Team 18

Rachel Davis

Rumsha Khan

Jiayi Tao

**Executive Report**

**Introduction:**

The original Breakout was created in 1976. It consisted of 3 “levels” and points are given for each brick destroyed. The ball has the ability to bounce off the top and sides of the window, while a life is lost if the ball reaches the bottom. In our version of Breakout, we implemented multiple changes. Some of these changes include: a different scoring function, various game “themes”, and sound files.

**Team:**

Work Distribution and Contribution:

Milestone 1:

For the first milestone, the bricklayer at the top of the screen was created by Rachel. Over break, Lucy was efficient enough to complete the following tasks. She created the player paddle and gave it the ability to be controlled via mouse interaction. The change in angle of reflection of the ball depending on where it hits the paddle was done by Lucy, after collaborating with the rest of the team on how to implement this functionality. Rachel worked on the level complete screen that occurs when the ball hits the top of the window. Rumsha and Lucy worked on the removal of a brick when hit by the ball.

Milestone 2:

All requirements for Milestone 1 were completed before the deadline and before work on Milestone 2 began. All group members collectively worked on adding level progression. However, Lucy implemented the majority of the code for level progression and changes while Rachel mainly worked on the pause and resume functionality as well as the startup screen. Rumsha worked on adding the score, number of lives, and timer to the display. Lucy helped implement a score function that adheres to the game objective given in the rubric. Rachel added the high score function in the arcade-style . Additionally, she worked on maintaining the high-score information in a flat file that is loaded at game startup. Rumsha implemented the ball reset function that occurs when the ball touches the bottom of the screen. She also worked on the loss of lives when the paddle missed the ball, as well as resuming the game after a pause.. Rachel also added a in-game screen for viewing game-play instructions and highscores. If the ball touches the bottom and the player is out of lives, the game ends with a “Game Over” message and allows the user to input his or her initials if a high-score has been attained. Rumsha and Rachel worked on this together. During the gameover screen, the player is given the option to restart, to view highscores, and to quit. These features were implemented by Rachel.

Milestone 3:

Milestone required 3 new features to be added to the gameplay. Lucy created 3 different brick themes, consisting of food, pets, and Christmas images. She and Rumsha also worked on the sound effects played during the the game, and Rachel completed all the important final changes for the game, including the changes from Milestone 2’s high score feature that needed to be made, the endgame options, and small bug fixes.

Strengths & Weaknesses:

Group Strengths & Weaknesses:

Our strengths included our teamwork and cooperation. We all got along and worked well together. We were each present at every team meeting and worked on the code together many times. Each team member was really good about helping the others when they were having problems with their code. Sometimes it takes an extra set of eyes to see an issue that isn’t apparent at first.

A definite weaknesses of our group was lack of organization. Although our efforts to divide individual tasks evenly were successful initially, there were often times when some group members completed more than what their assigned parts entitled, overlapping with another group member’s assigned task. However, as a team we communicated well and in order to keep the workload fair, we assigned bits of other tasks to any group member whose assigned code was done by another member.

All in all, our group worked well together.

Individual Strengths & Weaknesses:

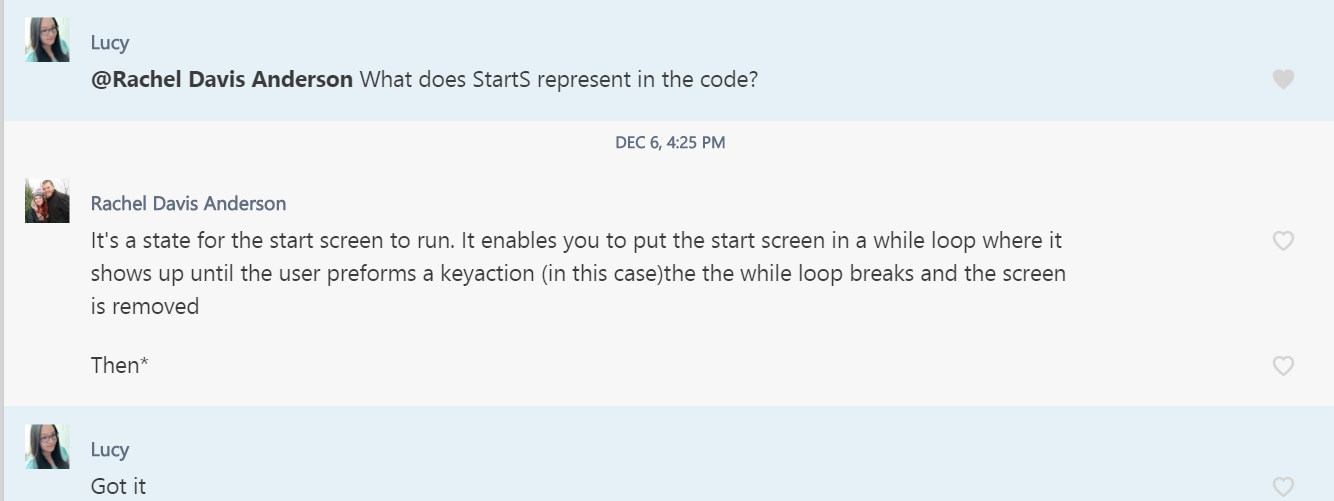
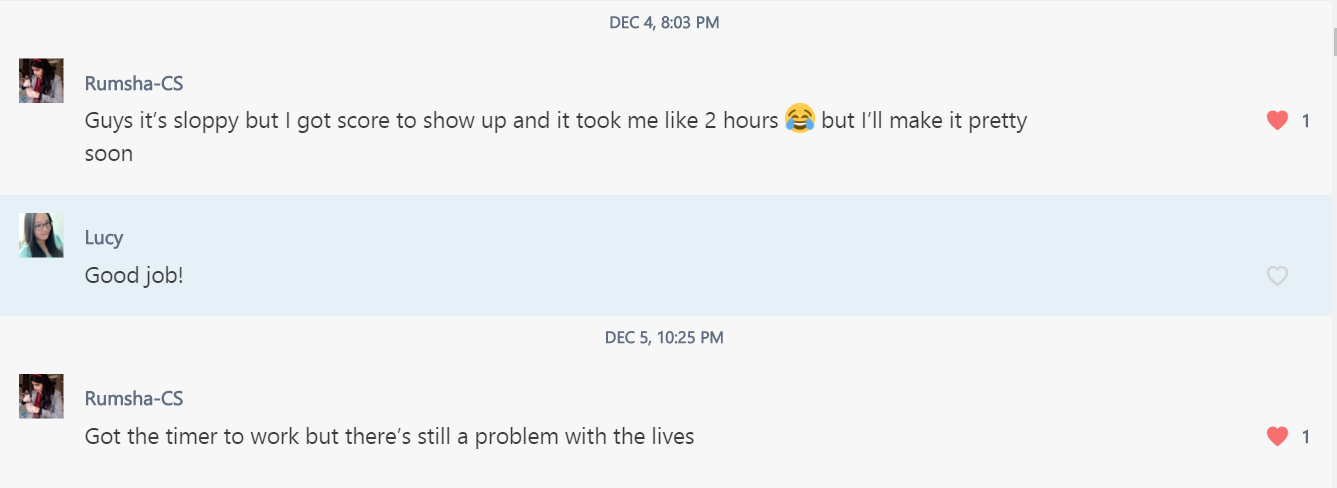
Rachel: Rachel was good at communicating. She explained her part of the code clearly and oftentimes interpreted code written by other group members to the rest. Her weaknesses were time management and persistence. There were many instances where she got stuck on a problem, and instead of moving to something else and coming back to the problem at hand, she would work on one problem for too long. Eventually, her team members helped her to realize that it’s okay to ask for help when she was stuck.

Rumsha: Rumsha was good at teamwork and communicating. She had a hard time with a lot of the code writing and so it took her longer to complete very simple tasks. Although she needed extra help, she managed to complete her tasks.

Lucy: Lucy is good at time management. She works efficiently and always be able to finish her part on time or even ahead of time. Her weakness is that when she encountered something that she doesn’t know how to do, she would get anxious and feel stressful. She always needed some time to calm down to move her attention purely on the project instead of her emotion.

Communication Techniques:

Aside from our meetings, we regularly communicated through GroupMe and also shared Google documents on Google Drive and added notes to tasks we needed to complete.

****

Features Implemented:

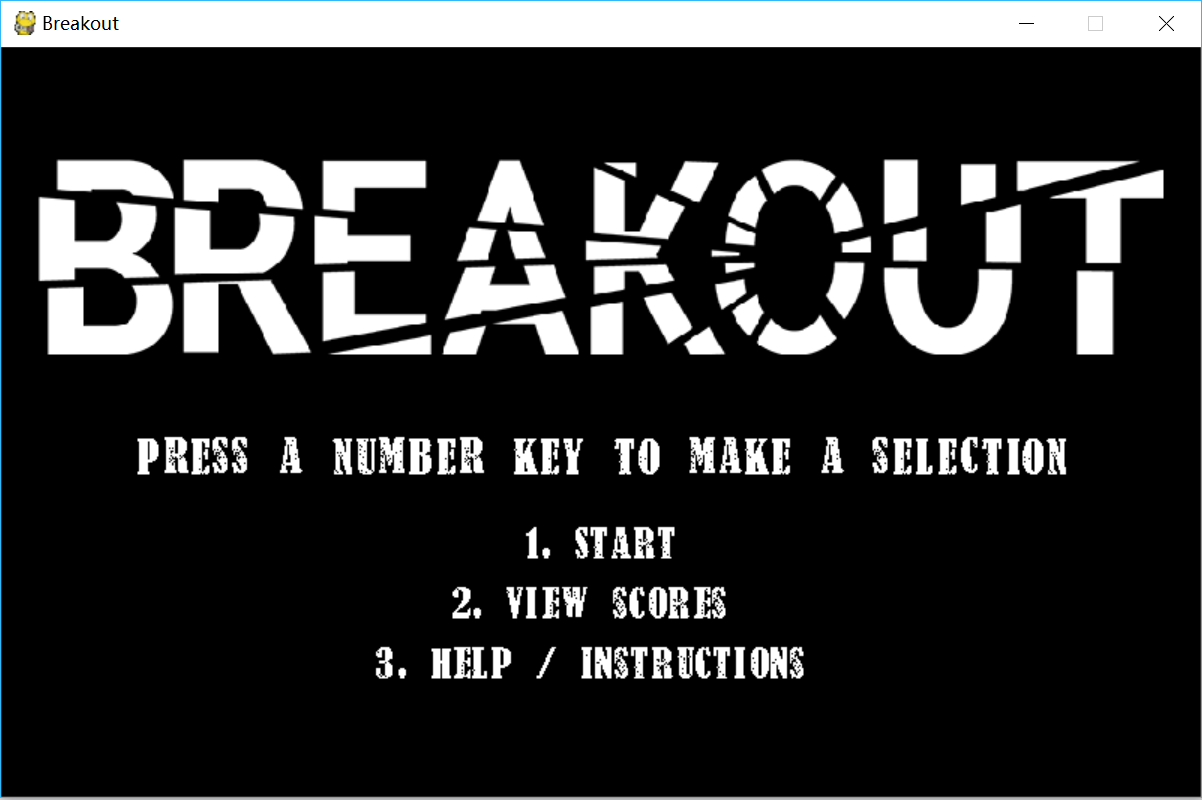
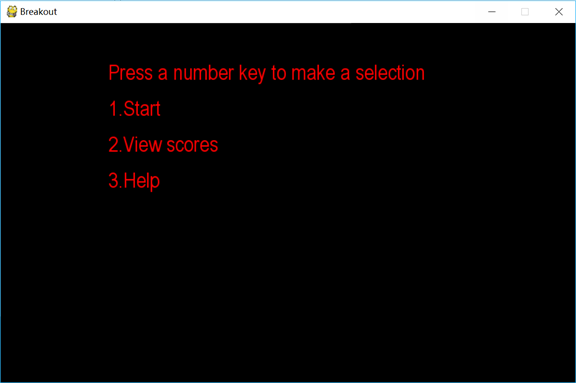
Our game includes all the basic features that were required as well as adding our own spin to it. Additionally, we added a few additional features that we decided on as a group. Each feature was worked on generally as a group, with some features being worked on more heavily by individuals.

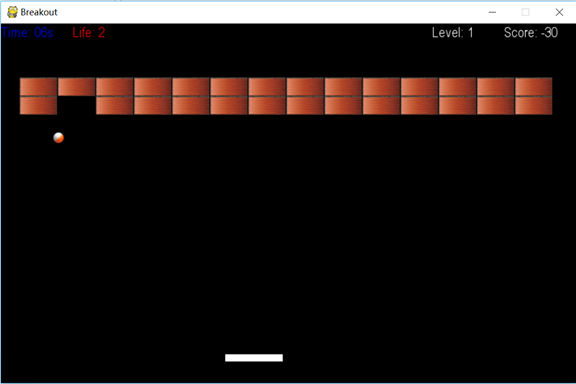
* Start up Screen
  + Starting options and game title
    - Start
    - Highscores
    - Instructions
* Brick Layers
  + Layers increase per level
  + Level arrangement changes per level
  + Different themes for the bricks(change every time you start the game):
    - Food
    - Cats and Dogs
    - Christmas
* Moving Paddle:
  + Moved by mouse
* Ball:
  + Bounces in different angles of direction based on which part of the paddle is hit
* Score, Lives, and Time are displayed on top of screen:
  + Change according to how game is played
    - Score: goes down by 10 every time the ball hits a brick
    - Lives: start with 3, decrease every time ball hits the bottom of screen and not paddle
    - Time: starts at 0 seconds and increases for the duration of the level
      * More points added for lower time
* Once all lives are lost, either a “Game Over” pops up or an option to enter your high score pops up
  + Once a high score is achieved, a box pops up that asks for the players initials and adds that score to the highscore.txt
* Sound effects

Additional Features:

* Different themes(all matching images)
  + Ball image
  + Paddle image
  + Brick images
* Different sound effects
  + Background music during the duration of the game
  + Sound for when the ball falls to the bottom and player loses a life
* Different brick arrangements for increasing difficulty
* We slightly improved and added our touch to the features required

Software Development Process:





*Special Software Features:*

Features include: game speed increases as you hit more bricks and go up levels, Christmas, food, and pet themes, brick arrangement changes, and sound upon certain events as well as background music.

*Scoring Function*

The scoring system is divided into two parts. First, each time the ball hits a brick, 10 points are deducted from the score. The initial score of the game is 0, so it is normal to get a negative score while playing in level 1. For the second part of the scoring function, the score adds a positive number to the score based on how quickly the player reaches the next level. The less time it takes a player to progress, the amount of bonus points that are added to the score increases. The points are then added to the total score at the end of each level. In each level, if the player finishes within 5 seconds, they are awarded 1200 points. If the the level ends within 10 seconds, 1100 points are awarded. For each additional 5 seconds of play-time, the player’s score is deducted 100 points. If the player takes 60 seconds or more to finish a level, 50 bonus points are awarded as a consolation. To ensure the score always increases when the level goes up, all the bonus points are multiplied by the level number. For example, if the player finishes level 2 in 38 seconds, the total number of point awarded are 1000 (500 \* 2). For clarification, at the end of each level, the time played and bonus points added are displayed on the screen for the player. 

*Mouse and Keyboard Usage*

To increase player engagement and enhance players’ coordination skills, our software is a bimanual game that requires both hands to control the mouse (or touchpad) and keyboard while playing. The player needs to press certain keys to start the game, view scores, and see instructions, while the mouse is used for moving the paddle during the game.

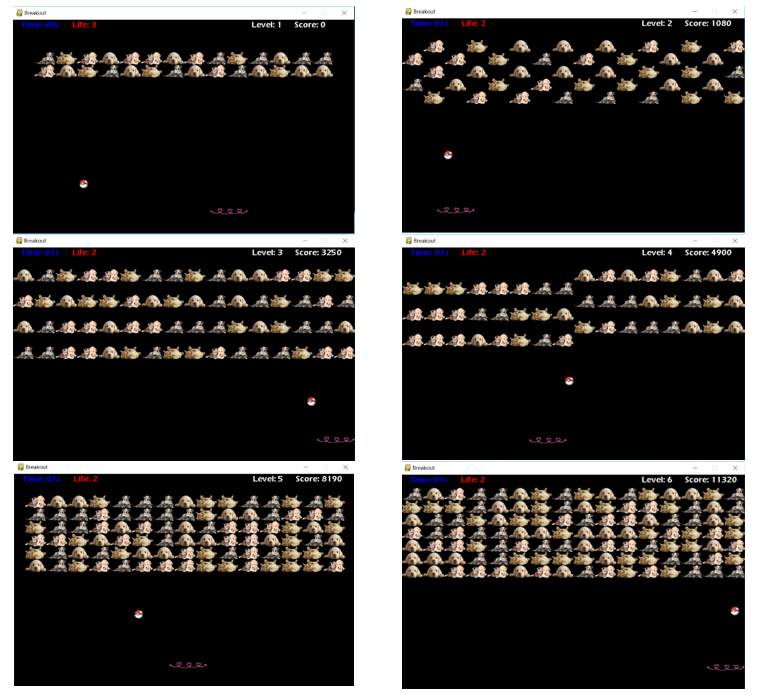
*New Brick Arrangements*

Our initial design for brick arrangement was that for each new level, an additional row of bricks was added. For example, in the first level, two rows of bricks were displayed. As the player progressed in levels, the number of rows increased by one. To increase the attractiveness of our game, we developed new brick arrangements for each level. All the new bricks arrangement codes are written in the reset function. We use multiple for-loops with if statements to create different patterns.

The levels are meant to increase in difficulty as the game progresses. As a starting point, level one has just two rows of bricks with fourteen columns. Level two has four rows of bricks with an interlaced pattern. Level three has four rows of bricks with a space line between each row. Since more space is occupied by the bricks, the moving space of ball is reduced, which increases the difficulty of the game. Level four has six rows of bricks with eight columns either showing on the right or left side. After completing levels one through four, the player moves to a collection of much harder levels. Level five has six rows by fifteen columns of bricks with little gaps on the each of the edges. From level six onwards, the player encounters seven rows by sixteen columns of bricks. In order to achieve the highest scores, it is essential that the player must find the quickest way through the brick arrangement and complete the levels in the shortest amount of time.

*Creative Work*

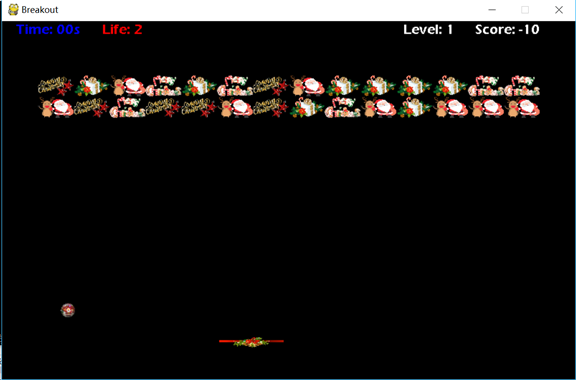
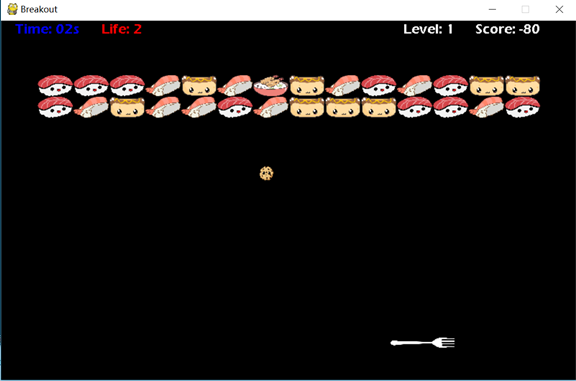
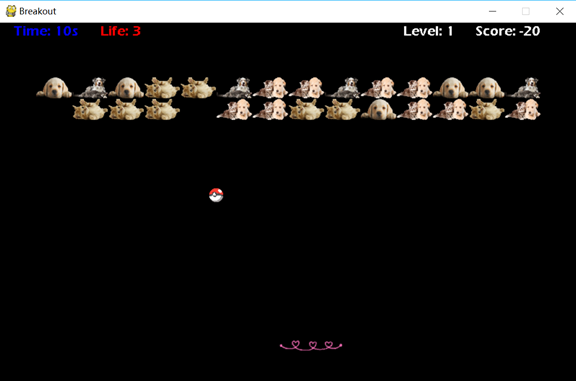
As a group we really tried to be creative with this project. Examples such as our different brick themes, different balls and different paddles show that. We tried to have a little fun with the pop up graphics for “Game Over” and “Level Up”. We also added various sound effects throughout the game to make it more fun.



*Different Game Themes*

Instead of using the traditional brick, ball and paddle images, we decide to find various meaningful images to represent three different themes: pets, food, and Christmas. This feature helps increase player enjoyment, and provides a break from the monotony of staring at a brick wall. When the player opens the game, a theme is randomly selected by the program. The purpose of this setting is to give the player a surprise each time they start new game, like a present on Christmas morning. Additionally, as the style of our start screen is totally opposite to the game inside, the sight of the varying brick images is more impactful.

From the software development aspect, to complete this extended feature, we have found and applied six high-quality png image files related to each theme. Additionally, we used the random function to select a theme for each game played. There were also some little fixes, such as changing the size the image and adding the theme variable to each class.



*Sound Effects and Background Music*

We have included some background music that plays as the game is in duration. There are also sound effects for various events. These include the ball touching the bottom as well as the player losing a life.

Code Listing:

C200\_Breakout\_Team18.py

Highscoremodule.py

HighScores.txt(file that highscores are saved into)