# CSC 481/581 Fall 24 Homework 5

## Overview

Your task for this assignment is to add more functionality to your engine. You will submit the code and a writeup of your results, including screenshots. As always, you are expected to abide by the University's Academic Integrity Policy (http://policies.ncsu.edu/policy/pol-11-35-01), which includes providing appropriate attribution for all external sources of information consulted while working on this assignment.

There are 125 points available on this assignment. Students enrolled in 481 are required to complete the first 100 points (Sections 1–3) but may also choose to complete Section 4. Students enrolled in 481 earning scores above 100 will receive a 100 on the assignment (i.e., extra points will not be given as extra credit). Students enrolled in 581 are required to complete all 125 points and will receive a grade as the percentage of the 125 points they earn. All students are required to submit a writeup addressing the sections of the assignment which were submitted.

## Section 1: Memory Management (12.5 points)

You are to build a custom memory allocator for your game engine. Your new games using your engine should use the new custom memory allocator. The previous game(s) you have worked on in past milestones do *not* need to be updated to utilize the custom memory allocator. Your engine must support the ability to use a pool allocator for some type of object (the game object used in your game object model is a good start). The pool allocator should allow the game designer to specify an object type, and the number of objects the allocator will support, reserving the memory for those objects. It may be beneficial to use a wrapper class to allow both dynamic memory allocation and the pool allocator, which will reserve memory and return the correct pointers.

## Section 2: Input Chords (12.5 points)

You are to build on the input representation of your events to detect chords (multiple buttons pressed simultaneously) and/or sequences of keypresses to raise an event that produces different gameplay behavior than the component inputs. The specific chord or sequence is up to you, but it must comprise key presses that alone are already in use for some action or gameplay behavior of the system. The behavior the chord or sequence triggers is up to you to determine, but it must be distinct from other behaviors of the system. A recommendation is to construct new movement tech which requires multiple simultaneous button presses, such as a horizontal dash.

# Section 3: Another Game (25 points)

At this point in the semester, you should have a very functional game engine. Congratulations! Now it's time to test its versatility. Your task for this assignment is to implement a second game using your engine code. You should make use of your engine's functionality to implement a second game that isn't a 2D platformer.

One of the two following games are good choices, but not the only options:

- One level of a classic Bubble Shooter. If you are not familiar with the genre, play some of the free games on http://bubbleshooter.net/ to get a feel for how they work.
- One level of the classic Space Invaders game. If you are not familiar with the game, simply search for "space invaders" on http://www.youtube.com/ and play some of the videos that result.

You may make these games single or multiplayer. Regardless of the choice, it must be playable, and it must use your game engine.

When implementing this second game, pay careful attention to how much reuse you get out of your core engine code. To include in your writeup, do a diff between your code for the first game (the 2D platformer) and code for the second game. What percentage of the lines of code are different?

Below are some suggestions for some games you may wish to implement. You are not required to implement a game from this list, and, with instructor permission, you may instead implement a game with your team members which has a larger scope.

## Bubble Shooter

If you choose to implement a Bubble Shooter, you should have a minimum of two colors of bubbles and there should be at least two layers of bubbles upon startup. There's no requirement to add additional layers during gameplay, but you should lower existing bubbles periodically as they do in a full bubble shooter implementation. Your gun should rotate and fire according to keyboard or mouse inputs. Also, make sure your game will end if bubbles touch the bottom of the screen.

## Space Invaders

If you choose to implement a Space Invaders game, you should have a minimum of two rows of spaceships upon startup. The ships should move across and down the screen as they do in the real game. Your gun turret should move and fire according to your keyboard or mouse inputs. You are not required to have defense structures typical of the first three levels on the full space invaders game.

## Other Options

You may choose to implement any other game you like. In the past, students have created versions of:

- Snake (https://snake.googlemaps.com)
- Gorillas (https://en.wikipedia.org/wiki/Gorillas_(video_game))
- Brick Breaker (https://en.wikipedia.org/wiki/Brick_Breaker)

etc. If you have questions about expectations for these or any other games you wish to implement, please reach out to the instructor.

# Section 4: Another Game (12.5 points 581 require/481 optional)

For this part of the assignment, you are to implement another game. You may choose to implement one of the other recommended games from Section 1, or another game of your choosing. If you choose to implement a different game, it is probably wise to consult with the instructor prior to beginning work on it. The criteria for this part of the assignment are the same as Section 1, including the code diff for the writeup.

# Writeup

Now that you have designed and implemented a number of features, write a 3-5 page paper summarizing your work, and the design your team made, with a minimum of 2 ***single spaced pages.*** It is **STRONGLY** suggested that you do not limit yourself to only answering the questions posed in this assignment, think creatively about what you have done. What did the design enable, why did you and your team make the design decisions you did, what will those decisions enable in the future? What worked the way you wanted and what didn't? Successful writeups will contain evidence that you have thought deeply about decisions and implementations as what they can produce and have gone beyond what is written in the assignment.

Additionally, reflect on the overall design of your engine as it has evolved over the course of the semester. Please explain how successful (or unsuccessful) you were at designing for reuse. What systems did you have to change to get your second (and third) game(s) to work? How did you change them? If you were going to start over again to design your game engine again, what would you do differently? What would you do the same? Why? Please use a diff tool to determine the number and percentage of lines of code that remained the same across your first, second, and third games. Note: your grade will not be determined by whether this percentage is high or low, but that you have a well-constructed explanation for why you got the result you did and whether you did (or did not) meet any goals you set for yourself.

As an appendix to your paper, please include relevant screenshots of your program and support your points by referencing the screenshots. These screenshots do not count towards the page requirement. Your points from the writeup represent ***50%*** of the total grade for the assignment.

# Submission

There are three deliverables for this assignment:

1. A team submission of the game engine. This is a single submission for all team members, and it should include the general features used in each team member's game.
2. Individual game submissions showcasing engine features.
3. An individual Reflection writeup, which discusses the design decisions and choices made in the games and engines.