**A screenshot of a computer

Description automatically generated with low confidence**

**Objective:**

Students will display their knowledge of all course material to build a complete computer program with a real-world application.

**Problem:**

**Pacman Rules:**

The rules of the game are quite simple. The goal is for Pacman (the player) to collect all the dots on the map, while avoiding the enemy ghosts (AI) that are chasing you. The game is won when all the dots on the map have been collected. The game is lost when a ghost catches the player.

All characters in the game are bounded by that level’s map. The map contains walls (W) that restricts the characters from passing through them, creating a maze of sorts. The dots (.) are what need to be collected for the player to win the game. The characters can only travel through the paths between the walls.

Below is the map that will be used in this project. The input text file will look slightly different, as it will show you the starting position for both Pacman and the Ghosts, and will not contain the outer walls. You can obtain the text file by clicking "Download Starter Code" on the right side of the page on Mimir, or by downloading it from OnQ.

**W W W W W W W W W W W**

**W . . . . W . . . . W**

**W . W W . W . W W . W**

**W . W . . . . . W . W**

**W . W . W W W . W . W**

**W . . . . . . . . . W**

**W . W . W W W . W . W**

**W . W . . . . . W . W**

**W . W W . W . W W . W**

**W . . . . W . . . . W**

**W W W W W W W W W W W**

**Instructions:**

Write a Pacman program in C that uses the following specifications:

* Read the map from a file called “map.txt” into an array created using malloc(). Make sure to implement correct memory management techniques for full credit.
* Have the user enter an input of ‘w’, ‘a’, ‘s’, or ‘d’ that **instantaneously** moves Pacman in that direction if there is not a wall. If there is a dot in the new location, Pacman “eats” it and it is removed from the map.
* Implement 2 Ghosts (Enemies) to move around the map. If they can see Pacman (there are no wall tiles between them), have the ghost move toward Pacman until it does not see him anymore.
  + **This is best done with a recursive function.**
  + The ghosts are allowed to pass through each other.
  + If you are an advanced programmer, investigate how to implement Dijkstra’s Algorithm. Keep in mind that this is beyond the scope of this course, and as such is not required.
  + See the Sample Output Video that shows the ghost chasing Pacman.
* Your code must contain the following functions. **You are encouraged to create more functions to help you with your project.**
  + The void function **printMap()** will output the current map to the console.
  + The function **isWall()** that accepts coordinates & a direction as parameters. It returns 1 if the nearest tile in the specified direction is a wall tile **or out of bounds**, and 0 if not.
  + The function **winCheck()** function will determine if the win condition has been met. Returns 1 if the game has been won, and 0 otherwise.
  + The function **loseCheck()** function will determine if the loss condition has been met. Returns 1 if the game has been won, and 0 otherwise.
* The updated map must be printed to the console after every move.
* When the win condition has been met, the following message must be printed:
  + Congratulations! You win! Press any key to exit the game
* When the loss condition has been met, the following message must be printed:
  + Sorry, you lose. Press any key to exit the game

Your program will need to take user input for each command from the player. Receive this input using the **getchar()** function.

Comments are mandatory for this project. Add comments as necessary for important parts of your code, such as function calls & definitions, conditions, and calculations to explain what the program is doing.

Your output must match the sample output below as closely as possible; otherwise, the auto grading software will not be able to grade your project, which may affect your mark.

**Sample Output:**

<https://stream.queensu.ca/Watch/f4DGw2a6>

**Helpful Hints:**

This project may look daunting at first; however, these tips should help you get started and keep you coding in the right direction.

* Start by setting up 2 different maps: one to hold all the dots and the other that is shown to the player. Whenever you need to replace a dot, you can then copy the contents from one map to another!
  + Having a function that reads the map from a file “map.txt” into a 2D Array and returns a pointer to that array van be very helpful for this, as well as code organization.
* Similarly, your check function can recursively call itself for every adjacent cell if the current cell contains a 0. Make sure you have a way to check if the cell has already been revealed (or else face an infinite loop)!
* You can check if the win condition has been met by counting the number of dots still in the dots map.
* Complete the isWall() function first. That function is used in almost every function in the program that requires moving a character. Also complete the printMap() function early on, as it will help you visualize your maps when debugging.

**Submission Instructions:**

Create your program using CLion and upload it here for grading. Your program file must be named “**apsc143project.c**” for your project to be graded. Do not include any personal information (student number, name, etc.) in your submission.

To compile and run the program in the Mimir IDE, enter the following command into the terminal:

gcc apsc143project.c -lm -o project && ./project

Refer to the project rubric on OnQ for a detailed breakdown of the grading criteria. Your submission must adhere to the project rules as outlined in the submission policy document for this course, which can also be found on OnQ. There is zero tolerance for plagiarism in this course. This auto grading software will automatically flag potential cases of plagiarism, which will be reviewed by the instructors.

More information on the specific definition and repercussions of plagiarism can be found in the “Begin Here (About This Course)” module in the “Plagiarism” video.