1. Requirements

Summary

Breakout is an 1976 arcade game developed by Atari. It is a simple yet fun game. The objective of the game is to destroy the layers of bricks at the top of the screen. To do this, the player controls a paddle at the bottom of the screen by moving their mouse. A ball travels across the screen bouncing against the walls and top of the screen. If the ball hits a brick, the brick is destroyed and the ball bounces back down.
The player must bounce the ball off of their paddle in order to bounce the ball back up and destroy more blocks. If the player fails to hit the ball with their paddle, the player loses a life. Each time the ball hits the player’s paddle, the speed is slightly increased. Once the ball is reset when the player misses the ball, the speed is also reset. The game is complete once the player has destroyed all of the blocks at the top of the screen. The player may be awarded by collecting power ups that appear randomly and move across the screen. The power ups have several different types of effects that may be possible. Additionally, there may be “anti power-ups” that effect the player negatively. These will be clearly marked for the player to distinguish from power ups. The player’s score is based on several different criteria including time and combos.

**Input**

In the main game, the input depends greatly on the player’s mouse position to control the position of the paddle. The position of the paddle upon collision with the ball will determine the bounce direction of the ball. For example, if the ball hits the edge of the player’s paddle then it will bounce at a steeper or wider angle depending on which edge of the paddle. If the ball hits the direct center of the paddle then the ball’s direction will be reversed but the angle will be the same.
Example of the ball’s change in direction when it hits the edge of the player’s paddle

Example of the ball’s change in direction when it hits the exact center of the player’s paddle. The angle remains the same.
Gameplay Flow

The gameplay flow is as follows:

Glossary

- **Game Over** - A game over occurs when the player has run out of time or lives.
- **Point** - A point is scored when the player breaks a “block”
- **Combo** - A combo can be achieved by breaking multiple blocks in only one bounce. Therefore, it is beneficial to aim the ball for an area where it will destroy the most blocks. When the ball collides with the paddle, the combo is reset.
- **Score** - The player’s score is based on a combination of both the blocks broken, combo points, and time remaining.
- **Paddle** - The paddle is the area controlled by the player’s mouse upon which the ball can bounce off of.
• Ball - The ball is a constantly moving object the bounces off of the screen’s walls, ceiling, blocks and player’s paddle.

• Block - Blocks are objects aligned in rows at the top of the screen which must be broken for points and to complete the game.

• Bounce - A bounce occurs when the ball hit’s the players paddle or the screen’s walls. If the ball hits the player’s paddle then the bounce direction depends upon the position of the paddle hit.

2. Design

Gameplay Bonuses and Penalties

The game itself is very simple. In order to add complexity different types of bonuses and penalties can be applied to the player through collected items. Many of these collected items are clearly marked so that the player can actively attempt to collect or avoid them. The effects are temporary lasting only several seconds. Some of the possible bonuses and penalties include:

Bonuses

○ Multiple balls - An additional ball is added to the screen. This allows the player to potentially finish the level twice as fast. If the player fails to keep one of the balls within the bounds then no penalty or life is lost as long as at least one ball remains.

○ Paddle Size Increase - The size of the paddle is increased, increasing the area for the ball to bounce off of.

○ Energy Ball - The ball will not bounce immediately off of the blocks and instead will break through them and bounce off of the bounds of the screen. This will get the player many points and also add complexity to the level to make it more interesting.

○ Multiplier - Points gained are multiplied.

○ Extra Life - The player gains an extra life.

Penalties
- Paddle Size Decrease - The size of the paddle is decreased, decreasing the area for the ball to bounce off of.
- Ball Speed Increase - The speed of the ball is increased making it more difficult for the player to keep up.
- Control Reversal - Player controls are reversed making it more difficult for the player to control the paddle.
- Point Penalties - The player loses points when break blocks after collecting this.

**Points**

The amount of points a player earns is determined in several different ways.

**Time Remaining**
- The amount of time remaining is added to the score when the level is completed.

**Block Color Variation**
- Block color is based on their core point value.
  - **Red** - 5 Points
  - **Yellow** - 10 Points
  - **Green** - 15 Points
  - **Blue** - 20 Points

**Combo Points**
- The block point value is multiplied each time the ball bounces off of a block and is reset when it hits the player’s paddle.

**Levels**

There is no designed limit to the number of levels a player can play before reaching a game over. However, the ball’s speed increases each time it comes into contact with
the player’s paddle. The speed increase is based on the level the player is currently on. For example, the speed of the ball will increase much faster at level 5 than level 1. The rate at which power-ups and power-downs appear depend on the level. The higher the level, the more hazardous it is.

Level Design

- There are many different possible ways levels can be arranged. The possible design types of levels will be defined in a text file and loaded when the game is run. An example of a few different level designs are shown below:
3. Coding

The Waterfall Model
In order to write code in an efficient manner, the “Waterfall Model” is used as a guideline. The game’s structure was designed before beginning programming to avoid designing the structure on the spot. Additionally, the design serves as a sort of “blueprint” of the game, the specification of the code already exists and all that remains is to write the code.

Coding Standards & Guidelines

Format
- Coding standards give many benefits including consistency and increased readability. The code format used in this project can be best described as K&R Style. The goal is to maintain consistent spacing, indentation, and braces.

Naming
- Variable names should be meaningful and able to communicate the use of the variable contents. Variable names start with a lowercase letter and each successive word starts with an uppercase letter.
- Constant variables are all uppercase and multiple words are separated with an underscore.
- Functions and methods use a verb and noun to describe their behavior. Functions have the same naming style as variables.
- File names briefly describe the contents.
Documentation

The code is meant to be as self documenting as possible. This relies on the above standards and guidelines for consistency. Comments are used to describe some pieces of code. Additionally, functions are given a comment header that gives a full description of it’s behavior.