# AP Computer Science Final Project - README

Instructions:
The first step in creating an excellent APCS final project is to write up a README. At this stage, this README file acts as your **project proposal**. Once you've filled in all components, Shelby will read through it and suggest edits. Ultimately, you need a document that adequately describes your project idea and **we must agree on this plan**.

Have one member of your group **make a copy of this Google Doc**. Then, they should share it with all other members so that every group member has edit permissions.

There's a lot of parts of this document that you might not have full answers for yet. Because you haven't written the program yet, it's difficult to think about the **instructions** or **which group members will do which parts**. Even though this is hard to think about, you must have something in these sections that acts as your current plan. However, during the course of the project, you'll **continuously update this document**. This means that you will not be *held* to exactly what you put here - components of this document can change (and it's pretty common!).

There is one exception: the **Features List** section. Once Shelby OKs your README, the Features List section **cannot be modified**. For this reason, it is most important that you get a solid idea of what you want to make and the primary features it will have *now*.

Talk with your group. Consider drawing some pictures of what you think your project might look like. Be precise. When you're ready, fill this out together. Each component in brackets below ( [these things] ) should be replaced with your ideas. Note that there are several sample READMEs posted on this assignment for you to use as guidance.

**-------------------When README is finalized, remove everything above this line--------------------**

# Mask On!

**Authors**: Felicia Zhang, Roshni Parulekar-Martins, Emily Tumacder
**Revision**: 4/ 23/ 2021

**<u>Introduction</u>**:
[In a few paragraphs totaling about ½ page, introduce the high-level concept of your program. What this looks like depends a lot on what type of thing you are making. An introduction for an *application* will look different than one for a *game*. In general, your introduction should address questions like these:
What does your program do?

- Our program takes the user on a journey through the times of 2020, across California, and provides them with a fun experience.

What problem does it solve? Why did you write it?

- Our problem solves the main issue in 2020: COVID. We wanted to make a game with real-life applications.

What is the story?

- This is the story of the best year ever, where everything ~~went badly~~ went great! But let's not kid ourselves, it would have been better if we could have prevented COVID. It's your task to go back in time to 2020 and see if you can save people from COVID, ending the suffering in the lives of millions of people in California.

What are the rules? What is the goal?

- Each tier (level) will have different rules and goals. In the yellow tier, you must try to mask everybody and take infected people to the hospital to get better. In the orange tier, your hospital system is overwhelmed with cases, so you must build a new hospital. In the red tier, you must open up vaccination clinics and try to get people vaccinated. In the purple tier, a new strain that is vaccine-resistant develops, so you need to recruit researchers to incorporate defense against this strain in the vaccine.
- The goal in all tiers is to bring the case count back to 0. This is achieved by making sure no one is infected, everyone is wearing a mask (lower levels), and everyone is vaccinated (higher levels).
- Rules:
  - You must wear PPE before taking infected people to the hospital.
  - You can only build new hospitals and vaccination centers in public places.
  - You can only move one square at a time on the grid.

Who would want to use your program?

- Anyone who wants to make 2020 a better year in their mind can use our program.

What are the primary features of your program?

- The primary features of our program are the map of California, the tiers of the game, and the sidebar with the case count, inventory, and tasks.

**<u>Instructions</u>**:

[Explain how to use the program. This needs to be **specific**:

Which keyboard keys will do what?

- The arrow keys will be used to manipulate your character across the screen. You can only move one square at a time with each press of an arrow key (holding an arrow key down will allow further movement). The key Q will be used to mask people you intersect with, while W is used to put PPE on yourself.

Where will you need to click?

- Clicking is only needed in the start and home screen. In the start screen, there's a button to start the game and a button for the instructions. In the home screen, you have to click which county you want to solve the COVID19 problem in.

Will you have menus that need to be navigated? What will they look like?

- The start menu will need to be navigated, with the start and instructions buttons.

Do actions need to be taken in a certain order?

- There is no certain order the tiers need to be solved in, they just all need to be solved by the end of the game.

<u>**Features List (THE ONLY SECTION THAT CANNOT CHANGE LATER)**</u>:

**Must-have Features**:

[These are features that we agree you will *definitely* have by the project due date. A good final project would have all of these completed. At least 5 are required. Each feature should be fully described (at least a few full sentences for each)]

- There will be a menu screen with the title, start button, and instructions.
- We need an interactive map so that players can “go” to different counties in California. By clicking on a certain area of the map, they will be able to complete a tier.
- In the main game screen, there will be a sidebar with your inventory, which will show how much ppe and/or masks you have; a task bar, for the extra tasks that need to be completed that level; and a COVID tracker, which will fluctuate depending on the number of people in the grid that are infected. There will also be a timer that shows how long the user took to complete a tier.
- The main game screen will have a grid with people, hospital buildings, vaccination centers, private residences, public spaces, and a PPE factory.
- The people need to be able to move on their own across the grid.

**Want-to-have Features**:

[These are features that you would like to have by the project due date, but you’re unsure whether you’ll hit all of them. A good final project would have perhaps half of these completed. At least 5 are required. Again, fully describe each.]

- Each character and building will have a different icon, reflective of what their character/building represents or what their character is wearing (PPE).
- We would like to have constant background music playing, which may increase tempo as covid cases rise.
- The infected people die after becoming infected for a certain period of time. If more than a certain number of people die, you must restart the task.
- The Player is able to choose accessories (not including any PPE) to look how they want to.
- The SideBar has more specific tracking features, and it tracks the number of people in every situation, not just infected.

**Stretch Features**:

[These are features that we agree a *fully complete version of this program would have, but that you probably will not have time to implement*. A good final project does not necessarily need to have any of these completed at all. At least 3 are required. Again, fully describe each.]

- We would like to have a multiplayer option, where two people can work together to solve the COVID crisis.
- The county that your mouse is hovering over will light up.

- We would like to have a game without the concept of a grid, just free placement of objects and people based on pixels.

**Class List**:
[This section lists the Java classes that make up the program and very briefly describes what each represents. It's totally fine to put this section in list format and not to use full sentences.]
- Main: runs the program, contains the main method, creates the window
- DrawingSurface: window
- Map: shows California
- Menu: shows info and instruction
- Timer: keeps track of time in timed version
- Covid Tracker: a bar that changes depending on how many infected are on the screen (single color).
- Inventory: a box that represents the player's inventory.
- Tier: a class for a level, it has the grid for the game
    - PurpleTier: fourth level (hardest)
    - RedTier: third level
    - OrangeTier: second level
    - YellowTier: first level (easiest)
- Person: represents each individual in the population in the grid
    - Doctor: people who have skills to work in a hospital
    - Researcher: people who have skills to improve vaccines
    - Player: represents the user

**Credits**:
[Gives credit for project components. This includes both internal credit (your group members) and external credit (other people, websites, libraries). To do this:
- Everyone: Tier, Person, Main, DrawingSurface
- Felicia: first 4 classes not including above
- Emily: next 4 classes not including above
- Roshni: remaining classes
- Java.awt.Color
- Java.util.Timer
- List the group members and describe how each member contributed to the completion of the final program. This could be classes written, art assets created, leadership/organizational skills exercises, or other tasks. Initially, this is *how you plan on splitting the work*.
- Give credit to all outside resources used. This includes downloaded images or sounds, external java libraries, parent/tutor/student coding help, etc.]

External resources: Processing (external java library), pixilart.com (drawing software), GridWorld, unsplash

Roshni: Map, Display, Menu, Location, DrawingSurface, Main, UML, README
Felicia: CovidTracker, TimerDisplay, UML, README
Emily: Inventory, Tier, Player, images, UML, README