# AP Computer Science Final Project - README Template

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 **and with Mr. Shelby** so that every group member has edit permissions, and Shelby can add comments on your ideas.

There are 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----------**

# [naM-caP]

**Authors**: Desiree Poon, Brianna Wang, Jenna Wang
**Revision**: [4/29/2022]

**Introduction**:
naM-caP is an action maze chase game where the player controls a single ghost through an enclosed maze. We wrote this game to provide an alternate perspective of the ghosts in the game Pac-Man. This game is easy to understand and fun for people of all ages, especially those who enjoy chasing games like the original Pac-Man. The primary features include the choice to pick a ghost to play, a maze with obstacles and fruits, and the character naM-caP, who will chase and be chased.

When the program starts, players will be able to choose their ghost and read the directions for the game. The player will begin in the center of the map, inside the box. They will use the four arrow keys to control their ghost. When the game begins, naM-caP will move towards the kiwis that are scattered throughout the map while avoiding the player, and when it is in the superpower phase after eating a kiwi, it will try to chase the player to eat them. During this time, the player will try to run away from naM-caP until it exits this phase. If the player is caught, the player loses one life. When naM-caP is not in its superpower phase, the objective of the player is to eat naM-caP to earn points. The overall goal of the game is to win as many points as possible before losing all three lives. After losing all three lives, the player is given the option to play again.

**Instructions**:

1. The game will start by showing a screen where you will be able to choose the ghost you wish to control and a start button below it. The available options are Blinky, Inky, Pinky, Clyde, and a ghost which you can choose the color of.
2. Click the start button at the bottom of the screen to begin the game.
3. A map will load — the game will start with your character in a box at the center of the map, and you may exit the box once the game finishes its countdown.
4. Use the arrow keys to control your ghost.
5. Your goal is to chase and eat naM-caP. If you manage to successfully eat naM-caP, the round will reset and you will return back to the box.
6. If naM-caP goes on a rampage, your goal is to make sure you are not caught. If you are caught, that will result in the loss of one life.
7. There are three lives in total, and the game ends once you have lost all lives.
8. An ending screen will pop up and you will have the option to exit or restart.

**Features List (THE ONLY SECTION THAT CANNOT CHANGE LATER)**:

**Must-have Features**:

- A starting menu with a start button and a panel that allows users to choose which ghost to control. The available ghosts are Blinky (red), Pinky (pink), Inky (teal), and Clyde (orange).
- Cohesive graphics that follow a common scheme. This game will adopt a pixel art style to mimic the original game naM-caP is based on.
- The behavior of naM-caP follows a set pattern. Before naM-caP eats the kiwi, naM-caP's main priority will be to eat the kiwi spread across the map and the player's main priority will be to chase naM-caP. After eating the kiwi, naM-caP's main priority will be to chase the player and the player's main priority will be to avoid naM-caP.
- The ability to control the ghost using the arrow keys. The arrow keys will not need to be held down. Instead, once clicking on an arrow key, the player will continue to move that way until a different key is pressed or if it encounters an obstacle.
- A life system. The player has three lives in total, and every time the player is eaten by naM-caP, the player will lose a life. The game ends when all lives are lost.

- An end screen with the score displayed and the option to play again. The text “Game Over” will be displayed on the top with a button to restart below.

**Want-to-have Features**:

- High score system. The game will keep track of the player’s highest score and display it on the top of the screen.
- Different map choices. The player may choose which map they wish to play on the starting menu. This is to create more variety within the game. Each map will have a different number of obstacles depending on the difficulty of the game.
- A fifth ghost option. This ghost will allow the player to customize the color of their ghost at the beginning of the game by inputting RGB values.
- Different fruits that will provide different power-ups for naM-caP and the player. The player may choose which fruits they wish to include at the start of the game. One possible fruit idea is to change the speed of naM-caP or the player.
- Players will be able to choose between single-player and multiplayer mode. In single-player mode, it will be just the player and naM-caP. In multiplayer mode, other players may play on the same keyboard as other ghosts to chase naM-caP. The more players there are, the more buffs naM-caP will have (faster speed, longer superpower phase).
- Computer-controlled ghosts. If the players desire, they may add computer-controlled ghosts, assuming they have less than four players in their “team.” These ghosts will follow a specific algorithm to assist the player in catching naM-caP.

**Stretch Features**:

- Smartness level. The player may choose how “smart” they wish naM-caP will behave on the starting screen. The smartness level will range from novice to expert. The smarter naM-caP is, the harder it will be to catch naM-caP.
- Multiplayer functions that will allow user control for naM-caP and the other ghosts on different devices as well as the ability to randomly match for a game. (Firebase)
- Certain maps will include different features, such as portals, secret passages, and hiding spots. This will add more complexity and allow for more strategies in the game.

**Class List**:

- Main: Initiates the game
- DrawingSurface: Draws all of the graphics, takes user input (choices and keys), and moves players according to keyPressed.
- Map: Holds the obstacles and map layout (may have subclasses if we have time for multiple maps). Has a 2D array of booleans (maybe character later for k = kiwi, p = player, n = namCap, f = wall, t = path;)
- NamCap: Contains the algorithm and graphic for the character naM-caP.
- Player: Contains the graphics for the different ghost options.
- Fruit: Superclass for all the different power-ups included in the game.
- Kiwi: Extends Fruit, represents the kiwi that allows naM-caP to power up

**Credits**:

- Responsibilities:
    - Desiree: Map, DrawingSurface, Main, UML diagram
    - Brianna: Player, Main, UML diagram
    - Jenna: NamCap, Fruit, Kiwi, UML diagram
- Other credits: Images of ghosts from Google, recursionIn2D, processing
- Alpha Release:
    - Desiree: Main, map, drawing surface (draws a boolean grid that is set to all false and red. Green = true (part of the path that the objects can run through), edited README, wrote javadocs for map, drawing surface, added in processing
    - Jenna: Created the NamCap, Fruit, and Kiwi class. Made Kiwi the subclass of the Fruit class and added methods and fields accordingly. Added a picture for NamCap. Wrote javadoc for all mentioned classes.
    - Brianna: Created Main, Player, edited UML diagram, edited README, added methods and fields for Player, wrote Javadocs for Player, added pictures for all ghosts and kiwi