**Міністерство освіти і науки України**

**Тернопільський національний технічний університет**

**імені Івана Пулюя**

Кафедра комп’ютерних наук

Курсова робота

з дисципліни

«Об’єктно**-**орієнтоване програмування»

на тему:

**"Гра Морський бій"**

Виконав

студент групи СН – 22 \_\_\_\_\_\_\_\_\_ Сеник В.В

(підпис)

Керівник роботи

Асистент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бревус В.М.

(оцінка, підпис, дата)

Тернопіль – 2015

Вступ 3

1. Аналіз технічного завдання 4

Змн.

Лист

№ докум.

Підпис

Дата

Арк.

2

**Кр СН-10-147 ПЗ**

Розроб.

Сеник В.В.

Перевір.

Бревус В.М.

Реценз.

Н. Контр.

Затверд.

**Зміст**

Літ.

Акрушів

16

ТНТУ ФІС гр.СН-22

1. Обґрунтування алгоритму і структури програми 5
2. Розробка програми 6

3.1. Розробка системи класів 6

3.2. Розробка методів 7

4. UML діаграма класів 12

5. Тестування програми та результат її виконання 13

Висновок 15

Список використаної літератури 16

**ВСТУП**

Всі, хто має справу з комп'ютером, так чи інакше стикалися з комп'ютерними іграми, і практично більшість може назвати декілька ігор, які їм особливо сподобалися. Ті, хто вже зовсім награвся, майже награвся або ще не награвся, але в процесі спілкування з комп'ютером вже почав суміщати ігри з чим-небудь кориснішим, можливо, хотіли б придумати які-небудь свої, не схожі ні на які інші ігри

Змн.

Лист

№ докум.

Підпис

Дата

Арк.

3

**Кр СН-10-147 ПЗ**

Розроб.

Сеник В.В.

.

Перевір.

Бревус В.М.

Реценз.

Н. Контр.

Затверд.

**Вступ**

Літ.

Акрушів

16

ТНТУ ФІС гр.СН-22

Багато що захоплює в такій творчості і не сам процес гри, а розробка ігрового всесвіту, її проектування і реалізація. Коли можна злити воєдино сценарій, графіку, музику, майстерно задуманий і вміло запрограмований алгоритм — створити єдиний фантастичний, що живе за законами, які ти ж для нього і придумав.

У даній курсовій роботі мова піде про створення нескладної ігрової програми «Морської бій», яка і буде об'єктом дослідження.

Гра «Морський бій» відома мабуть всім ще з дитинства. Суть гри полягає в тому, щоб потопити всі кораблі на невідомій карті противника, по черзі називаючи координати. Ігрове поле – квадрат 10×10 кожного гравця, на якому розміщується флот кораблів. Перемагає той, хто першим потопить всі кораблі противника.

1. **АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ**

Змн.

Лист

№ докум.

Підпис

Дата

Арк.

4

**Кр СН-10-147 ПЗ**

Розроб.

Сеник В.В

Перевір.

Бревус В.М.

Реценз.

Н. Контр.

Затверд.

**Аналіз технічного завдання**

Літ.

Акрушів

16

ТНТУ ФІС гр.СН-22

## Назва задачі: “Реалізація інтерактивних ігрових програм методами ООП, мовою програмування С++” мета задачі полягає у реалізації інтерактивної ігрової програми «Морський бій» – ця програма відома всім з дитинства і для її реалізації було використано середовище програмування Microsoft Visual Studio 2010.

## Постановка задачі Розробити ігрову програму «Морської бій», яка включає ігрову панель, що складається з двох полів для людини і комп'ютера. На полі розставляються 10 кораблів. Чотири кораблі розміром в одну клітинку, три кораблі розміром в дві клітинки, два кораблі розміром в три клітинки, і один корабель розміром в чотири клітинки. «Палуби» кораблів на полі зображені літерою «K». По черзі супротивникам надається можливість пострілу по чужому полю. При попаданні клітинка збитої «палуби» корабля замінюється літерою «X», і надається можливість додаткового пострілу, в протилежному випадку пуста клітинка змінюється на літеру «O», і хід переходить до противника. Перемога присуджується гравцю, що потопив всі кораблі супротивника.

Запуск програми здійснюється таким чином: необхідно відкрити папку, що містить дану гру, і знайти виконуваний файл **SeaBattle.exe**. На цьому файлі слід виконати подвійне клацання лівою клавішею маніпулятора мишки.

Розроблена програма може працювати на будь-якому ІВМ-сумісному комп’ютері починаючи з 80286 і вище. Робота програми не обмежена об’ємом ОЗП (оперативного запам’ятовуючого пристрою) чи жорсткого диску, але рекомендується мати не менше 2 мегабайтів вільного дискового простору.

1. **ОБГРУНТУВАННЯ АЛГОРИТМУ І СТРУКТУРИ ПРОГРАМИ**

Змн.

Лист

№ докум.

Підпис

Дата

Арк.

5

**Кр СН-10-147 ПЗ**

Розроб.

Сеник В.В.

Перевір.

Бревус В.М.

Реценз.

Н. Контр.

Затверд.

**Обґрунтування алгоритму і структури програми**

Літ.

Акрушів

16

ТНТУ ФІС гр.СН-22

На початку гри на екран виводиться запитання про вибір автоматичної чи самостійної розстановки кораблів. При виборі автоматичної розстановки автоматично вибирається напрям корабля і його координати. Якщо ж вибрати самостійну розстановку, то з’явиться повідомлення про вибір напряму і координати першої «палуби» корабля, при цьому здійснюється перевірка суміжних клітин. Корабель не розміститься, якщо на сусідніх клітинах буде виявлено інший корабель. Кораблі комп’ютера розставляються автоматично.

Після розміщення кораблів по полю, починається сама гра. Першим «стріляє» гравець. Надається вибір координати на карті комп’ютера, при цьому вносяться зміни в масиви кораблів комп’ютера, та пострілів гравця. Координати за якими стріляє комп’ютер вибираються випадково, при цьому перевіряється, чи вибрав вже комп’ютер ці координати.

В масив карти добавляється інформація з масиву кораблів гравця, де вони зображені літерами «K». Потім в туди заноситься інформація про постріли зроблені гравцями, і відбувається виведення на екран масиву карти, при цьому використовуються масиви кораблів і пострілів гравця і комп’ютера.

Потім встановлюються і перевіряються прапори гравців. Якщо в комп’ютера не залишилося неушкоджених кораблів, на екран виводиться повідомлення «Переміг гравець!», якщо ж кораблів не залишилось в гравця то з’являється «Переміг комп’ютер!».

1. **РОЗРОБКА ПРОГРАМИ**

Змн.

Лист

№ докум.

Підпис

Дата

Арк.

6

**Кр СН-10-147 ПЗ**

Розроб.

Сеник В.В.

Перевір.

Бревус В.М.

Реценз.

Н. Контр.

Затверд.

**Розробка програми**

Літ.

Акрушів

16

ТНТУ ФІС гр.СН-22

* 1. **Розробка системи класів**

Для того щоб забезпечити виконання поставлених завдань у програмі використовується два основних класи: клас ігрового поля – class BattleField, та похідний від нього клас гравця – class Player. В класі BattleField ініціалізуються масиви кораблів і пострілів (hits, ships), розставляються кораблі, і здійснюється перевірка можливості розташування кораблів в даній координаті.

Клас BattleField є батьківським класом для класу Player. В даному класі вносяться зміни в масиви hits і ships після здійснення ходів комп’ютера та гравця. Для збереження напрямків (горизонтального та вертикального) використовується перечислення enum.

* 1. **Розробка методів**

Клас BattleField використовує методи Set, ShipPlase та ShipInit.

Void ShipsInit – здійснює ініціалізацію кораблів.

void player**::**ships\_init**()**

**{**

for **(**int i **=** 0**;** i **<** 10**;** i**++)**

**{**

for **(**int j **=** 0**;** j**<** 10**;** j**++)**

**{**

ships**[**i**][**j**] =** 1**;**

hits**[**i**][**j**] =** 0**;**

**}**

**}**

*// розстановка кораблів*

set**(**4**);**

set**(**3**);**

set**(**3**);**

set**(**2**);**

set**(**2**);**

set**(**2**);**

set**(**1**);**

set**(**1**);**

set**(**1**);**

set**(**1**);**

**}**

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

7

**Кр СН-10-147 ПЗ**

Void Set – розташовує кораблі, та здійснює перевірку суміжних клітин.

void player**::**set **(**int deck**)**

**{**

int my **=** deck **-** 1**;**

bool isset **=** 0**;**

int s**,**c**;**

direction dir**;**

while **(**isset **==** 0**)** *// перевірка чи вдалось розмістити корабель*

**{**

dir **=** static\_cast**<**direction**>(**rand**()%**2**);** *// вибираємо напрямок*

s **=** rand**()%**10**;**

c **=** rand**()%**10**;**

int e **=** 0**;**

switch **(**dir**)**

**{**

case h**:**

if **(**ships**[**s**][**c**+**deck**-**1**] ==** 1**)**

**{**

e **=** place\_ship**(**s**,**c**,**dir**,**deck**);**

if **(**e **==** 0**)**

**{**

for **(**int i **=** 0**;** i **<** deck**;** i**++)**

**{**

ships**[**s**][**c**+**i**] =** 2**;**

**}**

isset **=** 1**;**

**}**

**}**

break**;**

case v**:**

if **(**ships**[**s**+**deck**-**1**][**c**] ==** 1**)**

**{**

e **=** place\_ship**(**s**,**c**,**dir**,**deck**);**

if **(**e **==** 0**)**

**{**

for **(**int i **=** 0**;** i **<** deck**;** i**++)**

**{**

ships**[**s**+**i**][**c**] =** 2**;**

**}**

isset **=** 1**;**

**}**

**}**

break**;**

**}**

**}**

**}**

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

8

**Кр СН-10-147 ПЗ**

Int PlaceShip – перевіряє, чи можна в даній координаті [s] [c] розмістити корабель з палубами deck.

int player**::**place\_ship**(**int s**,** int c**,** direction dir**,** int deck**)**

**{**

int e **=** 0**;**

switch **(**dir**)**

**{**

case h**:**

if **(**ships**[**s**-**1**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

9

**Кр СН-10-147 ПЗ**

**}**

if **(**ships**[**s**-**1**][**c**+**deck**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**1**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**1**][**c**+**deck**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**][**c**+**deck**] ==** 2**)**

**{**

e **=** 1**;**

**}**

for **(**int i **=** 0**;** i **<** deck**;** i**++)**

**{**

if **(**ships**[**s**-**1**][**c**+**i**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**1**][**c**+**i**] ==** 2**)**

**{**

e **=** 1**;**

**}**

**}**

break**;**

case v**:**

if **(**ships**[**s**-**1**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**-**1**][**c**+**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**deck**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**deck**][**c**+**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**-**1**][**c**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**deck**][**c**] ==** 2**)**

**{**

e **=** 1**;**

**}**

for **(**int i **=** 0**;** i **<** deck**;** i**++)**

**{**

if **(**ships**[**s**+**i**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**i**][**c**+**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

**}**

break**;**

**}**

return e**;**

**}**

void player**::**turn**(**player**&** enemy**)**

**{**

bool flag **=** 0**;**

while **(**flag **==** 0**)**

**{**

int character **=** rand**()%**10**;** *// координати по яких стрілятиме компютер*

int digit **=** rand**()%**10**;** *// рандом*

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

10

**Кр СН-10-147 ПЗ**

if **(**hits**[**character**][**digit**] !=** 1**)** *// перевірка, чи не вибрав комп’ютер ці координати*

**{**

hits**[**character**][**digit**] =** 1**;**

flag **=** 1**;**

if **(**enemy**.**ships**[**character**][**digit**] ==** 2**)**

**{**

enemy**.**ships**[**character**][**digit**] =** 3**;**

**}**

**}**

**}**

**}**

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

11

**Кр СН-10-147 ПЗ**

Функціїя CheckEnding встановлює і перевіряє прапори defeat\_flag гравців.

int check\_ending**()**

**{**

int flag **=** 0**;**

int human\_flag **=** 0**;**

int computer\_flag **=** 0**;**

for **(**int i**=**0**;** i**<**10**;**i**++)**

**{**

for **(**int j**=**0**;** j**<**10**;**j**++)**

**{**

if **(**human**.**ships**[**i**][**j**] ==** 2**)**

human\_flag **=** 1**;** *// у гравця ще є цілі кораблі*

if **(**computer**.**ships**[**i**][**j**] ==** 2**)**

computer\_flag **=** 1**;** *// у комп’ютера ще є цілі*

**}**

**}**

if **(**human\_flag **==** 0**)**

flag **=** 2**;**

if **(**computer\_flag **==** 0**)**

flag **=** 1**;**

if **(**flag **==** 1**)**

**{**

std**::**cout **<<** "Виграв гравець!\n"**;**

return 2**;**

**}**

if **(**flag **==** 2**)**

**{­­­**

std**::**cout **<<** "Виграв комп’ютер!\n"**;**

return 2**;**

**}**

**}**

Змн.

Арк.

№ докум.

Підпис

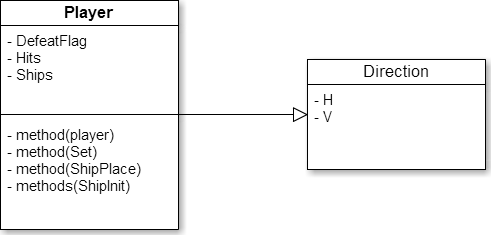
Дата

Арк.

12

**Кр СН-10-147 ПЗ**

1. **UML ДІАГРАМА КЛАСІВ**



1. **ТЕСТУВАННЯ ПРОГРАМИ ТА РЕЗУЛЬТАТ ЇЇ ВИКОНАННЯ**

Тестування програми здійснювалося на комп’ютерах в яких була встановлена операційна система Windows 7 або Windows XP.

Змн.

Лист

№ докум.

Підпис

Дата

Арк.

13

**Кр СН-10-147 ПЗ**

Розроб.

Сеник В.В.

Перевір.

Бревус В.М.

Реценз.

Н. Контр.

Затверд.

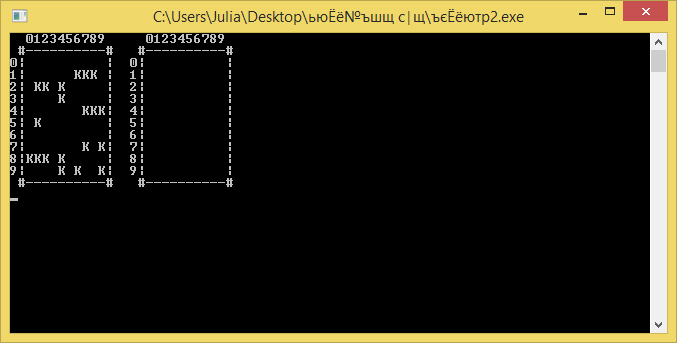
**Тестування програми та результат її виконання**

Літ.

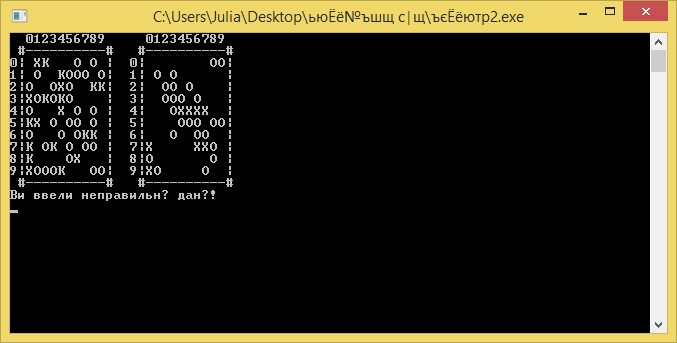
Акрушів

16

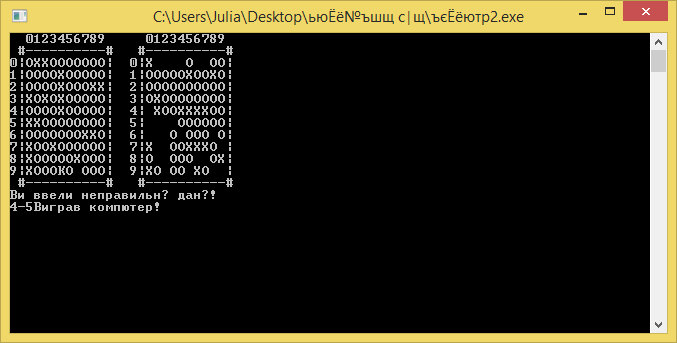
ТНТУ ФІС гр.СН-22



Мал 5.1 – запуск гри



Мал 5.2 – процес гри



Мал 5.3 – кінець гри (Переміг гравець)

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

14

**Кр СН-10-147 ПЗ**

**ВИСНОВКИ**

Мова програмування С++ є універсальною. Вона створена з метою зробити процес написання програм більш приємним для програмістів. Завдяки використанню визначення нових типів програміст може розділяти програму на частини, що робить програму гнучкішою. Також ООП дає можливість створення власних класів. Саме ці властивості були використані в даній програмі.

Застосувавши для розробки власної програми середовище Microsoft Visual Studio 2010я переконався в тому, що використання методів об’єктно – орієнтованого програмування приводить до:

1. Скорочення часу розробки програми;
2. Скорочення програмного коду;
3. Зменшення часу виконання програми;
4. Збільшення ефективності програмування;
5. Збільшення гнучкості програми;

Змн.

Лист

№ докум.

Підпис

Дата

Арк.

15

**Кр СН-10-147 ПЗ**

Розроб.

Сеник В.В.

Перевір.

Бревус В.М.

Реценз.

Н. Контр.

Затверд.

**Висновки**

Літ.

Акрушів

16

ТНТУ ФІС гр.СН-22

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Белецкий Я. Энциклопедия языка Си: Пер. с польск. — М.: Мир, 1992. — 687 с.
2. Дьюрхаст С., Старк К. Программирование на С++ – К.: “ДіаСофт”, 1993.
3. Сван Т. Освоение Borland C++ 4.5.Практический курс. –К.: “Діалектика ”
4. Уейт М., Прата С., Мартін Д. Язык Си. – Москва: “Мир” , 1988.
5. Страустрап Б. Язык программирования С++ – К.: “ДіаСофт”, 1993.`
6. Фейсон Т. Объектно-ориентированое программирование на Borland C++ 4.5. – К.: “Діалектика”, 1996.
7. Вайнер Р. Пінсон Л. С++ изнутри – К.: “ДіаСофт”, 1993.

Змн.

Лист

№ докум.

Підпис

Дата

Арк.

16

**Кр СН-10-147 ПЗ**

Розроб.

Сеник В.В.

Перевір.

Бревус В.М.

Реценз.

Н. Контр.

Затверд.

**Список використаної літератури**

Літ.

Акрушів

16

ТНТУ ФІС гр.СН-22

# ДОДАТОК А

## Текст програми

#include <clocale>

#include <conio.h>

#include <stdlib.h>

#include <iostream>

#include <ctime>

enum direction**{**h**,**v**};**

class player

**{**

public**:**

bool defeat\_flag**;**

int hits**[**10**][**10**];**

int ships**[**10**][**10**];**

player **():** defeat\_flag**(**0**) {}**

void ships\_init**();**

void set**(**int deck**);**

int place\_ship**(**int s**,** int c**,** direction dir**,** int deck**);**

void turn**(**player**&,** int character**,** int digit**);**

void turn**(**player**&);**

**};**

*// карта розміром в 13 рядків та 29 стовбців*

const int s **=** 13**,** c **=** 29**;**

char map**[**s**][**c**] = {**

" 0123456789 0123456789 "**,**

" #----------# #----------#"**,**

"0| | 0| |"**,**

"1| | 1| |"**,**

"2| | 2| |"**,**

"3| | 3| |"**,**

"4| | 4| |"**,**

"5| | 5| |"**,**

"6| | 6| |"**,**

"7| | 7| |"**,**

"8| | 8| |"**,**

"9| | 9| |"**,**

" #----------# #----------#" **};**

player human**;**

player computer**;**

void map\_init**(**char map**[**s**][**c**]);**

void show **(**char map**[**s**][**c**]);**

int input**(**char**&,** char**&);**

void test**();**

int check\_ending**();**

char numbers**[**10**] = {** '0'**,**'1'**,**'2'**,**'3'**,**'4'**,**'5'**,**'6'**,**'7'**,**'8'**,**'9'**};**

int main**()**

**{**

setlocale**(**LC\_CTYPE**,** "Russian"**);**

int message **=** 0**;** *// змінна зберігає код повідомлення*

srand**(** static\_cast**<**unsigned int**>(**time**(**NULL**)) );**

human**.**ships\_init**();** *// створення масиву гравця*

computer**.**ships\_init**();** *// створення масиву компютера*

map\_init**(**map**);**

while **(**message **!=** 2**)**

**{**

int user\_input **=** 0**;**

system**(**"cls"**);** *// очищення екрану*

show**(**map**);**

if **(**message **==** 1**)** *// код повідомлення 1-введено неправильні дані*

std**::**cout **<<** "Ви ввели неправильні дані!\n"**;**

message **=** 0**;**

char character**,** digit**;**

user\_input **=** input**(**character**,** digit**);**

if **(**user\_input **==** 1**)**

**{**

message **=** 1**;**

continue**;**

**}**

human**.**turn**(**computer**,**character**,** digit**);**

computer**.**turn**(**human**);**

message **=** check\_ending**();**

**}**

\_getch**();**

return 0**;**

**}**

void map\_init**(**char map**[**s**][**c**])**

**{**

for **(**int i**=**0**;**i **<** 10**;** i**++)**

**{**

for **(**int j**=**0**;** j **<** 10**;** j**++)**

**{**

if **(**human**.**ships**[**i**][**j**] ==** 2**)**

map**[**i**+**2**][**j**+**2**] =** 'К'**;**

**}**

**}**

**}**

void show**(**char map**[**s**][**c**])**

**{**

for **(**int i**=**0**;** i**<**10**;** i**++)**

**{**

for **(**int j**=**0**;** j**<**10**;** j**++)**

**{**

if **(**computer**.**hits**[**i**][**j**] ==** 1 **&&** human**.**ships**[**i**][**j**] ==** 3**)**

**{**

map**[**i**+**2**][**j**+**2**] =** 'Х'**;**

**}**

else if **(**computer**.**hits**[**i**][**j**] ==** 1 **&&** human**.**ships**[**i**][**j**] !=** 3**)**

**{**

map**[**i**+**2**][**j**+**2**] =** 'О'**;**

**}**

if **(**computer**.**ships**[**i**][**j**] ==** 3 **&&** human**.**hits**[**i**][**j**] ==** 1**)**

**{**

map**[**i**+**2**][**j**+**2**+**15**] =** 'Х'**;**

**}**

else if **(**computer**.**ships**[**i**][**j**] !=** 3 **&&** human**.**hits**[**i**][**j**] ==** 1**)**

**{**

map**[**i**+**2**][**j**+**2**+**15**] =** 'O'**;**

**}**

**}**

**}**

for **(**int i **=** 0**;** i **<** s**;** i**++)**

**{**

for **(**int j **=** 0**;** j **<** c**;** j**++)**

**{**

std**::**cout **<<** map**[**i**][**j**];**

**}**

std**::**cout **<<** "\n"**;**

**}**

**}**

*// ввід координат*

int input**(**char**&** character**,** char**&** digit**)**

**{**

character **=** \_getche**();** *// ввід горизонтальних координат*

int match **=** 0**;**

for **(**int i **=** 0**;** i **<** 10**;** i**++)**

if **(**numbers**[**i**] ==** character**)**

**{**

match **=** 1**;**

character **=** i**;**

**}**

if **(**match **==** 0**)**

return 1**;**

std**::**cout **<<** '-'**;**

match **=** 0**;**

digit **=** \_getche**();** *// ввід вертикальних координат*

for **(**int i **=** 0**;** i **<** 10**;** i**++)**

if **(**numbers**[**i**] ==** digit**)**

**{**

match **=** 1**;**

digit **=** i**;**

**}**

if **(**match **==** 0**)**

return 1**;**

return 0**;**

**}**

void test**()**

**{**

std**::**cout **<<** std**::**endl**;**

for **(**int i **=** 0**;** i **<** 10**;** i**++)**

**{**

for **(**int j **=** 0**;** j **<** 10**;** j**++)**

**{**

std**::**cout **<<** human**.**hits**[**i**][**j**];**

**}**

std**::**cout **<<** " "**;**

for **(**int j **=** 0**;** j **<** 10**;** j**++)**

**{**

std**::**cout **<<** computer**.**hits**[**i**][**j**];**

**}**

std**::**cout **<<** std**::**endl**;**

**}**

std**::**cout **<<** std**::**endl**;**

for **(**int i **=** 0**;** i **<** 10**;** i**++)**

**{**

for **(**int j **=** 0**;** j **<** 10**;** j**++)**

**{**

std**::**cout **<<** human**.**ships**[**i**][**j**];**

**}**

std**::**cout **<<** " "**;**

for **(**int j **=** 0**;** j **<** 10**;** j**++)**

**{**

std**::**cout **<<** computer**.**ships**[**i**][**j**];**

**}**

std**::**cout **<<** std**::**endl**;**

**}**

**}**

int check\_ending**()**

**{**

int flag **=** 0**;**

int human\_flag **=** 0**;**

int computer\_flag **=** 0**;**

for **(**int i**=**0**;** i**<**10**;**i**++)**

**{**

for **(**int j**=**0**;** j**<**10**;**j**++)**

**{**

if **(**human**.**ships**[**i**][**j**] ==** 2**)**

human\_flag **=** 1**;** *// у гравця ще є цілі кораблі*

if **(**computer**.**ships**[**i**][**j**] ==** 2**)**

computer\_flag **=** 1**;** *// у комп’ютера ще є цілі*

**}**

**}**

if **(**human\_flag **==** 0**)**

flag **=** 2**;**

if **(**computer\_flag **==** 0**)**

flag **=** 1**;**

if **(**flag **==** 1**)**

**{**

std**::**cout **<<** "Виграв гравець!\n"**;**

return 2**;**

**}**

if **(**flag **==** 2**)**

**{­­­**

std**::**cout **<<** "Виграв комп’ютер!\n"**;**

return 2**;**

**}**

**}**

void player**::**ships\_init**()**

**{**

for **(**int i **=** 0**;** i **<** 10**;** i**++)**

**{**

for **(**int j **=** 0**;** j**<** 10**;** j**++)**

**{**

ships**[**i**][**j**] =** 1**;**

hits**[**i**][**j**] =** 0**;**

**}**

**}**

*// розстановка кораблів*

set**(**4**);**

set**(**3**);**

set**(**3**);**

set**(**2**);**

set**(**2**);**

set**(**2**);**

set**(**1**);**

set**(**1**);**

set**(**1**);**

set**(**1**);**

**}**

void player**::**set **(**int deck**)**

**{**

int my **=** deck **-** 1**;**

bool isset **=** 0**;**

int s**,**c**;**

direction dir**;**

while **(**isset **==** 0**)** *// перевірка чи вдалось розмістити корабель*

**{**

dir **=** static\_cast**<**direction**>(**rand**()%**2**);** *// вибираємо напрямок*

s **=** rand**()%**10**;**

c **=** rand**()%**10**;**

int e **=** 0**;**

switch **(**dir**)**

**{**

case h**:**

if **(**ships**[**s**][**c**+**deck**-**1**] ==** 1**)**

**{**

e **=** place\_ship**(**s**,**c**,**dir**,**deck**);**

if **(**e **==** 0**)**

**{**

for **(**int i **=** 0**;** i **<** deck**;** i**++)**

**{**

ships**[**s**][**c**+**i**] =** 2**;**

**}**

isset **=** 1**;**

**}**

**}**

break**;**

case v**:**

if **(**ships**[**s**+**deck**-**1**][**c**] ==** 1**)**

**{**

e **=** place\_ship**(**s**,**c**,**dir**,**deck**);**

if **(**e **==** 0**)**

**{**

for **(**int i **=** 0**;** i **<** deck**;** i**++)**

**{**

ships**[**s**+**i**][**c**] =** 2**;**

**}**

isset **=** 1**;**

**}**

**}**

break**;**

**}**

**}**

**}**

int player**::**place\_ship**(**int s**,** int c**,** direction dir**,** int deck**)**

**{**

int e **=** 0**;**

switch **(**dir**)**

**{**

case h**:**

if **(**ships**[**s**-**1**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**-**1**][**c**+**deck**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**1**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**1**][**c**+**deck**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**][**c**+**deck**] ==** 2**)**

**{**

e **=** 1**;**

**}**

for **(**int i **=** 0**;** i **<** deck**;** i**++)**

**{**

if **(**ships**[**s**-**1**][**c**+**i**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**1**][**c**+**i**] ==** 2**)**

**{**

e **=** 1**;**

**}**

**}**

break**;**

case v**:**

if **(**ships**[**s**-**1**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**-**1**][**c**+**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**deck**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**deck**][**c**+**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**-**1**][**c**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**deck**][**c**] ==** 2**)**

**{**

e **=** 1**;**

**}**

for **(**int i **=** 0**;** i **<** deck**;** i**++)**

**{**

if **(**ships**[**s**+**i**][**c**-**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

if **(**ships**[**s**+**i**][**c**+**1**] ==** 2**)**

**{**

e **=** 1**;**

**}**

**}**

break**;**

**}**

return e**;**

**}**

void player**::**turn**(**player**&** enemy**)**

**{**

bool flag **=** 0**;**

while **(**flag **==** 0**)**

**{**

int character **=** rand**()%**10**;** *// координати по яких стрілятиме компютер*

int digit **=** rand**()%**10**;** *// рандом*

if **(**hits**[**character**][**digit**] !=** 1**)** *// перевірка, чи не вибрав комп’ютер ці координати*

**{**

hits**[**character**][**digit**] =** 1**;**

flag **=** 1**;**

if **(**enemy**.**ships**[**character**][**digit**] ==** 2**)**

**{**

enemy**.**ships**[**character**][**digit**] =** 3**;**

**}**

**}**

**}**

**}**

void player**::**turn**(**player**&** enemy**,** int character**,** int digit**)**

**{**

hits**[**character**][**digit**] =** 1**;**

if **(**enemy**.**ships**[**character**][**digit**] ==** 2**)**

**{**

enemy**.**ships**[**character**][**digit**] =** 3**;**

**}**

**}**