```
        Main function:
Declare boolean value “userCont”, set to true

while(userCont):

        Display 3 menu options:
        1. Load premade level
        2. Open custom level
        3. Exit

        Take integer input from the user to select the one they want.

        If 1 is selected:
                Display a menu with premade levels.
                Take integer input from the user to select which one they want to play.
                Find the level in the program's folder and open it as istream.
                Declare a “levelType” class called “level” with the istream as the argument.
                Call gameLoop() function with “level” as argument.

        If 2 is selected:
                Print “Enter name of level to load: “
                Take string input from user
                Search folder the program is in for a level with a file name that matches the string the
                        user entered.
                If found, open the level as ifstream. Else, return to the main menu.
                Declare “levelType” class called “level” with the istream as the argument.
                Call gameLoop() function with “level” as argument.

        If 3 is selected:
                Switch userCont to false.

END WHILE

        gameLoop function:
Declare char value “userInput”.

While(true):
        Take char input from user, and store it in “userInput”.
        Declare integer “toContinue”.
        Call “level”'s iterateMap function with userInput as argument. Set “toContinue” equal to its
                return value.
        Call “level”'s drawMap function.

        If toContinue == 0: continue while loop.
        If toContinue == 1: call drawWin function, break from while loop.
        If toContinue == 2: call drawLose function, break from while loop.
END WHILE
```

Level class: Values: int vertLevelSize = 25; int horizLevelSize = 80, playerType player, enemyType enemyArray[25], char levelArray[80][25]

function checkEnemyPositions(int x, int y):
iterate through the enemyArray , and returns true if an enemy positon matches the coordinates passed as parameters

function checkIfWall(int x, int y):
checks levelArray, returns true if there is a char representing a wall (-, |, or +) at the given coordinates

function iterateMap (takes parameter “input”) (returns int, 0 to continue the game, 1 if the player wins, 2 if the player loses):
If player wants to move up, check if enemy or wall is directly above the player. Repeat for the other directions. If there is no enemy and no wall, call “player”'s appropriate movement function (moveUp() for moving up, etc.) If the tile the player is moving to is the goal tile, return a value of 1.
Then iterates through enemyArray, calling the moveEnemy() function in each one, having them follow the paths programmed into the level file.
Then iterates through enemyArray again, this time checking the vision cones of each enemy. If the player object's coordinates match up with any coordinates in any vision cones, return a value of 2. Else, returns a value of 0.

Information seen by player:

Screen 1: Main Menu.

“What would you like to do?
1. Load premade level
2. Load custom level
3. Exit”

Screen 2: Premade level selection

1. **level title (hard coded)**
2. **level title (hard coded)**
3. **level title (hard coded)**

Screen 3: Custom level selection

“Enter the name of your custom level. Make sure it is in the same folder as the game executable.
_”

Screen 4: Game screen

A representation of the level. '+', '-' and '|' represent walls, '.' represents floors, '@' will represent the player, and '<', 'v', '^', and '>' represent enemies (the open end is the direction they are facing.

Screen 5: Win screen

“Congratulations! You have successfully snuck into a place! Press ENTER to return to main menu.”

Screen 6: Loss screen

“You were spotted. That sucks, but you might get it next time! Press ENTER to return to main menu.”

Data to be stored:

levelType class:
char levelArray[80][25] – Stores characters that make up the level.
PlayerType player – player object.
EnemyType enemy[25] – stores up to 25 enemy objects.
Int vertLevelSize = 25 – Stores vertical level size. Defaults to 25.
int horizLevelSize = 80 – Stores vertical level size. Defautls to 80.

playerType class:
int x, y – Stores player's location in x and y coordinates.

EnemyType class:
int x, y – Stores enemy's location in x and y coordinates.

Main function:
Main function: userInput – stores user's menu choice

gameLoop function:
levelType level – level object holding level/player/enemy data
char userInput – stores the user's input for which direction they want to move
bool toContinue – Set to 0 to break out of the game loop. Will be set to 0 upon a returned value of 1 or 2 from level.iterateMap.