**Використовуючи патерни проектування** написати гру «Хрестики нулики».

Передбачити можливість гри проти комп’ютера.

Передбачити можливість відміни ходів (на будь-яку кількість ходів назад), зберегти поточну гру у файл, зчитати з файлу.

\*\*класична версія хрестиків-нуликів передбачає ігрове поле розміром 3\*3. Спробуйте реалізувати користувацький розмір поля.

Завдання

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include<Windows.h>

#include<string>

#include<vector>

#include<map>

#include<time.h>

#include<fstream>

using namespace std;

class Elem

{

char x\_o;

int pos;

public:

Elem(char x\_o,int pos)

{

this->pos = pos;

this->x\_o = x\_o;

}

char& Get\_x\_o()

{

return this->x\_o;

}

int& Get\_pos()

{

return this->pos;

}

};

class GameXO

{

protected:

int pole[3][3];

vector<Elem\*>moves;

void SetPole(int pos, char s)

{

int m;

if (s == 'X')

m = 1;

else

m = 2;

switch (pos)

{

case 1:  
 pole[0][0] = m;  
 break;

case 2:  
 pole[0][1] = m;  
 break;

case 3:  
 pole[0][2] = m;  
 break;

case 4:  
 pole[1][0] = m;  
 break;

case 5:  
 pole[1][1] = m;  
 break;

case 6:  
 pole[1][2] = m;  
 break;

case 7:  
 pole[2][0] = m;  
 break;

case 8:  
 pole[2][1] = m;  
 break;

case 9:  
 pole[2][2] = m;  
 break;

}

}

public:

void Print()

{

system("cls");

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

if (pole[i][j] == 1)

cout << "X ";

else if (pole[i][j] == 2)

cout << "O ";

else

cout << "- ";

}

cout << endl;

}

}

void Get\_x\_y(int a, int\* x, int\* y)

{

switch (a)

{

case 1:

\*x = 0;

\*y = 0;

break;

case 2:

\*x = 0;

\*y = 1;

break;

case 3:

\*x = 0;

\*y = 2;

break;

case 4:

\*x = 1;

\*y = 0;

break;

case 5:

\*x = 1;

\*y = 1;

break;

case 6:

\*x = 1;

\*y = 2;

break;

case 7:

\*x = 2;

\*y = 0;

break;

case 8:

\*x = 2;

\*y = 1;

break;

case 9:

\*x = 2;

\*y = 2;

break;

}

}

bool Win\_or\_not\_win( int k)

{

int sum = 0;

//по горизонталі

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

if (pole[i][j] == k)

sum++;

}

if (sum == 3)

return true;

sum = 0;

}

//по вертикалі

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

if (pole[j][i] == k)

sum++;

}

if (sum == 3)

return true;

sum = 0;

}

//по головній діагоналі

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

if (i == j)

{

if (pole[i][j] == k)

sum++;

}

}

}

if (sum == 3)

return true;

sum = 0;

//по бічній діагоналі

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

if (j == 3 - 1 - i)

{

if (pole[i][j] == k)

sum++;

}

}

}

if (sum == 3)

return true;

return false;

}

/\*int\*\* Get\_pole()

{

return pole;

}\*/

GameXO()

{

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

pole[i][j] = 0;

}

}

}

~GameXO()

{

for (int i = 0; i < moves.size(); i++)

{

delete moves[i];

}

}

void AfterUndo()

{

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

pole[i][j] = 0;

}

}

for (int i = 0; i < moves.size(); i++)

{

SetPole(moves[i]->Get\_pos(), moves[i]->Get\_x\_o());

}

Print();

}

void CreateGame()

{

cout << "Гра почалась\n";

}

void OpenGame()

{

fstream f("Game.txt", ios::in);

int y = 0;

while (!f.eof())

{

int m;

f >> m;

char s;

f >> s;

SetPole(m, s);

moves.push\_back(new Elem(s, m));

}

f.close();

}

void SaveGame()

{

fstream f("Game.txt", ios::out);

for (int i = 0; i < moves.size(); i++)

{

f << endl<< moves[i]->Get\_pos() << " " << moves[i]->Get\_x\_o();

}

f.close();

}

bool MakeAMove(Elem \*el)

{

int x = 0, y = 0;

int pos\_now = el->Get\_pos();

Get\_x\_y(pos\_now, &x, &y);

while (pole[x][y] == 1 || pole[x][y] == 2)

{

cout << "Неккоректний ввід!\n";

cout << "->";

cin >> pos\_now;

while (pos\_now < 1 || pos\_now > 9)

{

cout << "Неккоректний ввід!\n";

cout << "->";

cin >> pos\_now;

}

Get\_x\_y(pos\_now, &x, &y);

}

el->Get\_pos() = pos\_now;

moves.push\_back(el);

SetPole(el->Get\_pos(),el->Get\_x\_o());

if (Win\_or\_not\_win((el->Get\_x\_o() == 'X' ? 1 : 2)))

{

Print();

cout << el->Get\_x\_o() << " переміг\n";

return false;

}

Print();

return true;

}

void ShowInstruction()

{

cout << "1|2|3\n4|5|6\n7|8|9\n";

}

vector<Elem\*>& GetMovesV()

{

return moves;

}

};

class Command

{

protected:

GameXO\* pgame;

Command(GameXO\* p):pgame(p){}

public:

virtual~Command(){}

virtual bool Execute(Elem\*p=NULL) = 0;

};

class CreateGameCommand :public Command

{

public:

CreateGameCommand(GameXO\* p):Command(p){}

bool Execute(Elem\* p = NULL)

{

this->pgame->CreateGame();

return true;

}

};

class OpenGameCommand :public Command

{

public:

OpenGameCommand(GameXO\* p) :Command(p) {}

bool Execute(Elem\* p = NULL)

{

this->pgame->OpenGame();

return true;

}

};

class SaveGameCommand :public Command

{

public:

SaveGameCommand(GameXO\* p) :Command(p) {}

bool Execute(Elem\* p = NULL)

{

this->pgame->SaveGame();

return true;

}

};

class MakeAMoveCommand :public Command

{

public:

MakeAMoveCommand(GameXO\* p) :Command(p) {}

bool Execute(Elem\* p = NULL)

{

if (!(this->pgame->MakeAMove(p)))

return false;

return true;

}

};

class ShowInstructionCommand :public Command

{

public:

ShowInstructionCommand(GameXO\* p) :Command(p) {}

bool Execute(Elem\* p = NULL)

{

this->pgame->ShowInstruction();

return true;

}

};

class UndoCommand :public Command

{

public:

UndoCommand(GameXO\* p) :Command(p) {}

bool Execute(Elem\* p=NULL)

{

int k;

while (p->Get\_pos() > this->pgame->GetMovesV().size()&& p->Get\_pos() >0)

{

cout << "Неккоректний ввід!\nВведіть ще раз";

cout << "->";

cin >> k;

p->Get\_pos() = k;

}

for (int i = 0; i < p->Get\_pos(); i++)

{

this->pgame->GetMovesV().erase(this->pgame->GetMovesV().end() - 1);

}

this->pgame->AfterUndo();

return true;

}

};

class Game

{

vector<Command\*>v;

int i = 0;

GameXO g;

void New\_Game\_with\_human()

{

g.Print();

int x,k,l1;

bool c = true, u=true;

for (int hodu = 0;;)

{

if (hodu % 2 == 0)

{

cout << "Зберегти гру -1,відмінити хід -2, зробити хід -3\n";

cout << "->";

cin >> k;

switch (k)

{

case 1:  
 v.push\_back(new SaveGameCommand(&g));  
 v[i]->Execute();  
 i++;  
 c = false;  
 break;

case 2:

v.push\_back(new UndoCommand(&g));

cout << "Скільки ходів відмінити ->";

cin >> l1;

Elem\* l2 = new Elem('f', l1);

v[i]->Execute(l2);

hodu -= l2->Get\_pos();

u = false;

i++;

break;

}

if (!c)

break;

if (!u)

{

u = true;

continue;

}

cout << "гравець робіть ваш хід\n";

cout << "->";

cin >> x;

while (x < 1 || x>9)

{

cout << "Не коректний ввід!Введіть ще раз(позиція повинна бути в діапазоні від 1 до 9 включно)\n";

cout << "->";

cin >> x;

}

Elem\* el = new Elem('X', x);

v.push\_back(new MakeAMoveCommand(&g));

if (!(v[i]->Execute(el)))

break;

hodu++;

i++;

if (hodu == 9)

{

cout << "Нічия!\n";

break;

}

}

else

{

if (hodu > 0)

{

cout << "Зберегти гру -1,відмінити хід -2, зробити хід -3\n";

cout << "->";

cin >> k;

switch (k)

{

case 1:  
 v.push\_back(new SaveGameCommand(&g));  
 v[i]->Execute();  
 i++;  
 c = false;  
 break;

case 2:

v.push\_back(new UndoCommand(&g));

cout << "Скільки ходів відмінити ->";

cin >> l1;

Elem\* l = new Elem('f', l1);

v[i]->Execute(l);

hodu -= l->Get\_pos();

u = false;

i++;

break;

}

}

if (!c)

break;

if (!u)

{

u = true;

continue;

}

cout << "гравець робіть ваш хід\n";

cout << "->";

cin >> x;

while (x < 1 || x>9)

{

cout << "Не коректний ввід!Введіть ще раз(позиція повинна бути в діапазоні від 1 до 9 включно)\n";

cout << "->";

cin >> x;

}

Elem\* e = new Elem('O', x);

v.push\_back(new MakeAMoveCommand(&g));

if (!(v[i]->Execute(e)))

break;

i++;

hodu++;

if (hodu == 9)

{

cout << "Нічия!\n";

break;

}

}

}

}

void New\_Game\_with\_bot()

{

g.Print();

int x, k, l1;

bool c = true, u = true;

for (int hodu = 0;;)

{

if (hodu % 2 == 0)

{

cout << "Зберегти гру -1,відмінити хід -2, зробити хід -3\n";

cout << "->";

cin >> k;

switch (k)

{

case 1:  
 v.push\_back(new SaveGameCommand(&g));  
 v[i]->Execute();  
 i++;  
 c = false;  
 break;

case 2:

v.push\_back(new UndoCommand(&g));

cout << "Скільки ходів відмінити ->";

cin >> l1;

Elem\* l2 = new Elem('f', l1);

v[i]->Execute(l2);

hodu -= l2->Get\_pos();

u = false;

i++;

break;

}

if (!c)

break;

if (!u)

{

u = true;

continue;

}

cout << "гравець робіть ваш хід\n";

cout << "->";

cin >> x;

while (x < 1 || x>9)

{

cout << "Не коректний ввід!Введіть ще раз(позиція повинна бути в діапазоні від 1 до 9 включно)\n";

cout << "->";

cin >> x;

}

Elem\* el = new Elem('X', x);

v.push\_back(new MakeAMoveCommand(&g));

if (!(v[i]->Execute(el)))

break;

hodu++;

i++;

if (hodu == 9)

{

cout << "Нічия!\n";

break;

}

}

else

{

x = rand() % 9 + 1;

for (int j = 0; j < g.GetMovesV().size(); j++)

{

if (g.GetMovesV()[j]->Get\_pos() == x)

{

x = rand() % 9 + 1;

j = 0;

}

}

Sleep(1000);

Elem\* e = new Elem('O', x);

v.push\_back(new MakeAMoveCommand(&g));

if (!(v[i]->Execute(e)))

break;

i++;

hodu++;

if (hodu == 9)

{

cout << "Нічия!\n";

break;

}

}

}

}

void SavedGame\_with\_bot()

{

int hodu = g.GetMovesV().size() - 1;

g.Print();

int x, k, l1;

bool c = true, u = true;

for (;;)

{

if (hodu % 2 == 0)

{

cout << "Зберегти гру -1,відмінити хід -2, зробити хід -3\n";

cout << "->";

cin >> k;

switch (k)

{

case 1:  
 v.push\_back(new SaveGameCommand(&g));  
 v[i]->Execute();  
 i++;  
 c = false;  
 break;

case 2:

v.push\_back(new UndoCommand(&g));

cout << "Скільки ходів відмінити ->";

cin >> l1;

Elem\* l2 = new Elem('f', l1);

v[i]->Execute(l2);

hodu -= l2->Get\_pos();

u = false;

i++;

break;

}

if (!c)

break;

if (!u)

{

u = true;

continue;

}

cout << "гравець робіть ваш хід\n";

cout << "->";

cin >> x;

while (x < 1 || x>9)

{

cout << "Не коректний ввід!Введіть ще раз(позиція повинна бути в діапазоні від 1 до 9 включно)\n";

cout << "->";

cin >> x;

}

Elem\* el = new Elem('X', x);

v.push\_back(new MakeAMoveCommand(&g));

if (!(v[i]->Execute(el)))

break;

hodu++;

i++;

if (hodu == 9)

{

cout << "Нічия!\n";

break;

}

}

else

{

x = rand() % 9 + 1;

for (int j = 0; j < g.GetMovesV().size(); j++)

{

if (g.GetMovesV()[j]->Get\_pos() == x)

{

x = rand() % 9 + 1;

j = 0;

}

}

Sleep(1000);

Elem\* e = new Elem('O', x);

v.push\_back(new MakeAMoveCommand(&g));

if (!(v[i]->Execute(e)))

break;

i++;

hodu++;

if (hodu == 9)

{

cout << "Нічия!\n";

break;

}

}

}

}

void Saved\_Game\_with\_human()

{

int hodu = g.GetMovesV().size()-1;

g.Print();

int x, k, l1;

bool c = true, u = true;

for (;;)

{

if (hodu == 9)

{

cout << "Нічия!\n";

break;

}

if (hodu % 2 == 0)

{

cout << "Зберегти гру -1,відмінити хід -2, зробити хід -3\n";

cout << "->";

cin >> k;

switch (k)

{

case 1:  
 v.push\_back(new SaveGameCommand(&g));  
 v[i]->Execute();  
 i++;  
 c = false;  
 break;

case 2:

v.push\_back(new UndoCommand(&g));

cout << "Скільки ходів відмінити ->";

cin >> l1;

Elem\* l2 = new Elem('f', l1);

v[i]->Execute(l2);

hodu -= l2->Get\_pos();

u = false;

i++;

break;

}

if (!c)

break;

if (!u)

{

u = true;

continue;

}

cout << "гравець робіть ваш хід\n";

cout << "->";

cin >> x;

while (x < 1 || x>9)

{

cout << "Не коректний ввід!Введіть ще раз(позиція повинна бути в діапазоні від 1 до 9 включно)\n";

cout << "->";

cin >> x;

}

Elem\* el = new Elem('X', x);

v.push\_back(new MakeAMoveCommand(&g));

if (!(v[i]->Execute(el)))

break;

hodu++;

i++;

if (hodu == 9)

{

cout << "Нічия!\n";

break;

}

}

else

{

if (hodu > 0)

{

cout << "Зберегти гру -1,відмінити хід -2, зробити хід -3\n";

cout << "->";

cin >> k;

switch (k)

{

case 1:  
 v.push\_back(new SaveGameCommand(&g));  
 v[i]->Execute();  
 i++;  
 c = false;  
 break;

case 2:

v.push\_back(new UndoCommand(&g));

cout << "Скільки ходів відмінити ->";

cin >> l1;

Elem\* l = new Elem('f', l1);

v[i]->Execute(l);

hodu -= l->Get\_pos();

u = false;

i++;

break;

}

}

if (!c)

break;

if (!u)

{

u = true;

continue;

}

cout << "гравець робіть ваш хід\n";

cout << "->";

cin >> x;

while (x < 1 || x>9)

{

cout << "Не коректний ввід!Введіть ще раз(позиція повинна бути в діапазоні від 1 до 9 включно)\n";

cout << "->";

cin >> x;

}

Elem\* e = new Elem('O', x);

v.push\_back(new MakeAMoveCommand(&g));

if (!(v[i]->Execute(e)))

break;

i++;

hodu++;

if (hodu == 9)

{

cout << "Нічия!\n";

break;

}

}

}

}

public:

~Game()

{

for (int i = 0; i < v.size(); i++)

{

delete v[i];

}

}

void StartWithHuman()

{

bool b = true;

int k;

cout << "створити гру -1, показати інструкцію -2, зчитати гру -3 \n";

cout << "->";

cin >> k;

while (k)

{

switch (k)

{

case 1:  
 v.push\_back(new CreateGameCommand(&g));

v[i]->Execute();

i++;  
 b = false;  
 break;

case 2:

v.push\_back(new ShowInstructionCommand(&g));

v[i]->Execute();

i++;  
 break;

case 3:

v.push\_back(new OpenGameCommand(&g));

v[i]->Execute();

i++;

Saved\_Game\_with\_human();

cout << "Успішно збережено\n";

b = false;

break;

}

if (!b)

break;

cout << "створити гру -1, показати інструкцію -2\n";

cout << "->";

cin >> k;

}

switch (k)

{

case 1:  
 New\_Game\_with\_human();  
 break;

}

}

void StartWithbot()

{

bool b = true;

int k;

cout << "створити гру -1, показати інструкцію -2, зчитати гру -3 \n";

cout << "->";

cin >> k;

while (k)

{

switch (k)

{

case 1:  
 v.push\_back(new CreateGameCommand(&g));

v[i]->Execute();

i++;  
 b = false;  
 break;

case 2:

v.push\_back(new ShowInstructionCommand(&g));

v[i]->Execute();

i++;  
 break;

case 3:

v.push\_back(new OpenGameCommand(&g));

v[i]->Execute();

i++;

SavedGame\_with\_bot();

cout << "Успішно збережено\n";

b = false;

break;

}

if (!b)

break;

cout << "створити гру -1, показати інструкцію -2\n";

cout << "->";

cin >> k;

}

switch (k)

{

case 1:  
 New\_Game\_with\_bot();  
 break;

}

}

};

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

srand(time(0));

int k;

Game a;

cout << "Зіграти з другом -1, зіграти з ботом -2, вихід-0\n";

cin >> k;

switch (k)

{

case 1:  
 a.StartWithHuman();  
 break;

case 2:  
 a.StartWithbot();  
 break;

}

system("pause");

return 0;

}