Name: Yunkai Liu
ID:29616425

First, I change my help menu a little bit to make it more comprehensible and follow the assignment brief.

HELP

```
**********************************************************************************
************************************
  Movement:
**********************************************************************************
************************************

To see the crime scene layout : MAP
To move around the crime scene : GO roomName (for example : GO garden)
To leave the case at any time : QUIT

**********************************************************************************
************************************
  Rooms and Items:
**********************************************************************************
************************************

To search the current room(find the items and doubtful point(murder scene?) in
this room) : SEARCH or SEARCH room
To examine an item in the current room or in the inventory : Examine itemName
(for example : Examine knife)
To take an item from the current room : TAKE itemName (for example : TAKE
knife)
To drop an item in the current room : DROP itemName (for exampke : DROP knife)
To see what you are currently carrying(inventory) : I

**********************************************************************************
************************************
  Suspects:
**********************************************************************************
************************************

To question a suspect(get his alibi information) : QUESTION suspectFirstName
(for example : QUESTION Mike)
To accuse a suspect of murder (You have to GATHER first)  : ACCUSE
suspectFirstName (for example : ACCUSE Mike)
```

```
To gather all the suspects in the current room : GATHER
To review questioned suspects notes(review all the conversations between player
and suspects) : REVIEW

********************************************************************************
**********************************
  General:
********************************************************************************
**********************************
To review this help menu : HELP
```

Development Outline

In this section, I'll further explain the points I made in assignment01 in details and additional features I include. I put the development outline for my assignment01 on the bottom of the page.

Class design:

1.Item class:

This class will contain three attributes which are name, description and position and behaviors including corresponding constructor, destructor, mutators and accessors.

Private:
string name;
string position;
string description;
Public:
Item();
Item(string initName, string initDescription);
~Item();
string getName(); //I don't have mutator for name and description because I think once
string getDescription(); //items are created, their names and descriptions won't change.
void setPosition(string newPosition);
string getPosition();

2.Player class:

This class will contain three attributes which are name, inventory and position and behaviors including corresponding constructor, destructor, mutators, accessors, drop, pick functions to add/remove items from the inventory and a printInventory function to display items in the inventory.

Private:

string name;
string position;
vector<Item> inventory;
Public:
Player();
~Player();
void setName(string newName);
string getName();
void setPosition(string newPosition);
string getPosition();
vector<Item> getInventory();
bool pick(Item i);
bool drop(Item i);
void printInventory();

3.Room class:

This class will contain two attributes which are name and description and behaviors including corresponding constructor, destructor, mutators, and accessors.

Public:
string name;
string description;
Private:
Room();
Room(string initName, string initDescription);
~Room();
string getName(); //Once room is created, its name won't change anymore.
string getDescription();
void setDescription();

4.Suspect class:

This class will contain five attributes which are name, description, location, statu, and alibi and behaviors including corresponding constructor, destructor, mutators, and accessors.

Public:
string name;
string description;
string location;
string statu;
string alibi;
Private:
Suspect();
Suspect(string initName, string initDescription, string initStatu);
~Suspect();
string getName(); //I do not need mutator since there is no need to change their values
string getDescription();

```
void setLocation(string newLocation);
string getLocation();
void setStatu(string newStatu);
string getStatu();
void setAlibi(string newAlibi);
string getAlibi();
```

Additional Feature:

1.All the player will randomly saunter around the crime scene.

i. First, I create an int variable turnCounter to hold the value of the number of turns.

ii. Then, every time the program goes through the loop, turnCounter will increment by one.

iii. Next, every six turns, people will begin to saunter around the scene(reset their position)

```
if(turnCounter % 6 == 0) {
    for(go through all the suspects){
        //get two random numbers first
        int randI = rand() % 3;       // get a value from 0 to 2;
        int randJ = rand() % 4;       // get a value from 0 to 3;
        //because the rooms are stored in a two dimension array, roomInformation[3][4], by
        //using roomInformation[randI][randJ], a random position can be created.
        suspect.setPosition(roomInformation[randI][randJ]);
    }
}
```

2.After player question murder, the murder will kill player in 15 turns

i. First, I create a bool variable questionMurder. If player question murder, this variable will be set to true

ii. Next, I create an int variable turnToBeKilled. If questionMurder is true, turnToBeKilled will begin to increment by one every time the program goes through the loop.

iii.Finally, if turnToBeKilled is greater than 15, a messenger will be shown to player and player will lose the game.

```
If(questionMurder) {
    turnToBeKilled++;
    if(turnToBeKilled > 15){
        player lose;
    }
}
```

Supplementary information for development outline

Processing player input:

For assignment 02, there are several new instructions, SEARCH, EXAMINE itemName, QUESTION suspectName, ACCUSE suspectName, GET itemName, DROP itemName, REVIEW, I, GATHER.

1. Guide the player to enter some instructions using specific format. (refer to the help menu above VERB + NOUN)

2. Read in the instructions and take corresponding actions

I. Use find to get the position of space and use substr to separate VERB and NOUN. If There is no space, just take the instruction as a VERB directly

II. Use selection structure to find the corresponding function according to that VERB.

III. If there is a NOUN, use NOUN as parameter to determine exact actions that need to be done.

SEARCH: i. There is no space in this instruction so SEARCH will be considered as a VERB.

ii. This function will find the current position of player by using player object's getPosition() function.

iii. The function will use a for loop to go through the array storing all the item objects to check if the positions of any items are in the same room as player.

iv. If item object's getPosition() == player object's getPosition, inform player that there is a item.

v. The function will check if the current position of player is same as the murder scene. If they are the same, display "there is blood on the wall" to inform player.

EXAMINE itemName:

i. There is a space so EXAMINE will be considered as a VERB and itemName will be assigned to NOUN.

ii. The function will use a for loop to go through the array storing all the item objects to check if the itemName is the name of existing items(Item.getName()). If not, display an error messenger. If it is, go step iii.

iii. Use a for loop to go through the array storing all the item objects to check if the current position of player is same as itemName's position or if the itemName is in the player's inventory. If not, print an error messenger, if it is, display the description of itemName.

QUESTION suspectName:

i. There is a space so QUESTION will be considered as a VERB and suspectName will be assigned to NOUN.

ii. The function will use a for loop to go through the array storing all the suspect objects to check if the suspectName is the name of existing suspects. If not, display an error messenger. If it is, go step iii

iii. Use a for loop to go through the array storing all the suspect objects to check if the current position of player is same as suspectName's position. If not, print an error messenger, if it is, display the alibi of suspectName.

ACCUSE suspectName:

i. There is a space so ACCUSE will be considered as a VERB and suspectName will be assigned to NOUN.

ii. The function will use a for loop to go through the array storing all the suspect objects to check if the suspectName is the name of existing suspects. If not, display an error messenger. If it is, go step iii

iii. Use a for loop to go through the array storing all the suspect objects to check if all the suspects are in the same room as player. If not, display a messenger to remind player use GATHER instruction first. If it is, go step iv.

iv. Check if the current position of player is the same as murder scene. If not, player lose the game, if it is, go step v.

v. Check if the item in player's inventory is the weapon. If player do not have the correct item or player's inventory is empty, player lose. If player have, go step vi.

vi. Check if the suspectName is the same as the murder. If it is, player win, if not, player lose.

TAKE itemName:

i. There is a space so TAKE will be considered as a VERB and itemName will be assigned to NOUN.

ii. The function will use a for loop to go through the array storing all the item objects to check if the itemName is the name of existing items(Item.getName()). If not, display an error messenger. If it is, go step iii.

iii. Use a for loop to go through the array storing all the item objects to check if the current position of player is same as itemName's position. If not, print an error messenger, if it is, go step iv.

iv. Check if the inventory of player has space. If it has, add this item object to the inventory, if not, inform player to use DROP first.

DROP itemName:

i. There is a space so DROP will be considered as a VERB and itemName will beassigned to NOUN.

ii. The function will use a for loop to go through the array storing all the item objects to check if the itemName is the name of existing items(Item.getName()). If not, display an error messenger. If it is, go step iii.

Iii. Check if player's inventory contain itemName object. If not, display an error messenger, if it is, use pop_back() to remove it from the inventory vector.

REVIEW:

i. There is no space so REVIEW will be considered as a VERB.

ii. There is a string variable called reviewNote. When QUESTION is successful invoked, the result of QUESTION instruction will be added to reviewNote.

iii. Check if the reviewNote is an empty string. If it is, inform player that they have not talked to anyone, if not, display reviewNote to the screen.

I:

i. There is no space so I will be considered as a VERB.

ii. Check if player's inventory is empty, if it is, show player "The inventory is empty", if not, print name and description of the item objects in the inventory.

GATHER:

i. There is no space so GATHER will be considered as a VERB.

ii. Use a for loop to go through the array containing all suspect objects to set their position to the current location of player.