CS202

Assignment 4: Mastermind

Description

Well it seems to be that part of the semester when life gets tough and you want to have a small escape. The obvious choice is for you and your friends to play some games together, as you're deciding your friend decides that league would be a good idea, or minecraft, or even fortnite, but then you realize that you want to play a game that actual makes sense! :P So you decide to implement and play mastermind.

Mastermind is a two player game where one player places a set of pegs with various colors to constructor his/her solution. Typically the width (the amount of pegs in the solution is 4 but we can make any length since we aren't using a physical board for this assignment). The other player tries to guess that solution in a series of rounds. The user that is trying to guess the solution places his/her set of pegs of various colors on the board. Then the player that has the solution gives clues to the other player as to how close they are. The player holding the solution puts down a set of gold pegs and silver pegs.

A gold peg denotes that the guessing player placed a peg with a color that matches a color on the solution board at the exact same location (or column), and a silver peg denotes the guessing player matched a color on the solution board but not at the same location (column), these give clues to the guessing player as to how close they are to the solution. They win once they match the solution and lose when they run out of attempts.

You can have duplicate peg colors in the solution and in the guess, but you do not want to duplicate the matches. But you need to treat each peg as distinct, so you want to match one user guess peg to a peg on the solution. The wikipedia link for more info on the game can be found using the link below

https://en.wikipedia.org/wiki/Mastermind_(board_game)

You will need to implement the following class to simulate this game

```
class mastermind
{
public:
  mastermind(int);
  mastermind(vector<string>);
  int move(vector<string>, int&, int&);
  int getMoveIndex() const;
  int getNumberOfColumns() const;
private:
```

```
    int maxMoves;
    vector<string> solution;
    int moves;
};
```

Each member contains/performs the following

- `int maxMoves` - contains the max number of moves that can be done during the duration of the game
- `vector<string> solution` - contains the colors of the pegs for the solution of the game
- `int moves` - contains the amount of moves done so far by the user
- `mastermind::mastermind(int size)` - constructor that sets the size of `solution` vector and assigns each element to contain `"red"`, sets `maxMoves` to 5 and `moves` to 0. This constructor might not be called but doesn't hurt to practice more with constructors etc.
- `mastermind::mastermind(vector<string> initial)` - constructor that sets the `solution` vector with the content of `initial` vector (make sure to store them in a case insensitive fashion), also set `maxMoves` to 5 and `moves` to 0
- `int mastermind::move(vector<string> playerMove, int& gold, int& silver)` - this function simulates one move/round of the mastermind game. The `playerMove` vector consists of the user's guess which is a set of pegs with their colors, and the function needs to determine how many pegs's colors are in the same respective column as the `solution` vector (update `gold` each time), and the colors from `playerMove` that matches with `solution` but not in the same respective position (update `silver` each time), then update the moves counter. The function returns 4 possible integers
  - If it returns 0, the game is not over
  - If it returns 1, the user won (i.e. the amount of gold matches the width of the board)
  - If it returns -1, the user lost (ran out of attempts)
  - If it returns 2, then the amount of pegs in the `playerMove` does not match the amount in`solution`, so not a valid move, do not update the moves counter in this case (this scenario most likely will not come up, but we're trying to handle this just in case)
- `int mastermind::getMoveIndex() const` - returns the moves amount, remember moves is set to 0 initial which is actual move 1...
- `int mastermind::getNumberOfColumns() const` - returns the width of the current game, i.e. the size of the `solution` vector

Contents of main

In main, you will have two pointers of `mastermind` type, so you could have

```
mastermind * currentGame = NULL; //nullptr is fine as well
mastermind * savedGame = NULL; //nullptr is fine as well
```

Where `currentGame` points to a `mastermind` object of the current game in play and `savedGame` points to a `mastermind` object of a saved game. A `NULL` or `nullptr` denotes that either there is no current game or saved game respectively. The contents of `int main()` that has the menu is given to you but you will need to implement the following functions

- `void setupGame(mastermind*& currentGame, mastermind*& savedGame)` - this function creates a new game by performing the following tasks
  1. Prompt the user for peg colors terminated with "0" (they must have at least one peg color)
  2. If no currentGame in progress, go to step 5

3. If there is a current game in progress, ask the user if they wish to save the game, if they wish to save the game call the saveGame function if the saveGame function did not actually save the game then exit the setup function, otherwise go to step 5
4. If there is a current game in progress, and the user does not want to save the game, deallocate the currentGame and go to step 5
5. Allocate a new game to currentGame pointer by passing the vector from step 1 to the constructor

- `void saveGame(mastermind*& currentGame, mastermind*& savedGame)` - this function saves a current game by performing the following tasks
  1. If there is no saved game, set saveGame to currentGame and assign currentGame to NULL and exit function
  2. If there is a saved game, ask the user if they still want to save over the savedGame, if they want to save over, deallocate savedGame and set it to currentGame, then set currentGame to NULL
  3. If the user does not want to save the game then do nothing and exit from the function
- `void loadGame(mastermind*& currentGame, mastermind*& savedGame)` - this function saves a current game by performing the following tasks
  1. If there is no saved game, exit out of the function
  2. If there is a current game, ask the user if they wish to destroy this game, if they do, destroy the game and go to step 4
  3. If they do not wish to destroy the current game, then exit the function
  4. Set current game with the game in saved game and set saved game to NULL
- `bool yesOrNo(string msg)` - **This function is written for you**, all this function does prompt the user by outputting `msg` and re-prompts if the user does not enter Y/y or N/n, once the user enters Y/y or N/n it returns `true` if the user entered Y/y and returns `false` if they entered N/n. You would call this function in the following way

```
if (yesOrNo("(Y/N): "))
{
  //User entered Y/y
}
else
{
  //User entered N/n
}
```

  So it prompts to the screen `"(Y/N): "` and will keep prompting them that until they enter a valid character

- `void playGame(mastermind*& currentGame, mastermind*& savedGame)` - this function saves a current game by performing the following tasks
  1. If there is no current game, exit out of the function
  2. Output the turn number and output the amount of peg colors the user must enter
  3. If they enter "P" or "p" ask the user if they wish to save their progress, if they do then call save game and exit the function, if they do not wish to save the game then just exit the function without modifying currentGame or savedGame
  4. If they entered their set of colors for the pegs, call currentGame's move function
  5. Output the silver and gold pegs

6. If the game is over (win/lose) output the appropriate function, the destroy the currentGame, set the pointer to NULL as well and escape the game
7. If the game is not over, go back to step 2

Specifications

- Comment your code and your functions
- Do not add extra class members or remove class members and do not modify the member functions of the class
- No global variables (global constants are ok)
- Make sure your program is memory leak free

Sample Run

```
$ make
g++    -c -o main.o main.cpp
g++    -c -o mastermind.o mastermind.cpp
g++ main.o mastermind.o -o game

$ ./game

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: C

No current game found

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: l


There is no game to load. Unable to complete the task.

No current game found

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: X
```

```
Nope. 0/10. Do not approve of this input

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: p

Enter the peg colors followed by 0
0
Sorry! You need at least 1 peg to play the game
0
Sorry! You need at least 1 peg to play the game
red blue BlUe GrEen ORANGE 0

Turn 1: Enter 4 colors or (P)ause:
red green orange BLUE
Gold pegs: 1
Silver pegs: 3

Turn 2: Enter 4 colors or (P)ause:
p
Ok, we all need a break sometimes
Would you like to save the game?
(Y/N): h
(Y/N): i
(Y/N): n
Ok that's fine, you can always resume

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: c

Turn 2: Enter 4 colors or (P)ause:
red red red red
Gold pegs: 1
Silver pegs: 0

Turn 3: Enter 4 colors or (P)ause:
p
Ok, we all need a break sometimes
Would you like to save the game?
(Y/N): y
Ok saving game

MAIN MENU
(P)lay a new game
```

```
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: C

No current game found

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: 1


Turn 3: Enter 4 colors or (P)ause:
p
Ok, we all need a break sometimes
Would you like to save the game?
(Y/N): n
Ok that's fine, you can always resume

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: p

Enter the peg colors followed by 0
green yellow blue
0

You already have a game in progress
Would you like to save the game?


(Y/N): y
Saving current game...

Turn 1: Enter 3 colors or (P)ause:
red red red
Gold pegs: 0
Silver pegs: 0

Turn 2: Enter 3 colors or (P)ause:
p
Ok, we all need a break sometimes
Would you like to save the game?
(Y/N): n
Ok that's fine, you can always resume
```

```
MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: l


This action will destroy current game.
This action cannot be undone.
Do you wish to continue?

(Y/N): y
Ok loading game

Turn 3: Enter 4 colors or (P)ause:
p
Ok, we all need a break sometimes
Would you like to save the game?
(Y/N): n
Ok that's fine, you can always resume

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: l


There is no game to load. Unable to complete the task.

Turn 3: Enter 4 colors or (P)ause:
p
Ok, we all need a break sometimes
Would you like to save the game?
(Y/N): n
Ok that's fine, you can always resume

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: p

Enter the peg colors followed by 0
green orange yellow 0

You already have a game in progress
```

```
Would you like to save the game?


(Y/N): y
Saving current game...

Turn 1: Enter 3 colors or (P)ause:
red red red
Gold pegs: 0
Silver pegs: 0

Turn 2: Enter 3 colors or (P)ause:
p
Ok, we all need a break sometimes
Would you like to save the game?
(Y/N): y
Ok saving game

There is a saved game
This action will destroy/overwrite previosuly saved game. Is this ok?
(Y/N): n

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: p

Enter the peg colors followed by 0
red blue orange pink purple 0

You already have a game in progress
Would you like to save the game?


(Y/N): y
Saving current game...

There is a saved game
This action will destroy/overwrite previosuly saved game. Is this ok?
(Y/N): n

Turn 2: Enter 3 colors or (P)ause:
red yellow blue
Gold pegs: 0
Silver pegs: 1

Turn 3: Enter 3 colors or (P)ause:
yellow blue green
Gold pegs: 0
Silver pegs: 2
```

```
Turn 4: Enter 3 colors or (P)ause:
green blue yellow
Gold pegs: 2
Silver pegs: 0

Turn 5: Enter 3 colors or (P)ause:
green orange yellow
Gold pegs: 3
Silver pegs: 0

You won! :)
Destroying game

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: c

No current game found

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: L


Turn 3: Enter 4 colors or (P)ause:
green orange blue
red
Gold pegs: 0
Silver pegs: 4

Turn 4: Enter 4 colors or (P)ause:
orange green red blue
Gold pegs: 0
Silver pegs: 4

Turn 5: Enter 4 colors or (P)ause:
red blue green orange
Gold pegs: 4
Silver pegs: 0

You won! :)
Destroying game

MAIN MENU
(P)lay a new game
(L)oad an existing game
```

```
(C)ontinue current game
(Q)uit the program

Choose an option: P

Enter the peg colors followed by 0
red blue yellow grey 0

Turn 1: Enter 4 colors or (P)ause:
blue red orange pink
Gold pegs: 0
Silver pegs: 2

Turn 2: Enter 4 colors or (P)ause:
red red red red
Gold pegs: 1
Silver pegs: 0

Turn 3: Enter 4 colors or (P)ause:
red blue red blue
Gold pegs: 2
Silver pegs: 0

Turn 4: Enter 4 colors or (P)ause:
blue blue blue blue
Gold pegs: 1
Silver pegs: 0

Turn 5: Enter 4 colors or (P)ause:
green yellow blue red
Gold pegs: 0
Silver pegs: 3

You lost! :(
Destroying game

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: p

Enter the peg colors followed by 0
red 0

Turn 1: Enter 1 colors or (P)ause:
p
Ok, we all need a break sometimes
Would you like to save the game?
(Y/N): y
Ok saving game
```

```
MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: p

Enter the peg colors followed by 0
blue 0

Turn 1: Enter 1 colors or (P)ause:
p
Ok, we all need a break sometimes
Would you like to save the game?
(Y/N): n
Ok that's fine, you can always resume

MAIN MENU
(P)lay a new game
(L)oad an existing game
(C)ontinue current game
(Q)uit the program

Choose an option: Q

Ok stay safe, I hope this took your mind off of everything
```

Submission

Compress mastermind.h, mastermind.cpp, main.cpp, and makefile into a zip file and upload to the canvas site by the deadline

References

- Link to the top image can be found at
  *https : //cdn.quizly.co/wp − content/uploads/*2015/12/29213834/*Mastermind.png*
- A supplemental video for the master mind class can be found at https://youtu.be/bJx4JEPKxb0
- A supplemental video for the main file can be found at https://youtu.be/ITWGzSLXfX8
- A supplemental video for the sample output can be found at https://youtu.be/B7iijXD2VNs