Game State Management Plan:

Game States:
Main Menu: Allows the user to select other stages

About: Remember to include your own details here.

Help: List a summary of commands – hard coded text

Select Adventure: Use a hard-coded list and the title of your test game

Gameplay: A test stage which only accepts “quit” and “hiscore” commands

New High score: Allows user to enter their name, doesn’t save details yet.

View Hall of Fame: Shows a list of name/score. Hard coded text.

Main Menu has 5 options:

1. Select Adventure and Play
2. Hall of Fame
3. Help
4. About
5. Quit

About contains my name and a way to return to the main menu.

Help contains summary of two commands and a way to return to the main menu:

- Quit
- Hiscore

Select Adventure contains any potential adventure files (currently just a hard coded name), and a way to return to the main menu.

Selecting the adventure takes you to the gameplay (one way).

The gameplay contains a way to finish the game and quit to main menu.

When the gameplay is finished, the user is taken to the High Score.

High Score contains a place for user to enter their name, then the user gets taken to the hall of fame.

Hall of Fame contains 10 names/score hardcoded, and a way to return to the main menu.

This layout is very similar to the layout given in the lecture.

Classes:
Abstract class State with virtual methods, update, and render.

Update performs the stack operations.

Render performs the console operations (what the user sees).

Contains GameManager object _manager initialized to nullptr

Game Manager:

Contains stack of pointers to state objects.

Method that returns bool, checks if the stack is empty.

Method that returns State pointer, retrieves top of stack.

Method that takes a State pointer, pushes the state pointer to the stack.

Method that removes State pointer from top of stack.

Destructor, continues to pop states while stack is not empty.

Class Welcome:
update():

- Pop state
- Push Main menu state

Render:

- Dotted lines for games start

Constructor:

- Pass gamemanager to be initialized.

Class MainMenu:

Update():

- If string entry is one of (About, Help, Select Adventure, View Hall of Fame, or Quit) then:
  - Pop state
  - Push corresponding state

Render():

- Give 5 options. (About, Help, Select Adventure, View Hall of Fame, Quit)

Constructor:

- Pass Gamemanager to be initialized

Class About:

Update():

- If string entry is MainMenu
    - Pop state

Render():

- Print details about me to console (Name, Swin Number, etc)

Constructor:

- Pass GameManager to be initialized

Class Help:

Update():

- If string entry is MainMenu
    - Pop state

Render():

- Print list of commands and explanations:
    - Quit
    - HiScore

Constructor:
Pass GameManager to be initialized

Class HallOfFame:

Update():

- If string entry is MainMenu
    - Pop states until top state is MainMenu (to account for HoF being accessed through gameplay)

Render():

- print list of 10 players and their high scores

Constructor:

- Pass GameManager to be initialized.

Class SelectAdventure:

Update():

- If string entry is Game (will eventually be one of multiple games)
    - Push Gameplay state
- If string entry is MainMenu
    - Pop state

Render():

- List of games to select

Constructor:

Pass GameManager to be initialized

Class GamePlay:

Update():

- If string entry is Quit
    - Pop state twice to return to MainMenu
- If string entry is Finish
    - Push high score state

Render():

- Print two options
    - Quit
    - Finish

Constructor:
Pass GameManager to be initialized

Class HighScore:

Update():

- Accept user input
    - Push HallOfFame

Render():

- Prompt user for name

Constructor:

Pass GameManager to be initialized.