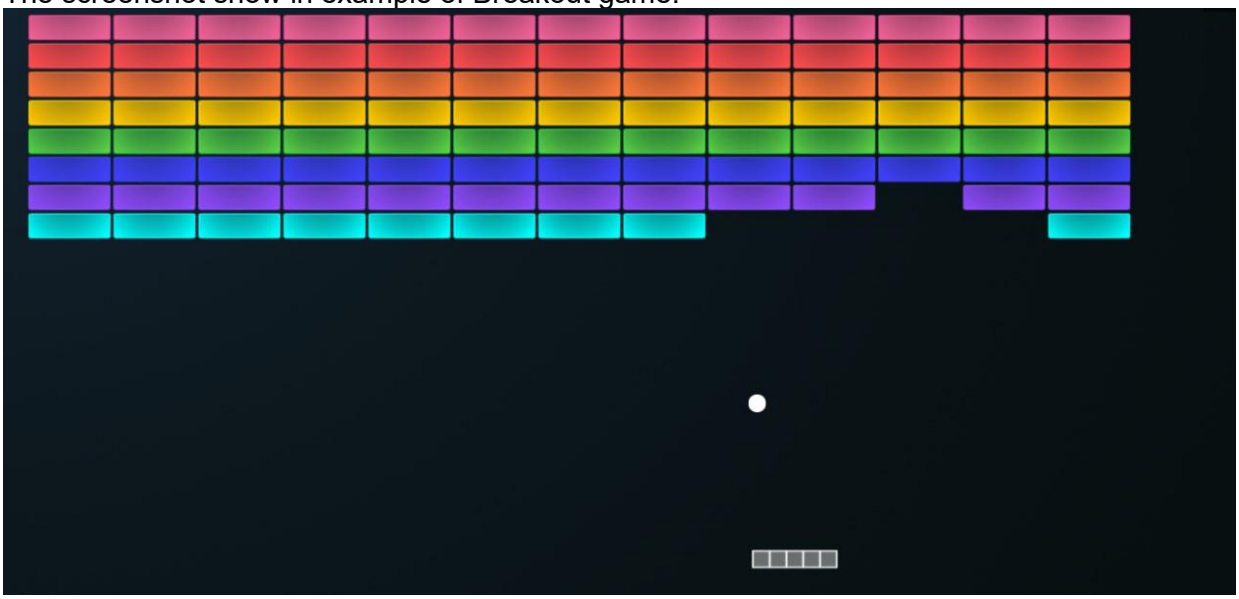
Assignment 2 Draft Version:

Problem statement

Breakout is and arcade game developed in 1970. In Breakout a layer of bricks lines are at the top of the form and the goal is to destroy them by bouncing ball off a paddle into the bricks. The user can move the paddle back and forth at the bottom of the screen with arrow keys or mouse. The ball bounce off the paddle and off the sides of the form. if the ball miss the paddle at the bottom the game ends.

Your Task is to create Breakout game using C# and .NET Framework Windows Forms.

The screenshot show in example of Breakout game.



https://en.wikipedia.org/wiki/Breakout_(video_game)

Requirements

You are provided with a C# starter project that you must use for this assignment. The starter project contains all necessary classes and file for you to complete the assignment. However, you can add more Forms or classes.

1. **Object-Oriented Design**: The game should be designed and implemented using object-oriented programming principles.
2. **Initial Screen:** Implement an introductory splash screen that allows the player to start the game.
3. **Game Interface:** The interface should display a paddle, a ball, and a layer of bricks at the top of the screen. Bricks must respond to ball collisions by either weakening (decreasing in durability) or disappearing when their durability reaches zero.
4. **Ball Dynamics:** The ball should rebound off the game window's edges. It must also bounce off the paddle. If the ball misses the paddle and hits the bottom edge, the game ends.

5. **Paddle Control:** The player should control the paddle using either the mouse or arrow keys.
6. **Animation and Timing:** Use a single timer to manage game animations and movements.
7. **Pause Feature:** Include an option for players to pause and resume the game.
8. **Scoring System:** Display the score, updating it by 10 points each time a brick is hit.
9. **Power-Ups:** Integrate at least two types of power-ups (e.g., paddle size increase, multi-ball feature) that activate when certain bricks are hit.
10. **End-of-Game Feedback:** Provide clear feedback and options to replay the game upon either losing or winning (all bricks destroyed).
11. **Additional Features:** Add at least two unique features to enhance gameplay and engagement. Suggestions include a leveling system, different brick types (e.g., unbreakable, explosive), or special level challenges.
12. **Visual Design:**
    Ensure the game's visual design is visually appealing and coherent, contributing to an enjoyable user experience.

Classes

Your project must be object oriented. You should add following classes:

- Ball
- Brick
- Paddle
- Manager

Your class will need fields and methods. For example for the ball, brick and Paddle objects needs to know where it is on the screen, its size and colour. The ball and paddle also need to know how to move. The ball should also know if it is collided with bricks or paddle. The Manger will manage all the objects, manage animation and calculate game.

Extra Features Examples

- Add different levels to game
- Add appropriate sounds to game
- Keep track of user score especially record the highest score or top three scores
- Allows the players to start, pause, resume, and restart games.
- Provides a way to Save the current game to a file.
- Provides a way to Load a saved game from a file and continue playing the loaded game.

Getting input from the Keyboard

When a user presses a key on the keyboard, a KeyDown event is generated. For the Form's KeyDown event, the event method signature is:

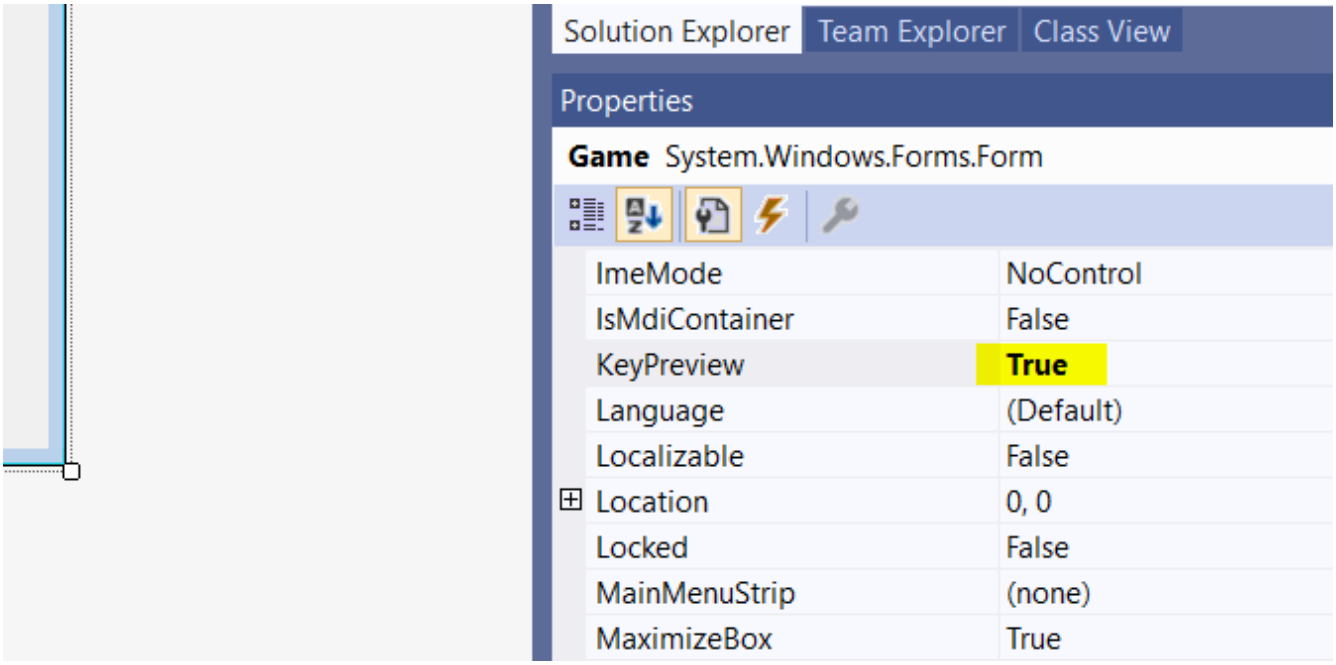
```
private void Game_KeyDown(object sender, KeyEventArgs e)
```

Game is the name of the form. Following is the sample code

```
private void Game_KeyDown(object sender, KeyEventArgs e)
  {
      switch (e.KeyCode)
      {
          case Keys.Left:
              //write code in response to the left arrow key
              break;
          case Keys.Right:
              //write code in response to the right arrow key
              break;


      }
  }
```

The Form's `KeyPreview` property must be set to True. Otherwise, it won't be able to respond to the `KeyDown` event



Required Tests

This is the minimum number of tests you need in your testing documentation:

1. Interface is displayed correctly when the program runs
2. Bricks are displayed correctly
3. Paddle is move with key Press or mouse move
4. The ball is moving
5. The ball is bouncing off the paddle
6. The ball is making the bricks disappear when hit by the ball.
7. Score is calculated correctly
8. The won message displayed when the user won
9. The lost message displayed when the user lost
10. User is getting an option to replay the game when the game is won or lost

Please note that your tests can involve more than one step and **must** be reproducible (i.e. explicit test data and user actions) and independent of each other (i.e. please do not use the output of one test as the input to another test.)

Test Cases

Please using the following format for your test cases

Requirement to test	Test Data Input	Expected Outcomes	Actual Outcomes

Required Unit Tests

You must write at least 7 Unit tests to test methods for your choice. You can select methods from any class, consider following classes.

- Ball
- Paddle
- Brick

Delivery

A soft copy must be uploaded onto **Moodle** as a single **.zip** file prior to the deadline and it must comprise:

- The testing documentation.
- ALL files needed to compile and run your application from the **Visual Studio Community 2022 or 2019**. 20-30 marks will be deducted if this is not done.

Please note that it is important to upload the correct version of your assignment onto Moodle. If you submit the wrong version onto Moodle, please notify me by email before the deadline date or late penalties may be incurred.

Checkpoints

There is an expectation that you will have completed certain aspects of your assignment at each checkpoint submission. If your submission does not meet the expectations of the checkpoint, your overall mark for the final submission may be penalised.

Expectations

Checkpoint :

- User interfaces are designed, and the ball is coded for the game
- Classes are added for Ball, Paddle and Bricks.

Demo

You must give demo of your assignment and answer all questions about your code. 50 marks will be deducted if this is not done. Without demo and Q&A your assignment will not be marked.