8.1: - application is a text based game

- Help, quit, go “north, east, south, or west”
  Help gives info on what commands can be used
  Quit closed the game
  go “chosen direction” goes through a door in the chosen direction
- 5 rooms in this scenario
-

8.2: Parser - interprets user input to see if its one of the accepted inputs or commands

Game - this class initalizes the room class and the parser class to create the world and commands for the world

Room - creates all the rooms and the connections between them using cardinal directions or null i none exist

CommandWords - holds all known inputs or commands and used to see if the user inputs a valid command

Command - has information on what the user inputs, it looks for two strings and then check with commandword to see if the user inputs a valid command and returns null if unknown

8.3:

8.4:

8.5:

```
    else {
            currentRoom = nextRoom;
            printLocationInfo();
        }
    }
```

Now the else statement in the goRoom() method moves rooms and calls out new printLocationInfo method

8.6: Changes implemented

8.7:

```
    public String getExitString()
    {
        String exits = "Exits: ";
        if(northExit != null) {
            exits += "North";
        }
        if(eastExit != null) {
            exits += "East";
        }
        if(southExit != null) {
            exits += "South";
        }
        if(westExit != null) {
            exits += "West";
        }
        return exits;
        }
```

Also a change to the printLocationInfo() method

System.out.println(currentRoom.getExitString());

This allows us to use our new method rather than using similar if statements

8.8:

8.9:

8.10:

8.11:

8.12:

8.13:

8.14: look command added by adding “look” to the array of accepted commands, a look method and a command to reference it

```
    private static final String[] validCommands = {
        "go", "quit", "help", "look"
    };



    private void look()
    {
        System.out.println(currentRoom.getDescription());
    }

        else if (commandWord.equals("look")) {
            look();
        }
```

8.15: //same thing as above

```
    private static final String[] validCommands = {
        "go", "quit", "help", "look", “drink”
    };

    Private void drink()
    {
    System.out.println("Thats some high quality H20!")
    }
```

```
else if (commandWord.equals("drink")) {
        drink();
     }

//And finally a way to call our command in game
```

8.16: code implemented
8.17: we no longer need to alter the game class to add commands because they are in the array contain validCommands and will be updated to the game class if they are added to the array in the commandWords class
8.18:
8.19:
8.20:
8.21:
8.22:
8.23
8.24
8.25
8.26
8.27
8.28
8.29
8.30