CS 111 Final Project  
Self Assessment

Your group should fill out one copy of this form and include it with your assignment when you turn it in.

# Group

*Who’s in your group?*

1. Joshua Lane
2. Connor Kotwasinski

# Goals

*Say a few words about what you wanted the game to be like. Note that if you just wanted to write some code so you could get an good grade on the project, it’s fine to admit that.*

Our game is based on the game Minecraft, and we wanted our game to be similar to Minecraft when making it. For our game, the player starts in their underground home and they get to travel into cave systems, the nether, and the end. Like in Minecraft, you can win by killing the ending dragon, but if you prefer not to try and win you can just go exploring.

# Lessons learned

*What went right?*

We were able to make it so you could explore through multiple rooms and explore everything inside. We also were able to make the players and mobs interreact with each other so that the player could potential die in the game while playing, making the game more interesting. Players also could to choose whether to just explore our world or try to win the game, giving the player more things and options in the game.

*What went wrong?*

We actually felt like the starter code was quite limiting in the way it was designed. Accordingly, we had to go and modify most of the starter code that we used so that it would better fit with the game we designed. Part of what made the starter code particular unhelpful for us was that we did not know it existed until we had coded a significant part of our project from scratch, which made it harder to use the code we had already written with the starter code.

*What do you wish you knew when you started?*

We wish we would’ve known about the starter code earlier so we could have saved ourselves the time when coding things similar to that in the starter code and so our code would have interacted better with the starter code.

# Annoying grading bookkeeping

## Types

*What are the types you added, and what are they for?*

1. We added a health type which allowed us to make a health bar for the player. Which ultimately dictated when the player would die and this lose the game.
2. We made our own struct for room rather than using the premade one. The room struct contained information about objects inside it and adjacent rooms that could be entered.
3. We made a sub-type of object called chest. Chests our objects within rooms which have their own objects inside them which can be viewed.
4. We made a generic mobs structs so we could make specific mob subtypes.
5. We a made a zombie\_pigman type. This Types is used to create the zombie pigmen found in the nether.
6. We made a struct called creeper. This type is used to create creepers which are found in the cave system in our game.
7. We made a struct called enderdragon. Which allowed us to create the enderdragon in the end which needs to be defeated in order to win the game.

## Fields

*What are the fields you added, what types did you add them to, and what are they for?*

1. We added a new field called name for objects. This field is used to put the name of every object we made in string form. This field was used to help us code. For instance, the field is called upon to help check if rooms are adjacent.
2. We added the field called number to the health type because it allowed us to set a starting amount of health a player started with and change the number field to reflect the damage taken when fighting mobs.
3. We made a field called viewroom for our room struct. Which is a list of objects in the room and rooms that can be entered from the given room.
4. The chest type as a field called viewchest. The field stores a list of all the objects within a chest.
5. We made a field called fly for the enderdragon type this field is used to store information about what happens when the enderdragon takes flight.
6. We made a field called fireballdamage for the enderdragon type to store the amount of damage the enderdragon inflicts when he uses his fireball attack.
7. We made a field called clawdamage which stores the amount of damage the player takes on when the ender dragon uses his claw attack.

## Procedures

*What are the procedures you added or significantly modified from their original form, and what are they for?*

1. We made a procedure called actions which displays all possible commands the player can execute while player our game.
2. We made a procedure called update-healthbar which subtracted a given amount of health from the player health bar and ended the game if the healthbar dipped to zero or below. This was usually used when the player is fighting mobs.
3. We made a procedure called viewhealthbar which told the player how much health he currently has when he typed the command into the command window.
4. We made a procedure called viewroom which takes a given room and outputs a list with the objects in the room and rooms that can be entered from the room.
5. viewchest is a procedure that allows you to look at the list of objects within a chest.
6. The procedure enter allows you to enter rooms that are adjacent to the room you are currently in.
7. We made a procedure called attack\_enderdragon which takes an enderdragon as an input and allows you to attack the enderdragon. This code uses random number generation to determine what happens to the player and the enderdragon during this fight.
8. The walkthrough procedure speed runs the game to the end.

## Methods

*What are the methods you added or significantly modified from their original form, what types were they added to, and what are they for? Note that if you have three different methods for the same generic procedure, list each one separately.*

1. For the object struct we added a new method called descriptions. This method takes an object an outputs the adjectives field, which is a description of the object.
2. We made a method called attack\_pigman which allows the player to attack the zombie pigman, inflicting damage on it and taking damage himself.
3. We made a method called attack\_creeper which allows you to attack a creeper and take damage when he explodes.

## Total stuff we built

*Write the total number of items listed above.*