Term Project

CSC 413

SAN FRANCISCO STATE UNIVERSITY

Table of Contents

Introduction

For the final assignment in CSC 413, every student will be completing a term project. This project will consist of two 2D games written in Java, a presentation, and a final documentation for the whole term project. For the first game, every student ***MUST*** implement the Tank Wars game. This is non-negotiable. For the second game, students may select from a list of second game options. The second game choices are open to interpretation, so variations of the games are allowed with instructor's approval.

The goal of this term project is to practice and apply Object Oriented Principles. The purpose of these two games, is for students to implement the tank game, and then re-use a good portion of the first game to write the second game. This means the design and implementation of the first game must practice good OOP. REUSUABBILITY will be the focus of this term project. A more reusable game one will make implementing game two easier (relatively speaking).

A chapter from Game Maker has been provided to get you started on implementing the first game. NOTE we are not writing these games in game maker, therefore you must convert the steps into Java code. There will also be resources that you may use for your games. There is a set for the tank game as well as one for each of the second game choices.

The following document will outline all the information needed for completing the term project. Please read this document very carefully. Mistakes made while implementing or handling submissions will cause points to be deducted.

Requirements

Below is the list of requirements for completing the term project. All steps must be completed.

1. Each student must complete a class diagram for the Tank game
   a. Failure to submit a diagram will cause a 5-point penalty.
2. Each student must implement the Tank Game and submit a working jar of the Tank Game for grading.
   a. Failure to submit a working jar incurs a 5-point penalty.
3. Each student must complete a class diagram for the Second game
   a. Failure to submit a diagram will cause a 5-point penalty.
4. Each student must implement the Second Game Chosen and submit a working jar of the Second Game for grading.
   a. Failure to submit a working jar incurs a 5-point penalty
5. Each student must write and submit documentation for both games (1 document for both games). Submission must a PDF.
   a. Failure to submit a document of type PDF incurs a 10-point penalty
6. Each student must give a 2-minute presentation.

<u>However, points will be deducted for mishandling submissions or changing the name of your GitHub repository.</u>

GitHub Repository

For the Term Project you will be using a GitHub repository to store your code. To make your repository please click the repo creation link posted on ilearn under the Term Project section. PLEASE DO NOT CHANGE THE NAME OF YOUR REPO. DOING SO WILL CAUSE A 10 POINT PENALTY TO BE APPLIED TO YOUR SCORE.

Milestones

Milestone One – Due Saturday March 21st, 2020 @ midnight - NO LATE SUBMISSIONS

Milestone One requires no coding. For this milestone, you will create a class diagram for the tank game. Some sample code has been given on ilearn. You can use this code but DO NOT COPY it verbatim. The class diagram does not need to be 100% perfect or complete. It is not expected that you know the full structure of your tank game this early on. The class diagram should be digitally drawn. Hand drawn submissions are OK but not advised. If you spend the time now drawing the diagrams correctly, you save time when you need to fill them in when completing final documentation. You can use sites like https://www.draw.io/ to draw your diagrams.

The class diagram does not need to list any members (variables and methods). Only classes, interfaces and abstract classes need to be drawn. Also, please correctly show relationships if they exist.

Please submit your class diagram to ilearn as an image or as a PDF.

Failure to submit a diagram will cause a 5-point penalty. Submitting a diagram that is lacking (for example, contains one or two classes) will lose points as well. Please do your best to try and really think about your game design.

NO WORD DOC OR ANY OTHER WORD PROCESSOR SUBMISSIONS ARE ALLOWED. FAILURE TO SUBMIT THE CORRECT FILE TYPE FOR THE CLASS DIAGRAM WILL RESULT IN A 5-POINT PENALTY.

Milestone Two – Due Sunday April 19th, 2020 @ midnight

Milestone two will require you to implement the Tank Wars Game in Java. You may use the sample code posted on ilearn to help you get started. There are resources posted on ilearn for the Tank Wars Game. These contain 2D sprites and sounds. You may also use your own resources as well.

The Tank Wars Game has the following requirements:

1. Tank Game must have 2 Players
2. Tank Game must have tanks that move forwards and backwards
3. Tank Game Must have tanks that rotate so they can move in all directions
   a. Note when only rotating left or right, the tank MUST NOT move forwards or backwards.
4. Tank Game must have a split screen.
5. Tank Game must have a mini-map
6. Tank Game must have health bars for each tank
7. Tank Game must have lives count (how many lives left before game over) for each tank
8. Tank Game must have power up (these are items that can be picked up to modify your tank. What these power ups are, is up to you).

9. Tank game must have unbreakable walls
10. Tank game must have breakable walls
11. Tank Game must have tanks that can shoot bullets that collide with walls
12. Tank Game must have tanks that can shoot bullets that collide with other tanks.
13. Tank Game must come with a correctly populated README.md file that contains what IDE the project was made in, the version of java used to build the game, the current working directory used for the game relative to the GitHub repo and how to run the game and the controls for playing the game.
14. Tank Game must be built into a JAR and stored into the correct folder in the GitHub repo. This will be the folder named jar

When submitting the tank game, make sure ALL code and README.md file is contained in the repo you made. Make sure you have a built JAR for your game and that it is placed in the correct folder in your repository. This folder is named jar. The late deadline for the tank game is determined by the timestamp of the last commit on the ***MASTER*** branch. No other branches will be checked. ***YOU HAVE BEEN WARNED.***

Milestone Three – Due Thursday April 23rd, 2020 @ midnight

Milestone two requires no coding. For this milestone, you will create a class diagram for your second game choice. The class diagram does not need to be 100% perfect or complete. It is not expected that you know the structure of their game this early on. The class diagram should be digitally drawn. Hand drawn submissions are OK but not advised. If you spend the time now drawing the diagrams correctly, you save time when you need to fill them in for the final documentation. You can use sites like https://www.draw.io/ to draw your diagrams.

The class diagrams do not need to list any members (variables and methods). Only classes, interfaces and abstract classes need to be drawn.

Please submit your class diagram to ilearn as an image or as a PDF.

Failure to submit a diagram will cause a 5-point penalty. Submitting a diagram that is lacking (for example, contains one or two classes) will lose points as well. Please do your best to make an effort and really think about your game design.

Milestone Four – Presentation Dates will be posted on iLearn

For milestone four you will be required to give a short 2-minute presentation on your term project. During this presentation you will present either your tank game or your second-choice game OR both. Either option is fine. Slides are not allowed during this presentation. We will treat these presentations as quick demos.

While presenting you will simply introduce yourself, state which game you are presenting. Then simply demo your game and talk about anything you feel was important in the game or you had fun doing while implementing your game. It could also be something that sets your game apart from others, a "special" feature if you will.

A grading rubric will be posted on iLearn, so you know what to expect during your presentation.

You will have AT MOST 2 minutes for your presentation.

Note that when you present, I WILL NOT BE GRADING THE GAMES, but instead grading your presentation skills. Presenting Games not fully completed is OK.

See your respective iLearn sections for presentation dates.

Attendance on these days is mandatory. NO excused absences will be granted unless approved by the instructor with documentation. Missing any of these days will cause a 5-point penalty to be applied to your grade for each day missed.

Milestone Five – Due Monday May 18th, 2020 @ midnight - NO LATE SUBMISSIONS

Milestone five is going to require you to implement your second game choice. Given that you have a completed tank game, implementing the second game should take less time. This is only true if you can reuse some of the code from the Tank game provided it was designed correctly.

For the second game you will need a new repository for the code you will write. . ***Again, changing the name of your repository will cause a 10-point penalty***.

There are no listed requirements for the second choice. But you do need to follow the rules of the game outlined the in Second Game Options section.

When submitting the second game, make sure ALL code and readme.txt file is contained in the repo you made. Make sure a JAR is built for the game like how you built a JAR for the Tank Game. The built JAR needs to be placed in the jar folder in your repo. The late deadline for the second game is determined by the timestamp of the last commit on the ***MASTER*** branch. No other branches will be checked. ***YOU HAVE BEEN WARNED. NO LATE SUBMISSIONS***

Milestone Six – Due Wednesday May 20th, 2020 @ midnight - NO LATE SUBMISSIONS

Milestone six will require you to write documentation for your term project. This document will cover the entire term project from beginning to end. There only needs to be ***one document for both games.*** Please ***DO NOT*** a document for each game.

Your documentation MUST contain the following sections:

1. Title page containing
   a. Student's Name
   b. Class, Semester
   c. A Link to ***BOTH*** repositories.
2. Introduction
   a. Project Overview
   b. Introduction of the Tank game (general idea)
   c. Introduction of the Second game (general Idea)
3. Development environment.
   a. Version of Java Used
   b. IDE Used
   c. Any special libraries used or special resources and where you got them from.
4. How to build or import your game in the IDE you used.
   a. Note saying things like hit the play button and/or click import project is not enough. You need to explain how to import and/or build the game.
   b. List what Commands that were ran when building the JAR. Or Steps taken to build jar.
   c. List commands needed to run the built jar
5. How to run your game. As well as the rules and controls of the game.
6. Assumptions Made when designing and implementing both games.
7. Tank Game Class Diagram
8. Second Game Class Diagram
9. Class descriptions of all classes shared among both Games
10. Class Descriptions of classes specific to Tank Game
11. Class Descriptions of classes specific to Second Game
12. Self-reflection on Development process during the term project
13. Project Conclusion.

When completing this documentation please make sure you are concise but that each section contains enough information. Point penalties will be added for insufficient information or missing sections.

The final documentation ***MUST*** be submitted in PDF FORM. Please submit your final documentation in PDF form to ilearn by the deadline at the heading of this section (note the repetition, must be important). ***Submitting documentation that is not a PDF will cause a 10-point penalty.***

THERE WILL BE NO LATE SUBMISSIONS.

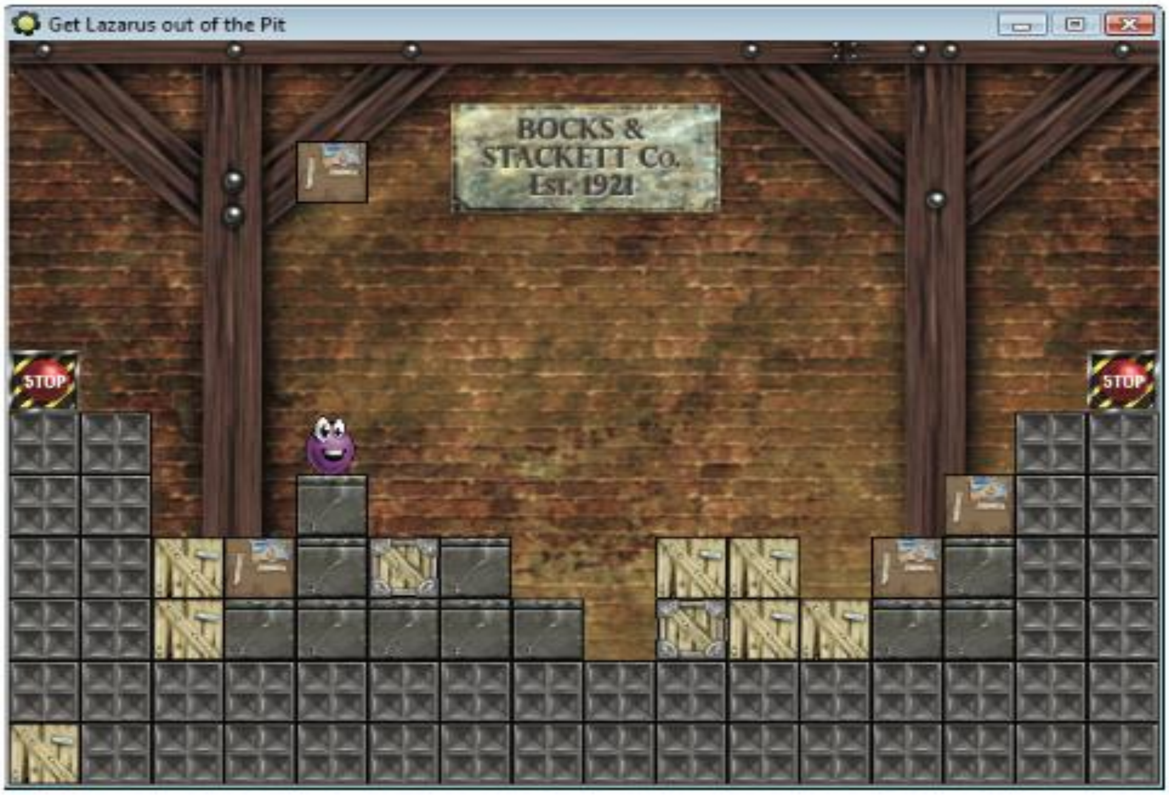
Second Game Options.

Lazarus

Lazarus has been abducted by the Blob Mob, who are intent on bringing this harmless creature to a sticky end. They've imprisoned him at the Blobfather's (sorry) factory, where they are trying to squish him under a pile of heavy boxes. However, they've not accounted for Lazarus's quick thinking, as the boxes can be used to build a stairway up to the power button that halts the machinery. Do you have the reactions needed to help Lazarus build a way up, or will the evil mob claim one more innocent victim?

Each level traps Lazarus in a pit of boxes stacked up on either side of the screen to contain him within the level. The arrow keys will move Lazarus left and right, and he will automatically jump onto boxes that are in his way. However, he can only jump the height of a single box, and stacks two or more boxes high will block his path. New boxes will periodically appear directly above Lazarus's current position and fall vertically down from the top of the screen until they come to rest. This means that the player will be able to use Lazarus's position to control where boxes fall and build a stairway up to the power button.

There will be four different types of boxes, increasing in weight and strength: cardboard, wood, metal, and stone. Falling boxes will come to rest on boxes that are stronger than them, but will crush boxes that are lighter. The type of each box is chosen at random, but the next box will be shown in the bottom-left corner of the window just before it appears. There will be a number of increasingly difficult levels, with higher stairways to build, and boxes that fall faster. When Lazarus gets squished, the level will restart to give the player another try. See Figure 4-1 for an example of how a level will look.



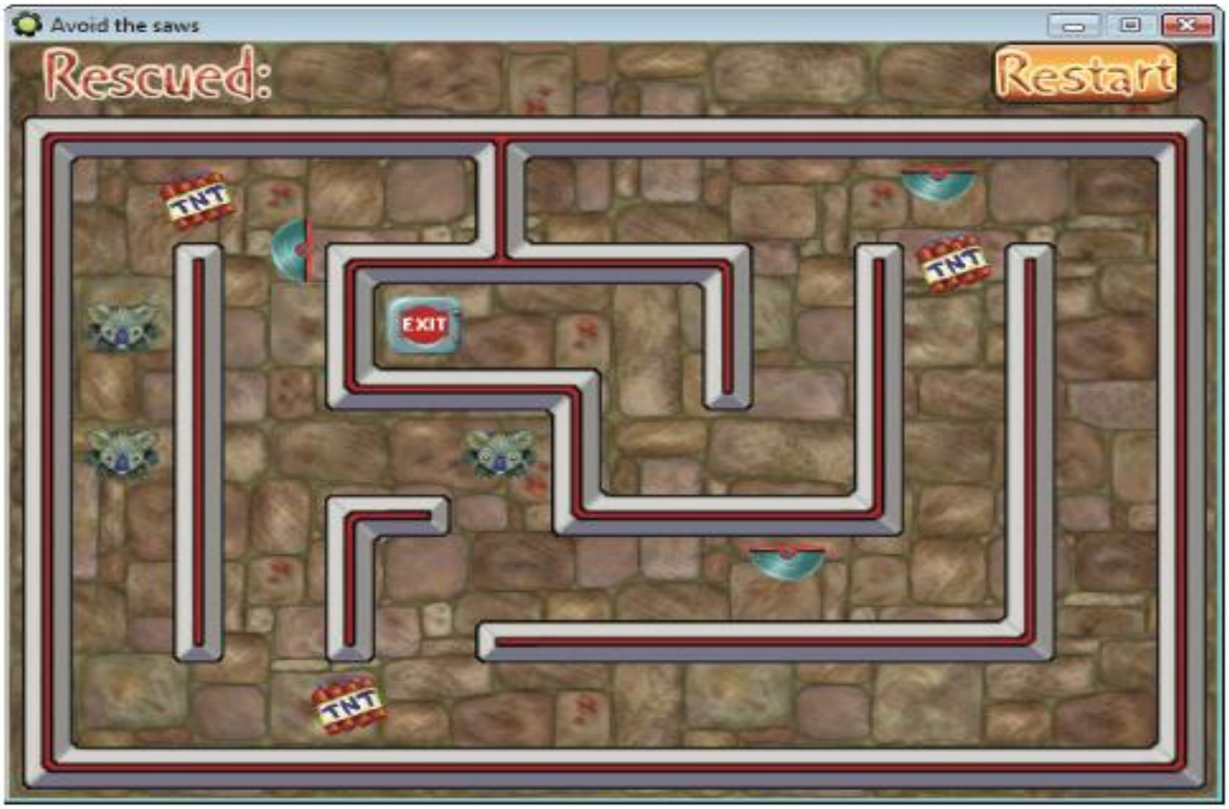
Koalabr8

A colony of koala bears have been captured by the evil Dr. Bruce for use in his abominable experiments. The koalas manage to escape from their cages only to find that the doctor has implanted some kind of mind control device in their brains. The only way they can overpower the controlling effect is to combine their thoughts and all perform the same actions at once. The koalas must work together to find their way past the many dangers in the doctor's laboratory and escape to freedom.

The arrow keys will simultaneously move all of the bears on a level, except bears whose paths are blocked by a wall or another bear. Each level will be a hazardous maze that is completed by getting all of the koalas to an exit. However, if a koala touches a dangerous hazard on the way, then he dies and the level must be replayed. The game will contain a number of fatal and nonfatal hazards shown in the following feature list:

- Fatal hazards
  - ExplosiveTNT
  - Moving circular saws

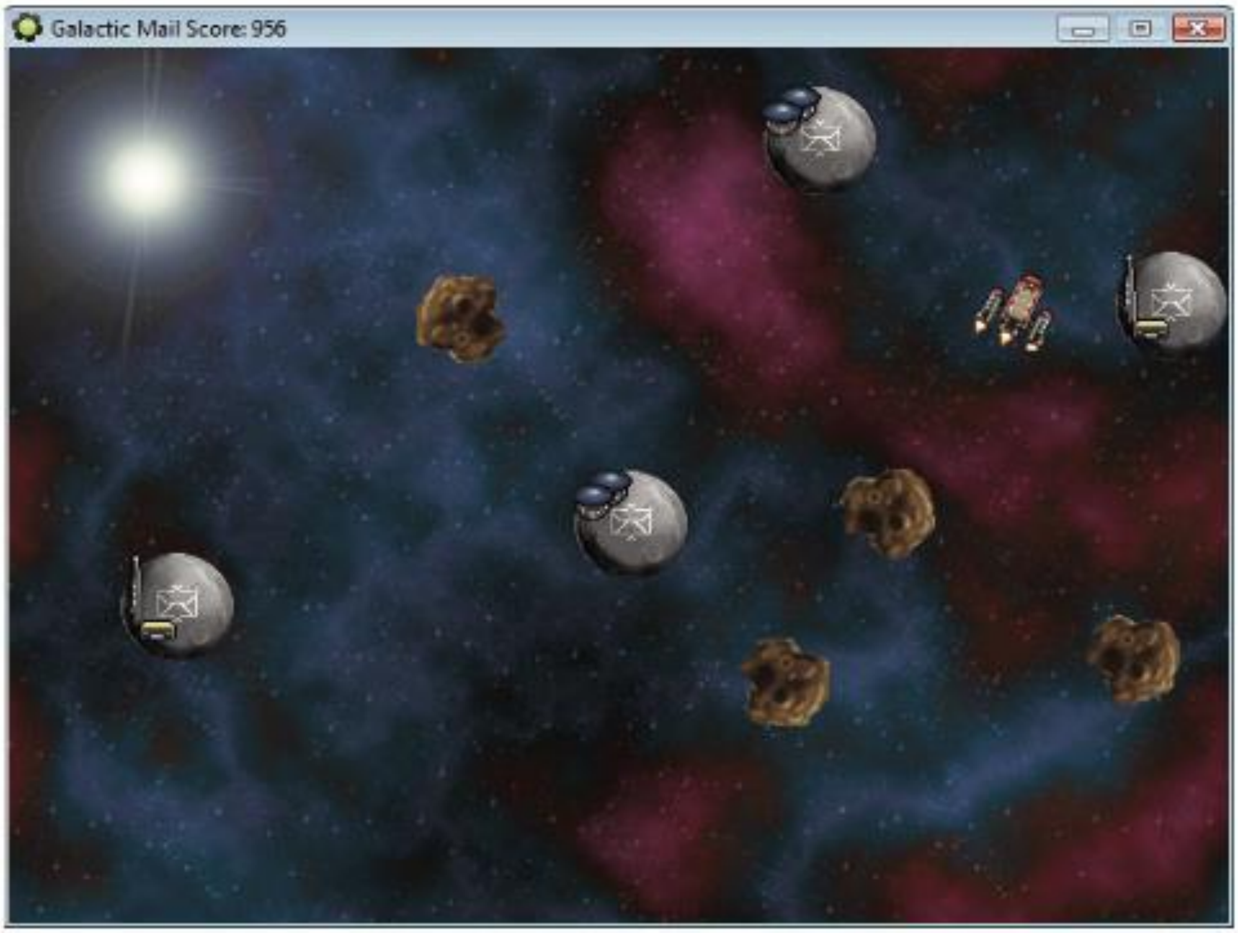
- Non fatal hazards
  - Red exits—Allow any number of koalas to exit the level
  - Blue exits—Allow a single koala to exit the level
  - Locks—Block the path of koalas(red, blue, and green)
  - Switches—Open locked passageways (normal, timed, and pressure)
  - Boulders—Can be pushed by koalas and destroy other hazards



Galactic Mail

You play an intergalactic mail carrier who must deliver mail to a number of inhabited moons. He must safely steer a course from moon to moon while avoiding dangerous asteroids. The mail carrier is paid for each delivery he makes, but pay is deducted for time spent hanging around on moons. This adds pressure to the difficult task of orienting his rickety, old rocket, which he cannot steer very well in space.

When the rocket is on a moon, the arrow keys will rotate it to allow the launch direction to be set. The spacebar will launch the rocket, and the moon will be removed from the screen to show that its mail has been delivered. In flight, the rocket will keep moving in the direction it is pointing in, with only a limited amount of control over its steering using the arrow keys. When things move outside the playing area, they reappear on the other side to give the impression of a continuous world. The player will gain points for delivering mail, but points will be deducted while waiting on a moon. This will encourage the player to move as quickly as possible from moon to moon. There will be different levels, with more asteroids to avoid. The game is over if the rocket is hit by an asteroid, and a high-score table will be displayed. Figure 3-1 shows an impression of what the final game will look like.

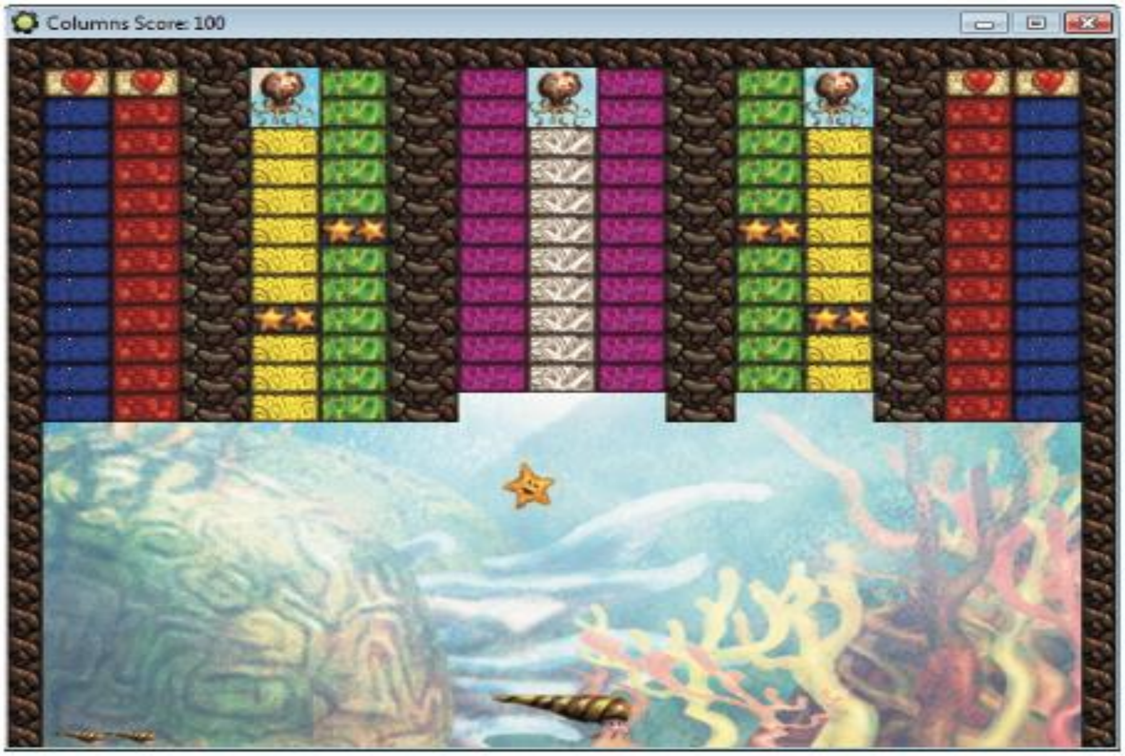


Super Rainbow Reef

The monstrous Biglegs have driven the peace-loving creatures of Rainbow Reef from their ancestral homes. Despite their inexperience in the ways of war, Pop and Katch have invented a way of combining their skills to fight back against the Biglegs. For this incredible feat, Pop must bounce from Katch's shell to attack the evil invaders. Katch must then move quickly to save Pop from plummeting into the deep waters below. The cowardly Biglegs often retreat behind coral defenses, so our heroes must be prepared to smash their way through if they are to finally drive the Biglegs from Rainbow Reef!

There will be no direct control over Pop's movement, and he'll bounce freely around a play- ing area enclosed by walls on all sides except the base. The left and right arrow keys will move Katch horizontally along the base in order to bounce Pop from Katch's shell and stop him from falling out of the level. The collision point along Katch's shell will determine the direction of Pop's bounce, and so allow the player to control his movement. Bounces toward the left will send Pop left and bounces toward the right will send him right. Pop's movement is also affected by gravity, and each time he collides with Katch, he gets slightly faster so that the game becomes increasingly difficult.

The game will have several levels, each containing a number of Biglegs that Pop must col- lide with in order to complete the level. Most levels will also contain coral block defenses, which must be knocked out of the way in order to reach the Biglegs. Breaking blocks will score extra points and special blocks give the player extra rewards, but they don't have to be destroyed to finish a level. If Pop leaves the screen, the player loses a life and Pop is brought back into play. Once three lives have been lost, the game ends and a high-score table is displayed.



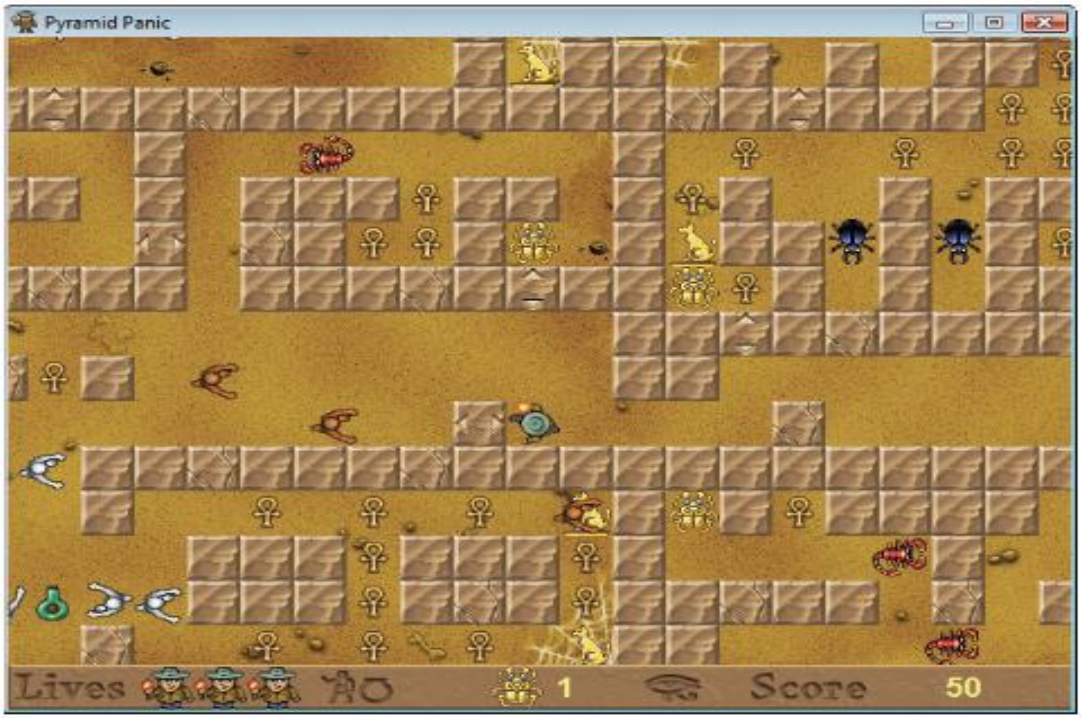
Pyramid Panic

You play an explorer (see Figure 14-1) who has become trapped while investigating a large pyramid complex. All around lie the treasures of an ancient pharaoh, but pyramids are hazardous places and danger lurks around every corner. Deadly scorpions and beetles will block your progress and mummies will hunt you down. Only by keeping your wits about you can you hope to unravel the secrets of the great pyramid and escape as a rich man.

You control the explorer using the arrow keys. Many obstacles will block your path, keeping you from taking the treasures and eventually escaping to freedom. Beetles will only move vertically while scorpions only move horizontally. Mummies move in all directions. These enemies are clever and will react when they see you by trying to catch you and end your explorations. Some wall segments can be pushed, allowing you to reach other areas or hide from enemies. The pyramid also contains scarabs that you can use to make the mummies temporarily vulnerable—allowing you to hunt them for extra points.

Deep within the center of the pyramid lies its greatest treasure, the fabled sword of the sun god Ra. It is this great treasure that casts the unnatural light which reaches throughout the pyramid and allows you to see your way so clearly. It is precious beyond measure, but in taking it you will upset that delicate system and the pyramid will be plunged into eerie darkness.

Only the small glow remaining in the sword will light your way now, and formerly simple puzzles will seem new and challenging. All is not lost, however, for the sword has a second function. When wielding the sword you will be able to press and hold the spacebar to temporarily reactivate its glow. The sword transmutes gold into pure light, lighting your way but reducing your score. When the sword is active, the mummies will flee as they do when a scarab is active, making your journey easier, but draining your wealth.



Grading

Grading will happen throughout the term Project. Each item completed for the term project has its on course grade weight. When totally the weights, your term project is worth 49% of your course grade.

Item	Points
Tank Game	150 (20% of course grade)
Second Game	150 (15% of course grade)
Documentation	75 (7% of course grade)
Presentation	30 (7% of course grade)

For the **Tank Game** your grade will depend on the number of successfully completed requirements.

For the **Second Game** your grade will depend on the completeness of the game based on the description given in this document and the rules of the game.

If you chose a different choice for the second game. A list of requirements needs to be designed by the student and negotiated with the instructor in order make sure there is enough work to be done. Your grade will mostly depend on the completion of the agreed upon requirements.

Both games will also be graded for:

- Smooth Performance (this is more about slow performance because of programming choices rather than hardware).
- pleasant user experience (score, background music, etc)
- Code Cleanliness
- Project Structure

In its simplest terms, I will simply play your games while grading.

Email Request to Change Second Game

If you would like to do something different than what is presented in the Second Game Options Section. Then you must email a request to the instructor. The request must follow the following email format:

Send email to instructor.

Subject: CSC413.NN Spring 2020 Second Game Request

Where NN is your section.

Then in the body of the email you will explain the high-level detail of your game.

Then list out a set of requirements like how I did for the tank game. Don't be afraid to list a lot. Let the instructor decide which ones to remove. It is the instructors job to reduce the scope of your game.

Any emails that break format (most importantly the subject line) will be ignored or returned. You are not allowed to work on the project unless it is approved by the instructor.

List of Penalties

Below is a list of penalties that can be incurred during the term project.

1. 5-point penalty for not submitting Tank Game diagram
2. 5-point penalty for not submitting Second Game diagram
3. 10-point penalty for changing the name of the Tank Game repo.
4. 10-point penalty for changing the name of the Second Game repo.
5. 5-point penalty for not submitting a working Tank Game jar
6. 5-point penalty for not submitting a working Second Game jar
7. 10-point penalty for not submitted a PDF for the term project documentation.
8. 5-point penalty for each unexcused absence during presentations. Excused absences will require documentation to be approved.