REPORT: project Caro/Gomoku game

GROUP 7

APCS 2

|  |  |  |  |
| --- | --- | --- | --- |
| STT | MSSV | Full name | Percentage (%) of contribution |
| 1 | 18125073 | Huynh Bao Di | 40% |
| 2 | 18125105 | Do Minh Nhut | 30% |
| 3 | 18125127 | Do Le Duan | 30% |

1. **Functions of the project (Gomoku game):**

* Loading screen
* Splash screen
* About-us screen
* Drawing the board game
* PvP mode
* Change player’s icons in PvP mode (2 pairs of icon)
* Save/load game in PvP mode (4 slots of saving/loading games)
* Two rules in PvP mode (Normal rule and two-ways-blocked rule)
* PvC mode
* Save/load game in PvC mode (4 slots of saving/loading games)
* Player statistic in PvC mode (saving players name and showing the number of played/won games)

1. Contribution of 1st student: **Huynh Bao Di – 18125073**

* PvP mode:

- void gotoxy(int x, int y): i used this function to help players move their pointer on the game board to tick.

* void draw(int height, int width, char matrix[60][60]): i used two “For-loop” and char matrix[60][60] to draw the game board. Each elements of the matrix[60][60] are ‘\_’, we “cout” each elements with “|” next to them to constitute to board.

* int X\_Win(int check), int O\_Win(int check), void icon1\_Win(), void icon2\_Win(): these four functions will load the suitable special word-arts that I repaired on file and shows it on the screen when one player win the game.
* void changeicon(int tick, char &player): this function will change the player icon turn-by-turn .
* rule2dau(x, y, tren, duoi, ngangtrai, ngangphai, cheotrentrai, cheoduoitrai, cheotrenphai, cheoduoiphai, quandichdoc, quandichngang, quandichcheothuan, quandichcheoxuoi, matrix, player): this function helps players to play the game in “Two-way-blocked rule”. It depends on the “int rule” that I will explain more carefully later. I declared 16 parameters just to check 8 directions 5 times around the previous tick of each players. If there is any of allies, a specific variable that belong to that direction will be increased, otherwise, a blocked-variable will be increased to block that direction from being checked in the next loop to save time. For example, the “for-loop” firstly checks the up-ward direction, if it finds an ally, the variable “int tren” will be increased 1 unit, otherwise, if it finds an empty blank, the variable “int checktren” will be increased instead to block the upward direction. There is also a special situation, if the “for loop”

finds an enemy, not only “int checktren” will be increased to block that direction but also the “int quandichtren” to count the number of enemies in vertical. The for loop will keep checking others direction and ignore the up-ward way. At the end, the function will check the sum of all the opposite direction, if the sum of allies is larger or equal to 4 (>=4) and the sum of enemies is smaller than two (<2), the player who uses that tick will win, otherwise, the game will go on.

* void gameplay\_(int q, int p, char matrix[60][60], int tick, char player, int rule, int turn): this is the main function of PvP mode. Firstly, I used “\_getch()” to catch signal from the keyboard and put it to the “char key”. I declared “int x” and “int y” as coordinates of the pointer (gotoxy(x, y)) and the “char matrix matrix[y][x]”. Then, I used “switch-case” to check the “char key”, if “key” is “W”, the “y” will be increased to let the gotoxy(x, y) move the pointer and also the matrix[y][x] up-ward. It also does the same to other direction. If “key” is “k”, the ticker belongs to current player will be “cout” on screen right at the coordinates of gotoxy(x, y) and the element of matrix[y][x] is also be changed. Then, the player icon will be changed to the other and the game goes on. After every single tick of each players, there is also a quite same system to function ” rule2dau” to check which player is a winner in “normal” mode. If “key” is “o”, four slots of saving/loading will be showed on the screen for player to choose. The save/load function will be explained later. The last situation is “key == ‘q’”, because all the actions above are done inside a “while-loop” with the condition “int out == 0”, when players press ‘q’, the “out” will be increased and breaks the “while-loop”, help player to quit playing and go back to the menu screen.
* Save games in PvP mode:
* void SaveGame1(int sa, int ve, char matrix[60][60], int tick, int rule): I used “ofstream” in the “fstream” library to store the size of the game board, the player’s icons, the game rule and also the whole game board in 4 separated “txt” files.

* void SaveTurn1(char player, int turn): I also used “ofstream” to store the player turn to “txt” file.
* Both of functions above save the current game to the 1st slot of saving, there are 3 more pairs of function like these to save the game to others.
* Load game in PvP mode:
* void LoadTurn1(char &player): This function help to load the icon of the last players when the game was saved from the “txt” file.
* void load1(char matrix[60][60], int &dai, int &rong): This function helps to load the whole board game which was saved and put it in the suitable element in matrix[60][60].

* void LoadGame1(int lo, int ad, int &tick, int &rule): This function helps to put the game board from “txt” file and prints it on the screen for player to see.
* Three functions above, just like “Saving”, load the game from the 1st slot of saving and there also a bunch of functions which have the same function of these for others.
* PvC mode:
* int TCdoc(int CurDong, int CurCot, char matrix[60][60], long MangDiemTanCong[7], long MangDiemPhongNgu[7]);
* int TCngang(int CurDong, int CurCot, char matrix[60][60], long MangDiemTanCong[7], long MangDiemPhongNgu[7]);
* int TCcheothuan(int CurDong, int CurCot, char matrix[60][60], long MangDiemTanCong[7], long MangDiemPhongNgu[7]);
* int TCcheonguoc(int CurDong, int CurCot, char matrix[60][60], long MangDiemTanCong[7], long MangDiemPhongNgu[7]);
* int PTdoc(int CurDong, int CurCot, char matrix[60][60], long MangDiemPhongNgu[7], long MangDiemTanCong[7]);
* int PTngang(int CurDong, int CurCot, char matrix[60][60], long MangDiemPhongNgu[7], long MangDiemTanCong[7]);
* int PTcheothuan(int CurDong, int CurCot, char matrix[60][60], long MangDiemPhongNgu[7], long MangDiemTanCong[7]);
* int PTcheonguoc(int CurDong, int CurCot, char matrix[60][60], long MangDiemPhongNgu[7], long MangDiemTanCong[7]);
* All of these function above have a same purpose: calculate the attack point and defense point from 8 directions around every single squares contained in the game board.
* void ComVsPlayer(int q, int p, int n, char matrix[60][60], PLAYER nguoichoi[5], char player): this is the main function of PvC mode. Generally, this function operates the same way as “void gameplay\_(int q, int p, char matrix[60][60], int tick, char player, int rule, int turn)” does. The biggest difference is when the player press “k” and tick the icon to the gameboard, the function will goes through all the point-calculated functions mentioned above and calculates the sum of all the attack and defense point. I declared an integer MAX, after comparing the sum of attack and defense point, the function will put the larger sum to the integer MAX. Finally, after checking all the squares from the game board, the function will tick on the square which relates to the integer MAX, or to say in other way, the squares which has the largest attack or defense point.
* void play\_er(char &player): i used “If-statement” to declare the “char player”, if player is currently the letter “X”, it will be changed to the letter “O” and so on.
* Save game in PvC mode:
* void SavePvC1(int sa, int ve, int n, char matrix[60][60]): this function, just like PvP mode, saves the size, the whole game board and also the information of the current player such as name, number of played game, number of won game (I will explain more carefully later). There are also three more function which help to save the game in others three slots of saving in PvC mode.
* Load game in PvC mode:
* void loadCom1(char matrix[60][60], int &dai, int &rong): this function put the game board which was saved from file and to the proper element of matrix[60][60] to maintain the game.
* void LoadPvC1(int lo, int ad, int &NumPlayer, PLAYER nguoichoi[5]): this function put the game board, the size and player’s name, which were saved on file, on the screen to continue the game.
* There also 3 more pairs of function for 3 more slots of saving.
* Option:
* Change player’s icon: I have mentioned this function above, the system will depends on the input of player to the integer “tick” to decide which pair of icons will be used.
* Change rules: having been mentioned, as well, the system will depends on the input to the integer “rule” to decide which rule will be used.
* But, unluckily, player can only change the rule and the icon on PvP mode.
* Player statistic:
* I used a stuct “PLAYER“ which contains player’s name, number of played game, number of won game. During the PvC mode procession, the system will store all the information of play to “txt” file. There are also 4 slots of player-information-saving, when player want to see information of a particular player, the system let them choose from the list of slots and show them information of the player which have been stored to that slot before.
* Splash screen, about-us screen and loading screen:
* I also helped Duan and Nhut to do these tasks of the project.
* Note:
* I shared the source code of all the tasks above and explained them to my team. I also convinced them to give opinions and feedback to my tasks during the procession.
* We tested the code toghether and I fixed all the bugs.
* I repaired the group report by combining all the member’s reports.

1. Contribution of 2nd student: **Do Le Duan – 18125127**

+ Function “**Loading**” allows user to see a loading bar on the screen which runs from 0% to 100% . We use Sleep(500) in library <window.h> to decide the interval of time when 0% becomes 10% , 500 means 0.5 second in this statement . Besides , we to gotoxy(45 + x, 13) to determine the place where the Loading bar will appear , TextColor to add color to word and make it look more attractive . All the things will be included in a loop “For” to make sure it will run continously and smoothly , we make Function “Loading” to make our game more interesting , first impression to the user is very important

+ The second thing is the function “**About Screen**” . The purpose of this function is to introduce about our first project to teacher from 3 members of our group and it contains 5 text from Text1 to Text5 . Because we want each word run step by step , so we use a loop “For” for each Text(Number) . In loop , we use statement Sleep(40) to determine the interval for the appearance of each word and we choose 0.04s and statement “gotoxy” to place the the first word of each Text . We use the same way from Text1 to Text5

+ Next , a word “ **GOMOKU** “ with color will appear gradually from high to low of the dark screen . It is the result of the “PS.txt” which is created outside and put in source code through ifstream myFile and use myFile.open(“PS.txt” , ios::in) to open it in program , we use a loop “While” with the condition (!myFile,eof()) to run all things in file smoothly . In the loop , we declare char splashscreen and import it by myFile.get(splashscreen) then print it on screen by cout << splashscreen , to close it we use myFile.close().

+ The next thing is that we use function “**Draw**” to sketch the board allowing us to play XO, we use two loop blendid together to define the graph, the loop for the width is run first and after it has done , the height will continue it work , matrix [0] [0] represents for the first small square on the left side , it will appear \_ gradually in horizon .

+ In function **play\_er**(), firstly , we declare char player and use structure if …. Else to allow the user to choose their role in this game ( X or O ) , if player 1 choose X then the other will be O and reverse it .

+ When the use choose X and X Win , we write the function **X\_Win** to annouce that X won that game . Like SpashScreen, by using library ”fstream” ,we create file Notepad with name “XWin.txt” and add it to this function through statement called “ifstream” . Moreover we use Xwin.open(“Xwin.txt”, ios::in) to open it in source file , then we use a loop “While” with the condition “!Xwin.eof()” to make sure all the character in file will be scanned and stop perfectly . In loop , we declare char X and use Xwin.get(X) to enter word gradually with color (TextColor(9)) and display it on the screen with statement “cout << X” . Outside the loop , we use XWin.close() to close the file

+ O\_Win is almost the same with X\_Win

1. Contribution of 3rd student: **Do Minh Nhut – 18125105**
2. void TextColor(int color)
   * Find it on the Internet at <http://www.cplusplus.com/forum/beginner/54360/>
   * Then use this function as TextColor(colorcode)
3. void gotoxy(int x, int y)
   * Find it on the Internet at <https://daynhauhoc.com/t/xin-giai-thich-ve-ham-gotoxy/13723>
   * Use this function as gotoxy(x,y)
4. void loading()
   * Create the loading bar
   * Set x = 0
   * Use the for loop and set the value i go to 10 and stop
   * While in the loop, make the cursor move to (45,13) then stop for 0.5s (use sleep() function from the library "windows.h") and print out the space with the cyan blue shading(176) color (when the color code >100 it change the shading color instead and because we print out the space so it will look like a blue box)
   * Then add x by 3 to move the cursor to the next point and print it out
   * Final go to (50,12) to print “cout <<" LOADING . . . " << i << "0%";” in white color each time the loop repeat and it will go from 10% to 100% (i<=10)
5. void AboutScreen()
   * Set the 5 string
   * Go to (35,8) and print the first line word by word by using for loop go from the first letter to the last one, each letter print out has a 0.04s delay
   * Use that code again but change the cordinate for better visuality and between each line delay 0.04s
6. void draw(int height, int width)
   * A function to draw the board using for loop
   * Before the function call a variable char matrix[60][60]
   * The function has 2 for loop:
     + The loop inside is used to set the cordinate of that matrix point to “\_” and after that print out another “|” to have a square at matrix[0][0] then repeat the process to till the j value and end the loop
     + Then paragraph to the next line and do the inside loop again at the second line and repeat untill the i value end. Many i mean many line and many j mean the square will spread out horizontally
7. void play\_er()
   * First we declare char player
   * When our gameplay meet this function, it will change the current move to another style like X to O or O to X
   * We use the if function, if current is X it change the next step to O else it change to X with the color changing (X is blue(9) and O is green(10))
   * So that we just need one button for 2 move
8. int X\_Win(int check)
   * Declare int check and a variable name i and let i = i + check then the end of function return that i for the main to check and use this function
   * Then we include the library <fstream> to use the function “ifstream XWin”, it will declare a varible name XWin
   * Then use XWin.open to open the "XWin.txt" file and the ios::in allows input from a file
   * The while loop with the condition !XWin.eof() is use to scan all the character in that file and stop correctly, then we declare char X and use XWin.get(X) to enter word with color blue(9) and output it by cout << X
   * Out of loop we close the file by XWin.close() and change the color back to white
9. int O\_Win(int check)
   * The same way as X\_Win function
10. void gameplay\_(int q, int p)
    * Use a while loop with the condition out == 0
    * Declare char key = \_getch(), getch is a function to make the console stay still until it get a character from the input device
    * Use switch…case of char key
      + When press w it will change the cordinate to (x-0.5,y) mean go up, here “I” is x cordinate and j is the y one
      + Press s will change the cordinate to (x+0.5,y) mean go down
      + Press s will change the cordinate to (x,y-1) mean go to the left
      + Press d will change the cordinate to (x,y+1) mean go to the right
      + Press k
        - use the function play\_er() to change the move when press k again
        - if (matrix[int(y\*2)][int(x)] == char(79) || matrix[int(y\*2 )][int(x)] == char(88)) this if function is use to define if at that point is already have a move set there or not
        - Then set the variable player to the matrix cordinate then cout << player to print out the move then change back the color to white
        - Next we declare 8 variable for 8 dimension checking winning condition, each time if the cordinate of matrix is a move that player has play it will plus 1 to that variable of that dimension and repeat that again 8 times in a for loop
        - Everytime player play a move, it will check immediately with the for loop
      + Press q and it will add 1 in to the out variable mean it will stop the outside while loop and quit the game
    * End the switch case is a if function use the check above to announce player is win
      + If the win is X it will gotoxy(1,p+2) and use the function X\_Win to print out the text in file and plus “out” by 1 to stop the game
      + It the same with the O ones

Else if nothing happen, the game will continue and the x, y will be plus with I, j and gotoxy(x,y)