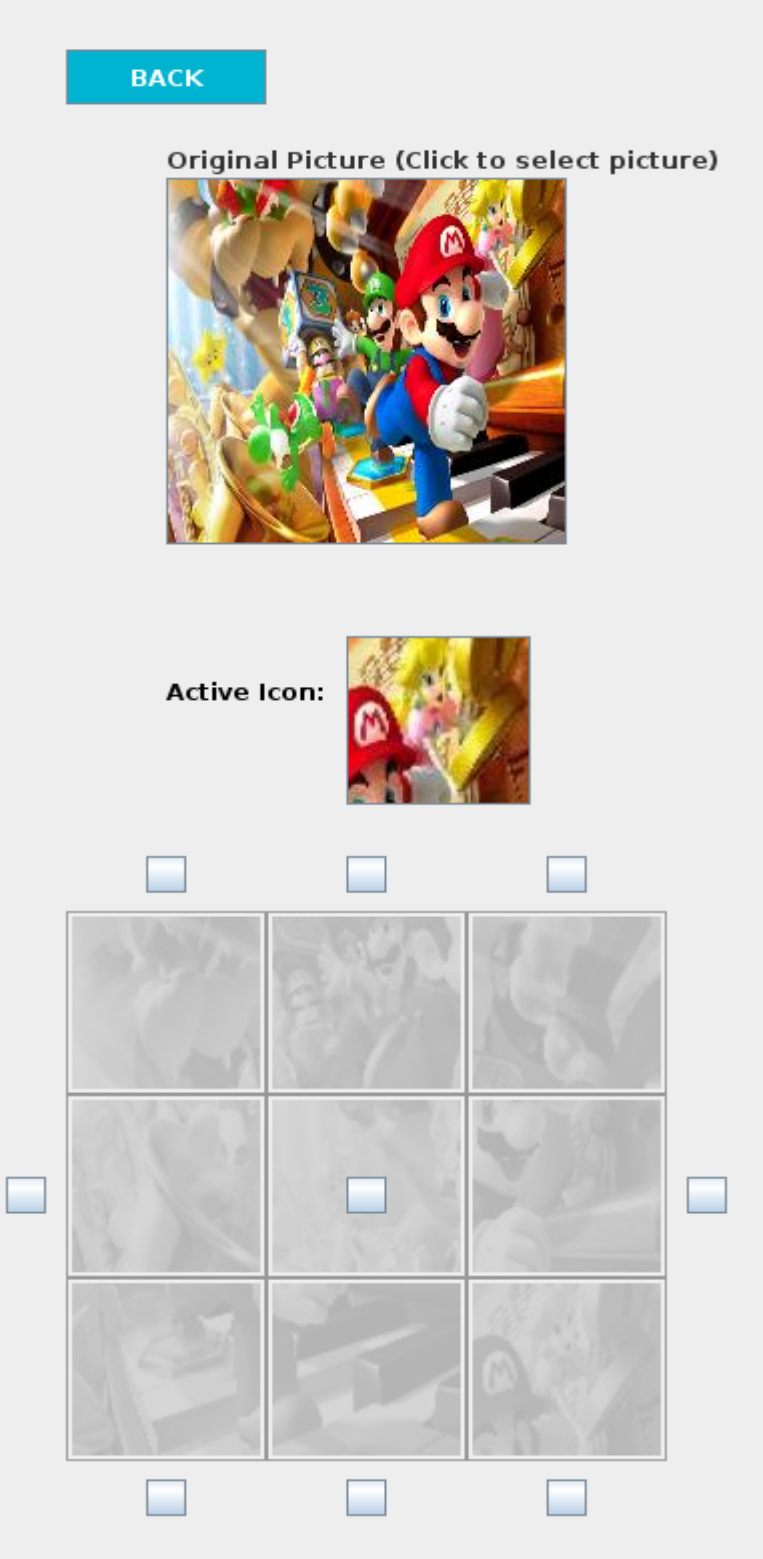
Easy Level (Picture is split into two pieces)



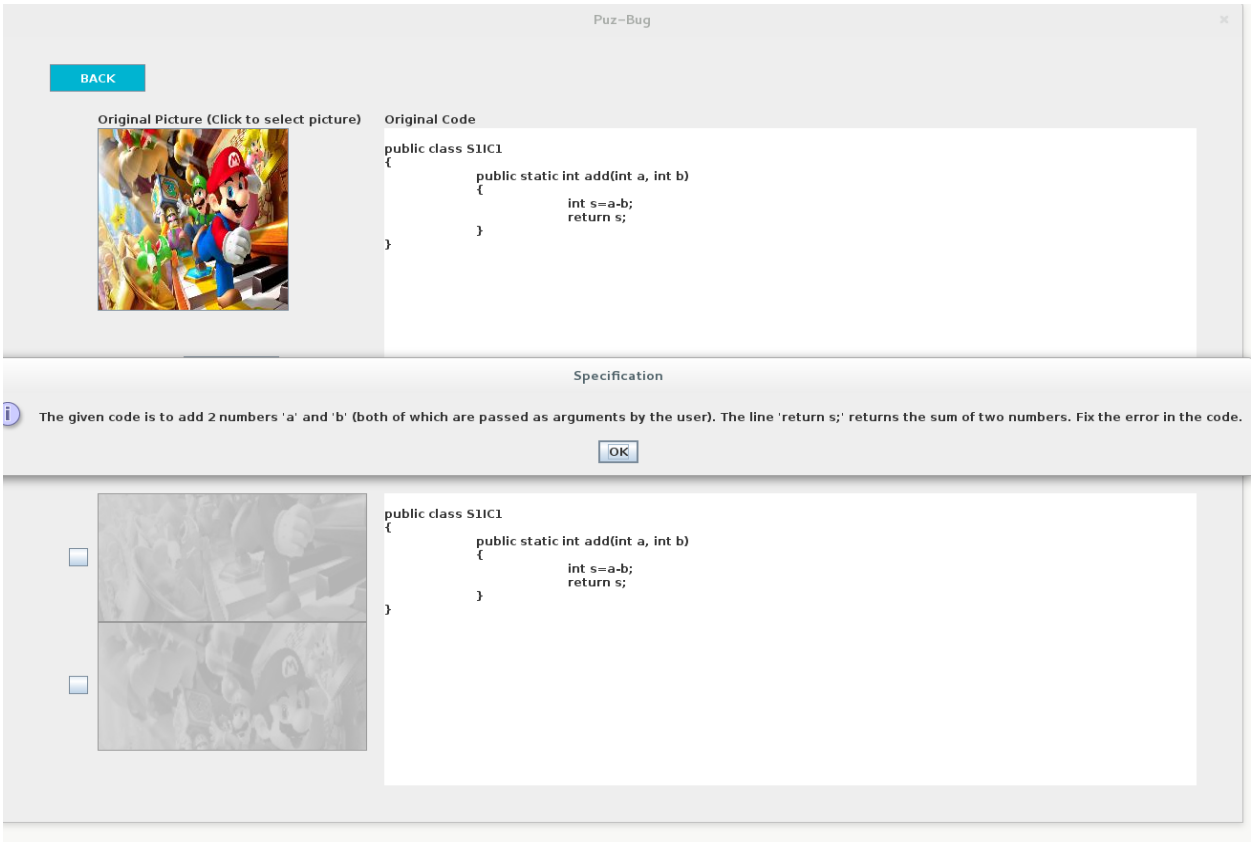
Medium Level (Picture is split into four pieces)



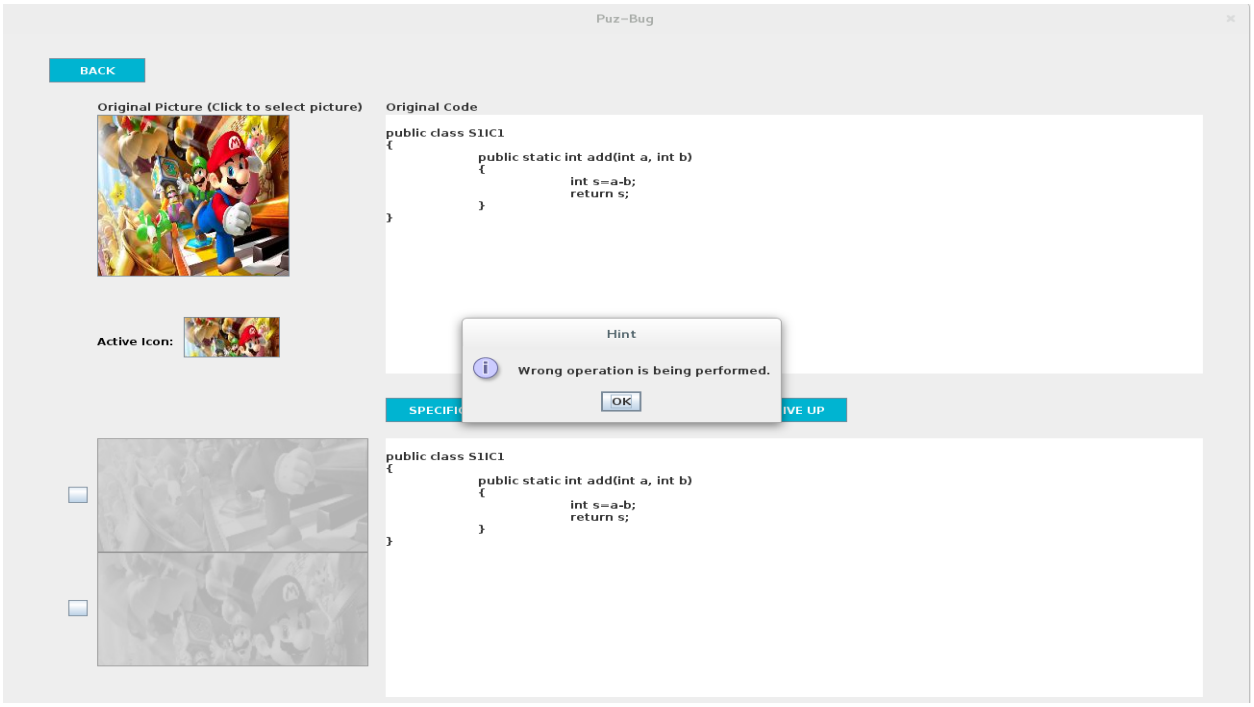
Hard Level (Picture is split into nine pieces)



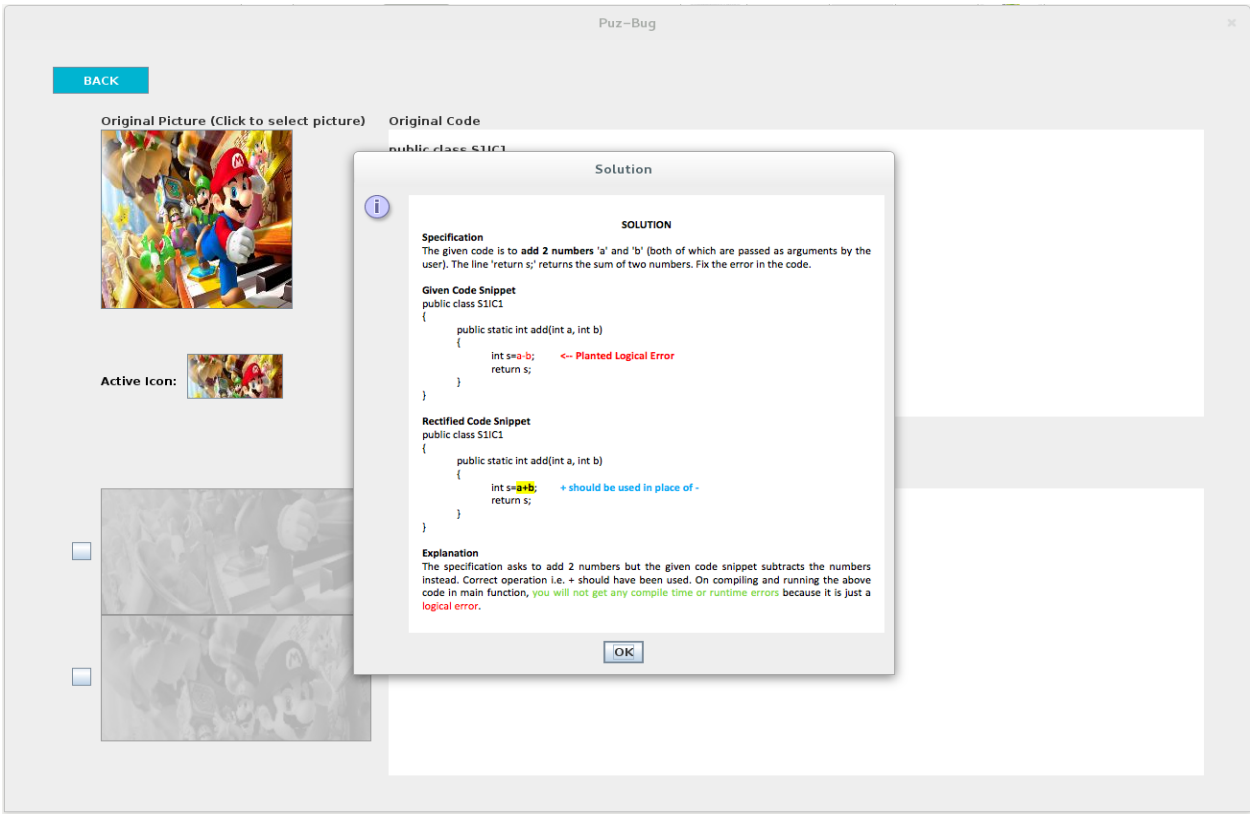
Specification & Original Code Snnipet



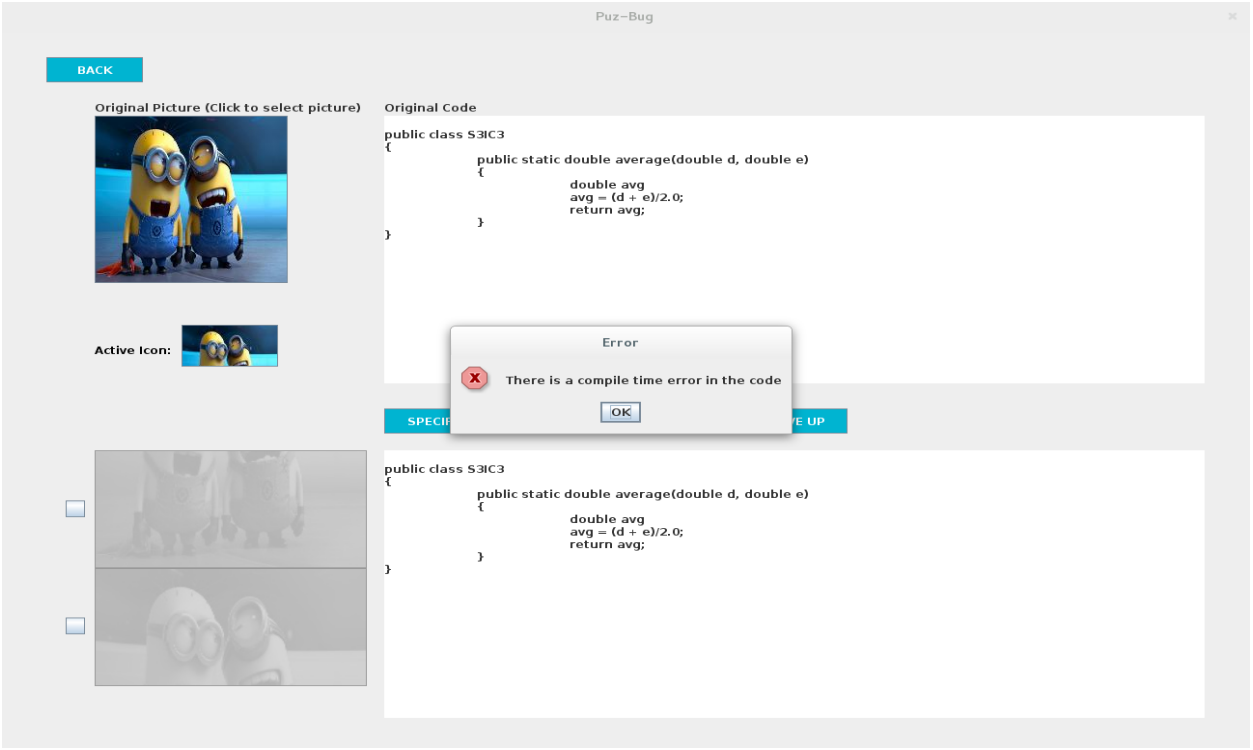
Hints



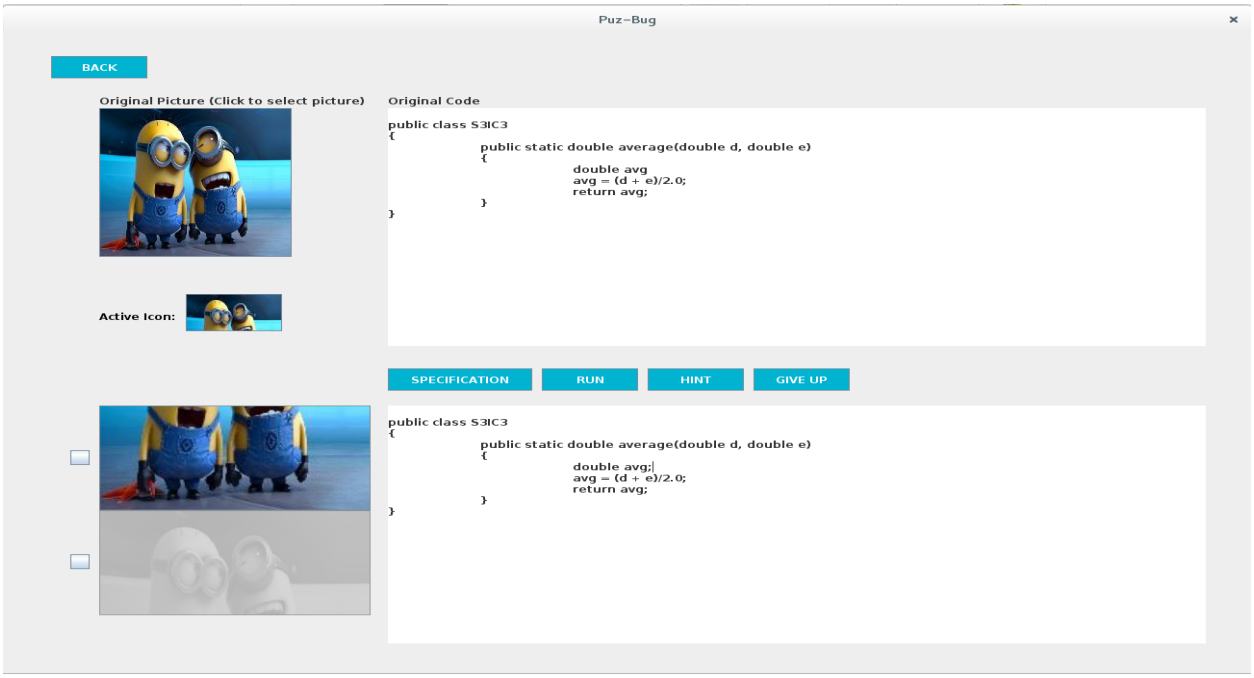
Solution

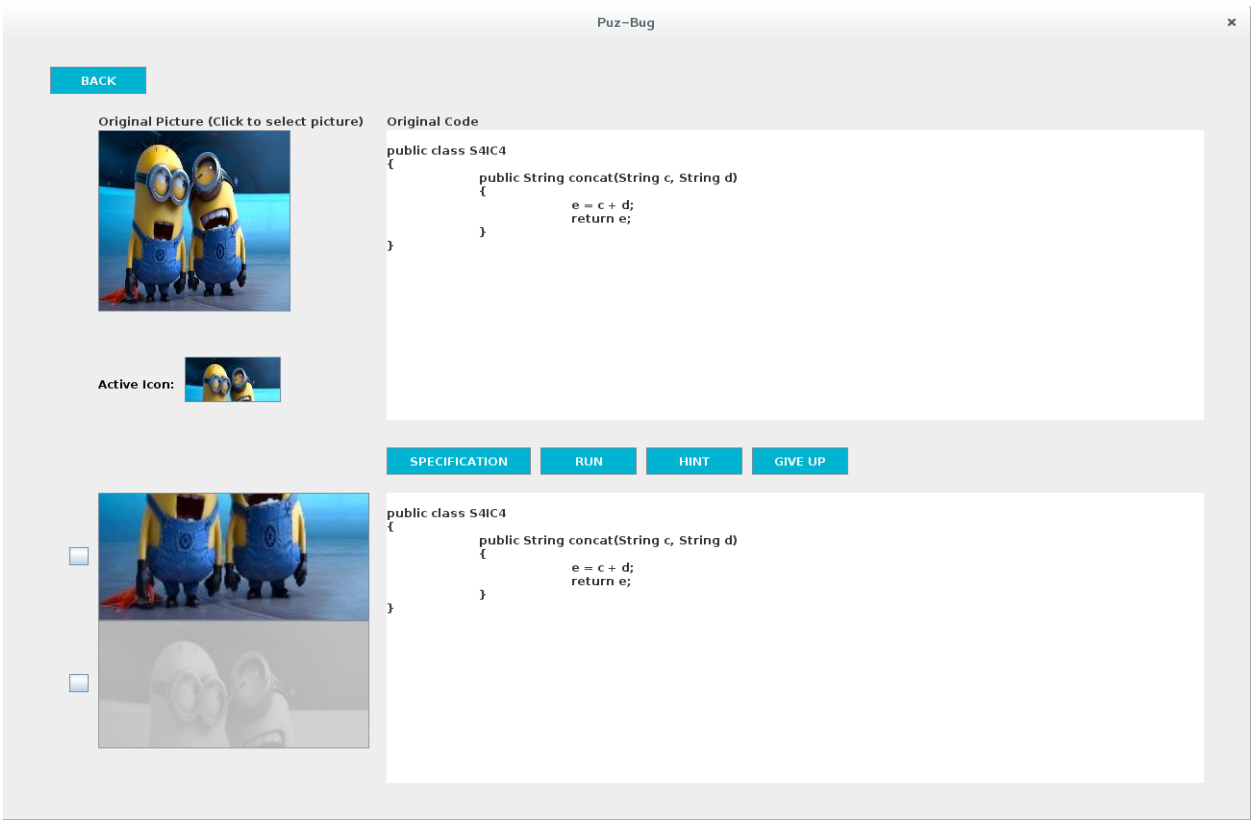


Run the code

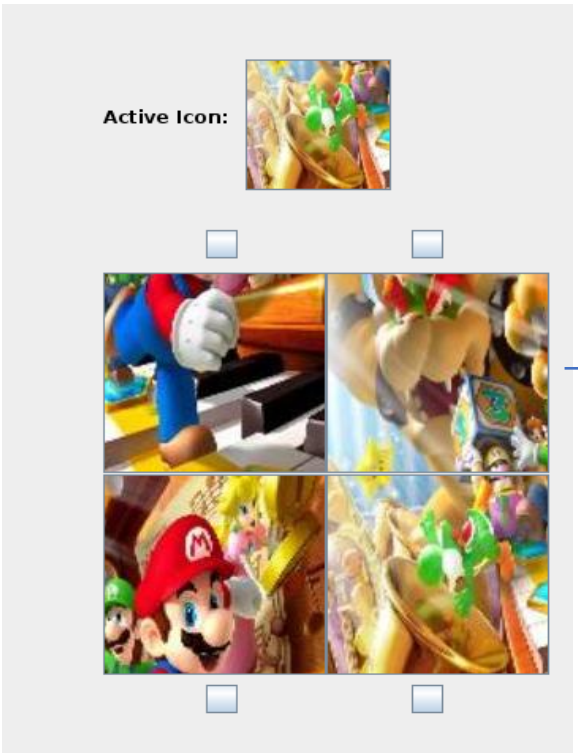


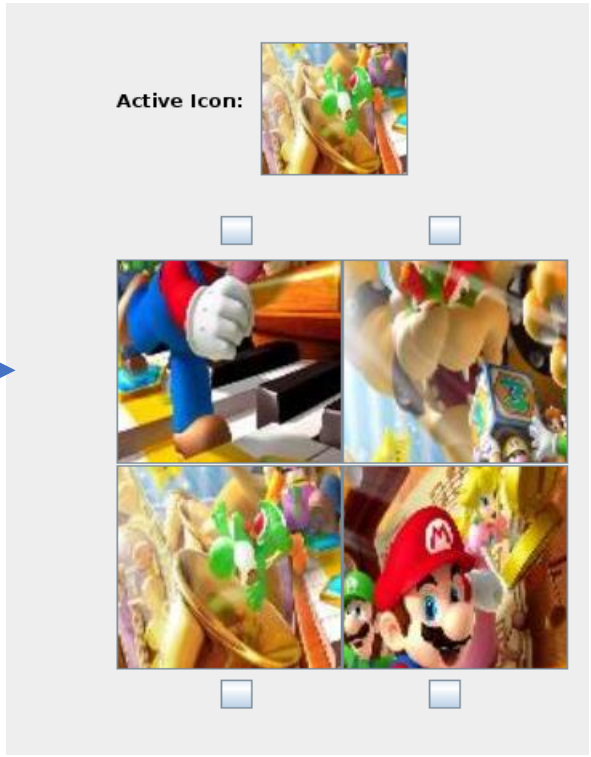
Switch to different puzzle pieces

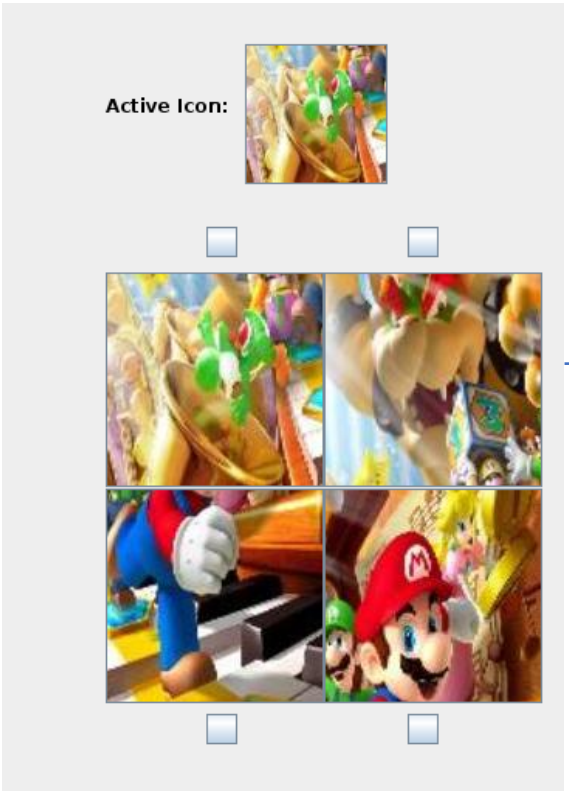


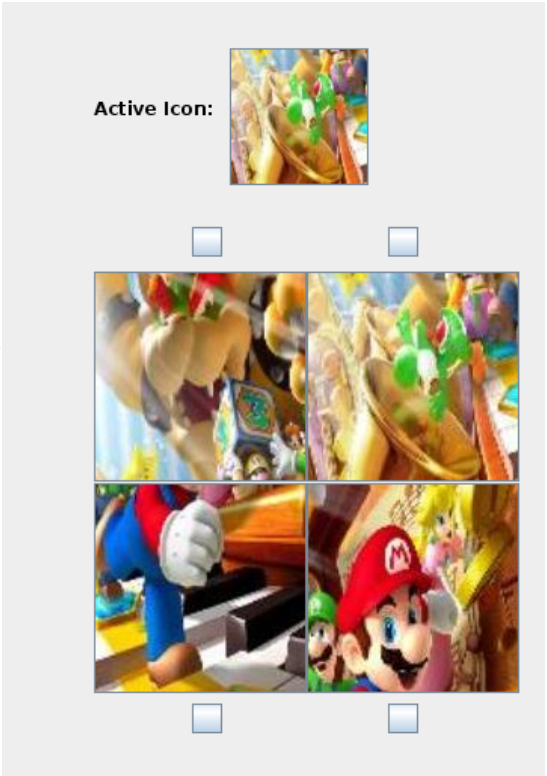


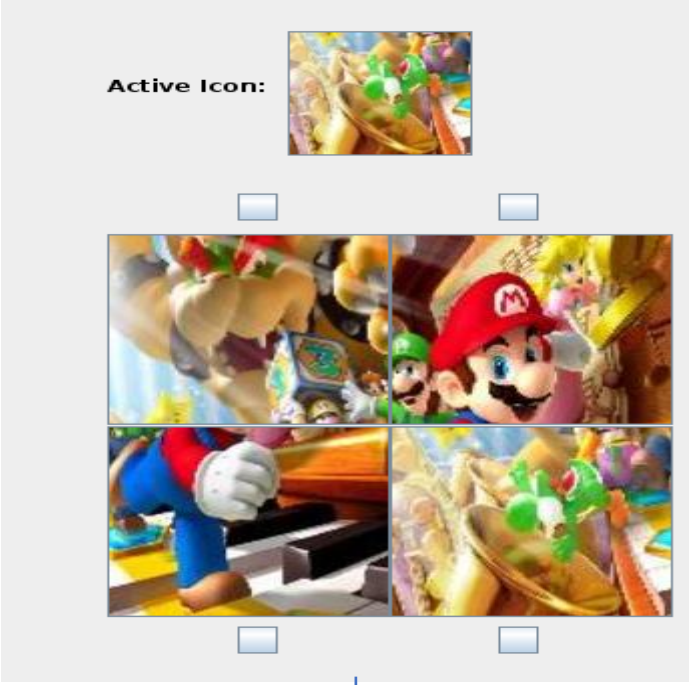
Puzzle Piece movement

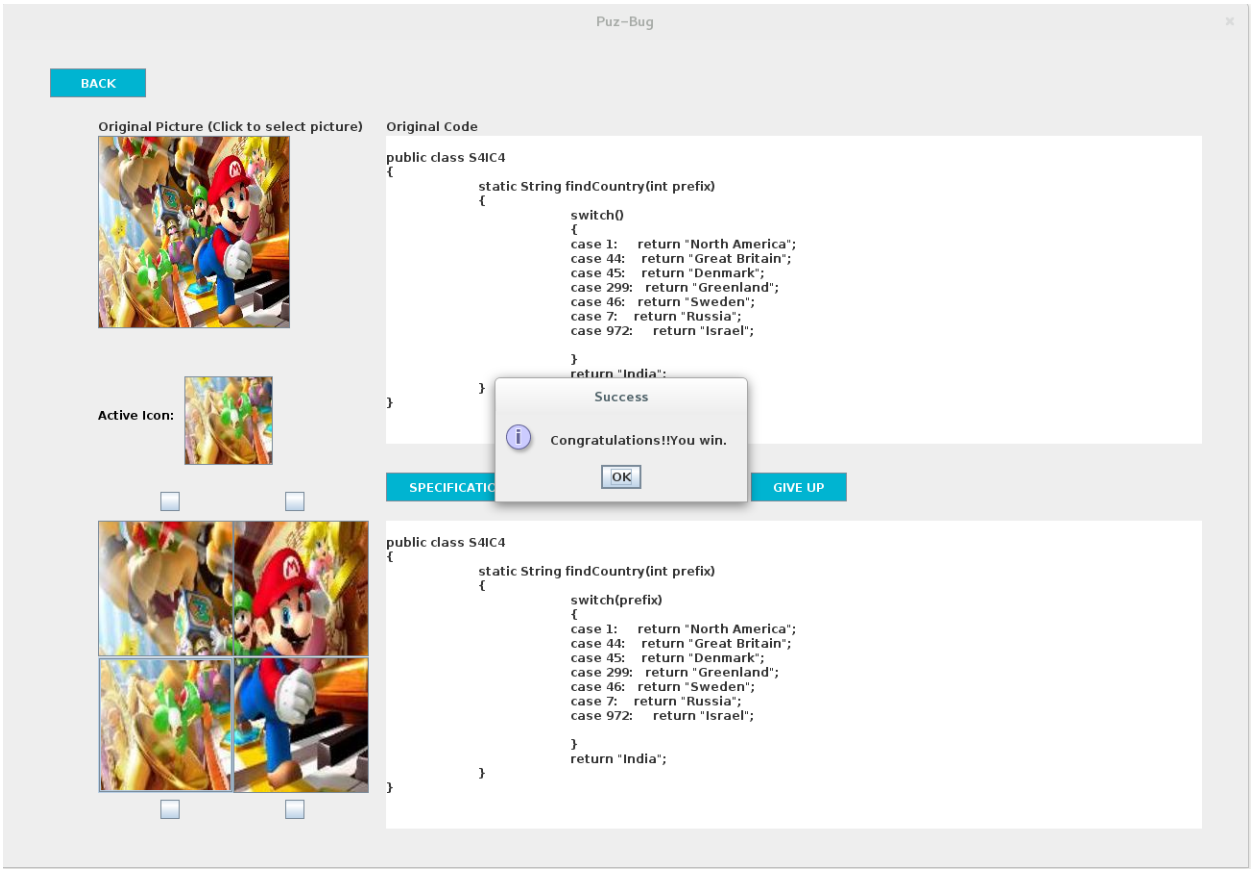









Active Icon:


Puz-Bug
BACK
Original Picture (Click to select picture)
Original Code
public class S4IC4
{
static String findCountry(int prefix)
{
switch()
{
case 1: return "North America";
case 44: return "Great Britain";
case 45: return "Denmark";
case 299: return "Greenland";
case 46: return "Sweden";
case 7: return "Russia";
case 972: return "Israel";
}
return "India";
}
}
Active Icon:
Success
Congratulations!!You win.
OK
SPECIFICATIO
GIVE UP
public class S4IC4
{
static String findCountry(int prefix)
{
switch(prefix)
{
case 1: return "North America";
case 44: return "Great Britain";
case 45: return "Denmark";
case 299: return "Greenland";
case 46: return "Sweden";
case 7: return "Russia";
case 972: return "Israel";
}
return "India";
}
}

PROJECT REFLECTION

Working as a team for this project helped us a lot. This project has augmented our knowledge with new things and concepts. For example, we were not aware that a class could be compiled and run from within another class. As our project required the integration of this aspect we came across different JAVA classes which helped us in achieving our requirement of being able to run a class from GUI on click of a button. This was something new and interesting and made us realize as to how much is out there in the JAVA world and the process of learning is rightly on-going. It was fun integrating the puzzle part of the game because this module has made our project interactive. Linking the codes with the puzzle pieces was a challenge for us but we were able to overcome it with team effort. Overall, the project experience was good and knowledgeable.