Project Plan for Master Mind :: Escape the Room

Jiaying Wu

How to play Master Mind :: Escape the Room

The four text paragraph display below will separate into two parts:

1. Title start with "~" and end with "~", display when calling the function **`displayTitle()`**.
2. The rest is the body, it will store in different text file.

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                        MasterMind :: Escape the Room
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

"You have been locked in an empty dark room, running out of food and water.
In front of you, there is a gate with a giant lock. The times you can enter
the numbers into the lock is limited! Once you ran out of the times, the
gate will lock forever! Hurry up! Clock is ticking......"

If you enter all correct elements with correct position into the lock in the
limited times, the gate will open. That you can escape the room. Otherwise,
you will be lock in the room forever......

 [S] to start a new game.

 [C] to continue the saved game.

 [H] for help, to display the game rule.

 [A] to view the achievements.

 [E] to end the game.
```

The above text is the main page of the game, it will stored in this text file **`masterMindMainPage.txt`**. It will load and display at the beginning of the game for player to read.

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                        MasterMind :: Escape the Room
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Three levels of this game:

 [1] wood gate = easy, 4 elements out of 6 with 10 times to try.

 [2] Rock gate = tricky: 5 elements out of 8 with 12 times to try.
```

```
 [3] Iron gate = hard: 6 elements out of 10 with 14 times to try.

After you enter the elements into the lock, the lock will provide you some clues.

- One `#` will show up if you enter a correct element in correct position.

- One `*` will show up if you enter a correct element in wrong position.

- No symbol for a worng element

- The order of symbols to show up is random in each time.

The way to caculate the points for each game:

1. The points for wood gate start from 100, rock gate start from 180 and
   iron gate start from 280.

3. Each time will cost you 10 points in wood gate, 15 points in rock gate,
   20 points in iron gate.

4. 2 points for one `#`, 1 point for one `*`.

5. Maximum points for opening wood gate is 98, that is four `#` in 1 time.
   Maximum 175 points for rock gate and maximum 272 points for iron gate.

6. The points you get from each game will add up to your total points.
```

The above text is the game rule, it will stored in this text file **masterMindHelp.txt**. It will load and display when user press `[H]` for help.

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                      MasterMind :: Escape the Room
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Three steps to start the game:

1. Enter your name.

2. Select the gate you want to open:

 [1] Wood gate = easy: 4 elements out of 6 with 10 times to try.

 [2] Rock gate = tricky: 5 elements out of 8 with 12 times to try.

 [3] Iron gate = hard: 6 elements out of 10 with 14 times to try.

3. Select the type of element for the lock:

 [1] Number
```

```
 [2] Letter

 [3] Symbol

 [4] Word

Then you will face to the gate you have chosen.
```

The above text is the setup for the game, it will stored in this text file **masterMindStart.txt**. It will load and display when user press [S] to start a new game.

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                        MasterMind :: Escape the Room
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
--------------------------------------------------------------------------------
Player name:
Total points:
Highest points for wood gate is: 0 / 98
Highest points for rock gate is: 0 / 175
Highest points for iron gate is: 0 / 272
--------------------------------------------------------------------------------
```

This is the initial version of player's achievement. It will stored all players' achievement, identify by player name. It will store in this text file **masterMindAchievement.txt**, it will load and display when player press [A]

Development outline of Master Mind :: Escape the Room

Outline the functionality of all game classes

After reading the assignment brief, I decided to use 3 classes for this game.

1. **Player class**

Player class is

2. **Board class**

Board class is

3. **Application file**

Application file include

What we need to create

Six sections for game setup

Section 1: Variables and functions for setup

Variables:

1. *string* `previousOption` to stored player's previous option.
2. *string* `currentOption` to stored player's current option.
3. *string* `playerName` to store player's name.
4. *int* `difficultyLevel` to store the chosen difficulty level of the game.

Functions:

1. *void* `displayTitle()` to display the title of the game.
2. *void* `readTextFile`(*string* `fileName`) to read the given text file and display into screen.
3. *string* `askForString`(*string* `question`) to ask user enter the option, return a *string* value. *string* `question` is a local variable in function `askForString()`.
4. *int* `askForInt`(*string* `question`) to ask user enter the option, return a *int* value. *string* `question` is a local variable in function `askForInt()`.

Section 2: Main page

The main page display everytime we start the game, or press `[M]` during the game.

1. Clear the screen.
2. Call the function `displayTitle()` to display the title.
3. Load and display the text file `masterMindMainPage.txt` using function`readTextFile()`.
4. Replace `previousOption` = "m".
5. Call the function `askForString()` to ask player enter their option, and check the input. If not in {"s", "c", "h", "a", "e"}, return error message. Call the function again until player enter the correct option.
6. Store the string return from function `askForString()` as `currentOption`.
7. Then go the page `currentOption` match. Eg, "h" for help page.

Since the screen will pause until the player enter the option, it provide time for player to read the information.

What we need:

Variables:

1. *string* `previousOption`
2. *string* `currentOption`

Functions:

1. *void* `displayTitle()` to display the title.
2. *void* `readTextFile`(*string* `fileName`) to read `masterMindMainPage.txt` then display main page.
3. *string* `askForString`(*string* `question`) to ask a string from player.

Section 3: Help page

The help page display when player press `[H]`.

1. Clear the screen.
2. Call the function `displayTitle()` to display the title.
3. Load and display the text file `masterMindHelp.txt` using function `readTextFile()`.
4. Replace the `previousOption` = `currentOption`.
5. Replace `currentOption` = "h".
6. Include a "Press any key to continue", pause the screen to allow player to read the information.
7. Then go back to the page `previousOption` match. Eg, "m" for main page...

What we need:

Variables:

1. *string* `currentOption`
2. *string* `previousOption`

Functions:

1. *void* `displayTitle()` to display the title.
2. *void* `readTextFile`(*string* `fileName`) to read `masterMindHelp.txt` then display Help page.

Section 4: Achievement page

The achievement page display when player press `[A]`.

1. Clear the screen.
2. Call the function `displayTitle()` to display the title.
3. Load and display the text file `masterMindAchievement.txt` using function `readTextFile()`.
4. Replace the `previousOption` = `currentOption`.
5. Replace `currentOption` = "a".
6. Include a "Press any key to continue", pause the screen to allow player to read the information.
7. Then go back to the page `previousOption` match. Eg, "m" for main page...

What we need:

Variables:

1. *string* `currentOption`
2. *string* `previousOption`

Functions:

1. *void* `displayTitle()` to display the title.
2. *void* `readTextFile`(*string* `fileName`) to read `masterMindAchievement.txt` then display achievement page.

Section 5: Start game page

Before the game start, player need to finish some setup.

1. Clear the screen.
2. Replace the `previousOption` = `currentOption`.
3. Replace `currentOption` = "s".
4. Call the function `displayTitle()` to display the title.
5. Call the function `askForString()` to ask player's name, store the return *string* as `playerName`.
6. Call the function `askForNumber()` to ask player to select the gate to open, that is the game difficulty level.
7. check the input, if it within {1, 2, 3}. Store the return *int* as `difficultyLevel`. Otherwise return error message, then call the function `askForNumber()` until player enter the correct integer.
8. Call the function `askForNumber()` to ask player select the type of element.
9. check the input, if it within {1, 2, 3, 4}. Store the return *int* as `elementType`. Otherwise return error message, then call the function `askForNumber()` until player enter the correct integer.
10. After player finish enter their name and game option, include a "Press any key to continue" in the end.

What we need:

Variables:

1. *string* `currentOption`
2. *string* `previousOption`
3. *string* `playerName`
4. *int* `difficultyLevel`

Functions:

1. *void* `displayTitle()` to display the title of the game.
2. *string* `askForString`(*string* `question`) to ask player's name, *string* **question** is the local variable.
3. *int* `askForNumber`(*string* `question`) to display the question and ask user's choice, *string* **question** is the local variable.

Section 6: Game page

After player press [S] and enter their name, select game difficulty level and element type, press a key to continue at last. Then shift to the page display below.

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                        MasterMind :: Escape the Room
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

 .~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~.
 | .---. .---. .---. .---.              |
 | |   | |   | |   | |   |  Hidden Code  |
 | '---' '---' '---' '---'              |
 '~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~'
```

1. Clear the screen.
2. Call the function `displayTitle()` to display the title.
3. Create the following local variables:

- An *int* `possibleElement` = 3 + 2* `difficultyLevel` to indicate the number of all possible elements, possible value is {6, 8, 10}.
- Ana *int* `codeElement` = 3 + `difficultyLevel` to indicate the number of elements in one row, that is the numbers of element in the secret code. Possible value is {4, 5, 6}.
- An *int* `codeRow` = 8 + 2 * `difficultyLevel` to indicate the number of rows, that is the times players can try. Possible value is {10, 12, 14}.
- Create and initialize an *int* `gameRound` = 0. That indicate the current game round.

4. Create a function to *string* `gameTableTitle`(*int* `codeElement`, *int* `elementType`) to generate the table title above. `elementType` to determine how many empty space inside the cell.
5. Store the return *string* as `gameTable` and display `gameTable`.
6. Generate the a string vector `secretCode` using function `generateCode()`.

What we need:

Variables:

1. *int* `difficultyLevel`
2. *int* `possibleElement`

3. *int* `codeElement`
4. *int* `codeRow`
5. *int* `elementType`
6. *int* `gameRound`
7. *string* `gameTable`
8. *vector* `secretCode`

Functions:

1. *void* `displayTitle()` to display the title of the game.
2. *string* `gameTableTitle`(*int* `codeElement`, *int* `codeRow`, *int* `elementType`)
3. *vector* `generateCode`(*int* `codeElement`, *int* `codeRow`, *int* `elementType`)

The player's turn

1. After the press a key to continue, clear the screen.
2. display the title of the game.
3.

Processing player input

Providing feedback to player

The end game conditions

Additional features included

UML class diagrams