НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КУРСОВА РОБОТА

по дисципліні «Програмування»

|  |  |
| --- | --- |
|  | Виконав  студент І курсу  групи КІ-20010  Воробець А.А.  (Призвіще, ініціали)  Перевірили  к.т.н., доцент  Боярінова Ю.Є.\_\_\_\_\_\_\_  Реутська Ю.Ю. \_\_\_\_\_\_\_\_  Оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_  ECTS \_\_\_\_\_\_\_\_\_\_\_\_\_\_  Кількість балів \_\_\_\_\_\_ |

Київ 2021

**Зміст**

[**Вступ** 2](#_Toc73955506)

[**Постановка задачі** 3](#_Toc73955507)

[**Структурна-схема алгоритму вирішення задачі** 4](#_Toc73955508)

[**Опис програми** 5](#_Toc73955509)

[**Опис функцій** 6](#_Toc73955510)

[Власно розроблених функцій 6](#_Toc73955511)

[**Результати роботи програми** 11](#_Toc73955512)

[**Висновки** 13](#_Toc73955513)

[**Літературні джерела** 14](#_Toc73955514)

[**Додаток** **А. Текст програми** 15](#_Toc73955515)

# **Вступ**

Вивчаючи мову програмування С, я вирішив написати гру хрестики-нулики на С, на це є декілька причин.

Основними перевагами мови С вважаться висока переча написаних на нiй програм мiж комп'ютерами з різною архітектурою, мiж різними операційними середовищами. Транслятори мови С дiять практично на всiх персональних комп'ютерах, які використовуються в наш час. Перечислимо окремі особливості мови С:

-в мові С реалізовані окремі операцii низького рівня (а саме операцii над бітами). Окремі з таких операцій напряму вiдповiдають машинним командам;

- мова С підтримує механізм показчикiв на змiннi i функцii;

-базовi типи даних мови С відображають тi ж об’єкти, з якими потрібно мати справу програмі на мові Асемблера,- байти, машинні слова, символи, строки;

- як в нiякiй iншiй мові програмування високого рівня в мові С накладаються лише незначні обмеження на перетворення типів даних;

- не дивлячись на ефективність і потужність конструкцii мови С, він відносно малий по об'єму, але в системне оточення мови С входить бібліотека стандартних функцій, в який реалізовані встроенi оператори введення - виведення, динамічного розподілу пам’ять, управління процесами i т.д.

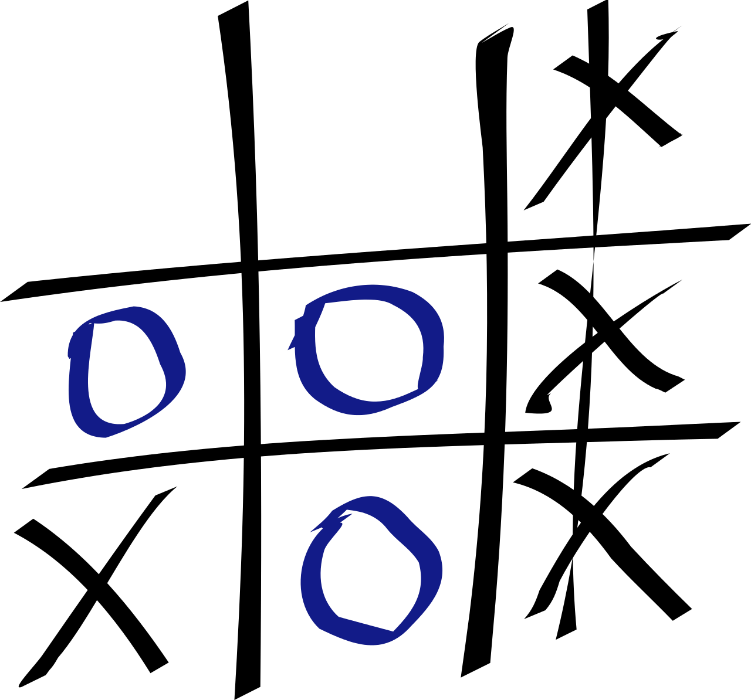
# **Постановка задачі**

Хрестики-нулики - логічна гра між двома противниками на квадратному полі 3 на 3 клітини або більшого розміру (аж до «нескінченного поля»). Один з гравців грає «хрестиками», другий - «нуликами». У традиційній китайській грі (Гомокі) використовуються чорні і білі камені.

Правила гри

Гравці по черзі ставлять на вільні клітини поля 3х3 знаки (один завжди хрестики, інший завжди нулики). Перший, який вибудував у ряд 3 своїх фігури по вертикалі, горизонталі або діагоналі, виграє. Перший хід робить гравець, що ставить хрестики.

Зазвичай після закінчення партії виграла, закреслює рисою свої три знака (нулика або хрестика), що становлять суцільний ряд.



Рисунок

# **Структурна-схема алгоритму вирішення задачі**

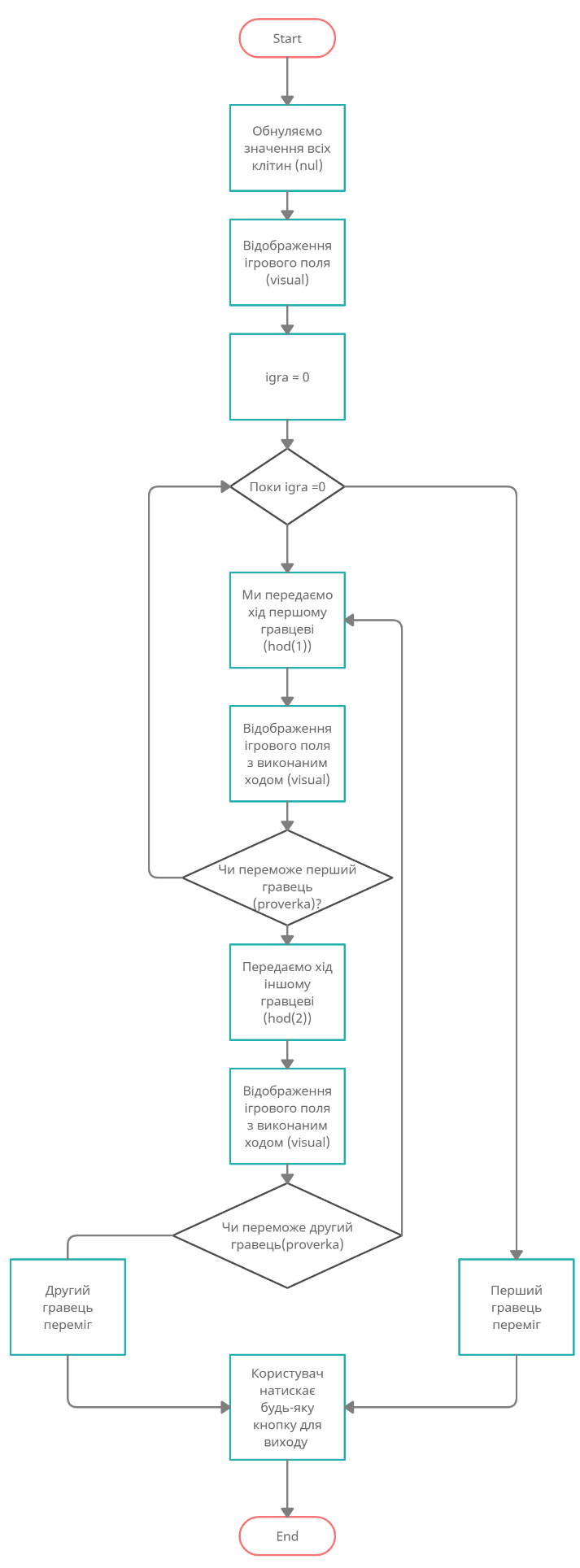


Рисунок Структурна-схема алгоритму вирішення задачі

# **Опис програми**

Короткий опис програми хрестики-нулики на С. При запуску програми показується ігрове поле, після чого перший гравець повинен ввести номер рядка і стовпця клітини, на якій він хоче зробити хід. Потім такі ж дії повинен здійснити другий гравець.

Програма зчитує номер рядка і стовпця, введені користувачем, перевіряє, чи зайняте вибране місце, якщо зайнято, то пропонує зробити хід заново, якщо немає, то хід буде зроблений. Якщо хто-небудь з гравців переміг, програма виводить повідомлення про те, хто з гравців переміг, після чого користувач натискають будь-яку кнопку, щоб вийти з програми. Змінні і масиви. Масив a [i] [j] відповідає за номер рядка і стовпця кожної клітини. i - номер рядка клітини, j - номер стовпця клітини.

Якщо a [i] [j] = 0 то дана клітина вільна

Якщо a [i] [j] = 1 то дана клітина зайнята нуликів

Якщо a [i] [j] = 2 то дана клітина зайнята хрестиком

Змінні igra і res відповідають за стан гри, якщо ці змінні дорівнюють нулю, то поки ніхто не переміг, якщо 1 - переміг нулик, якщо 2 - переміг хрестик. У змінну hi записується хід гравця, наприклад, «12» - перший рядок, другий стовпець. str і sto відповідають за номер обраних рядки і стовпці. ver - перевіряє можливість ходу, якщо хід неможливий - 0, якщо можливий - 1.

У програмі є чотири процедури.

# **Опис функцій**

# Власно розроблених функцій

1. nul()
2. visual()
3. hod(n)
4. proverka()

1. nul () обнуляє знаходяться елементи кожної клітини на поле, це потрібно для початку гри. Якщо клітина була порожньою, то процедура залишить її порожній, якщо клітина була зайнята хрестиком або нуликів, процедура прибере їх звідти.

Изображение выглядит как текст, человек, снимок экрана

Автоматически созданное описание

Рисунок

2. visual () призначена для візуалізації поля при кожному ході. Якщо клітина порожня, то нічого в ній не вималюється, якщо на клітці зробили хід, то процедура намалює цей хід хрестик або нулик.

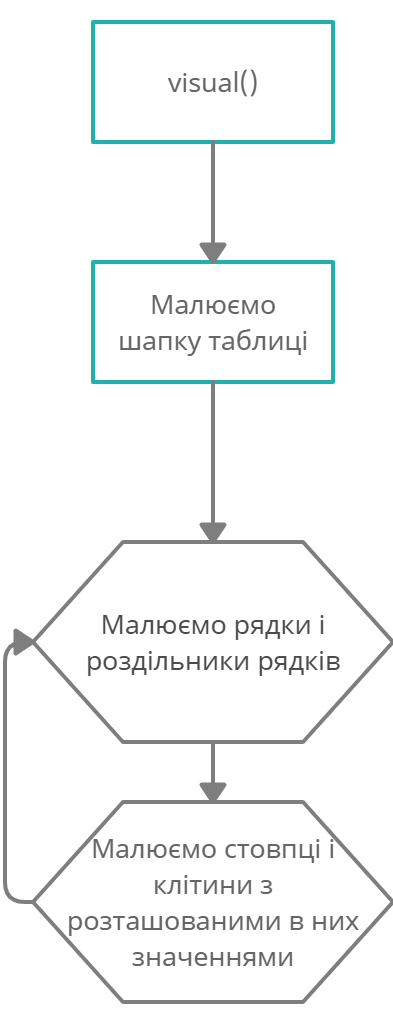


Рисунок Алгоритм процедури візаулізаціі поля

3.Процедура hod (n). Процедура запитує номер рядка і стовпця в клітини, в яку ходить гравець і призначає на обрану клітку нулик або хрестик. n - номер гравця, який повинен ходити в даний момент. Якщо n = 1 то ходить нулик, якщо n = 2, то ходить хрестик.

Для перевірки коректності ходу, клітина повинна бути порожня і номер повинен бути один з номерів таблиці 3 на 3.

if (a [str] [sto] == 0 && (hi == 11 || hi == 12 || hi == 13 || hi == 21 || hi == 22 || hi == 23 || hi == 31 || hi == 32 || hi == 33)) (ver = 1);



Рисунок Алгоритм процедури хід

4. proverka () при кожному ході перевіряє, чи не переміг чи будь-якої гравець, і повертає одне з можливих значень. Якщо proverka () = 0, то ніхто поки не переміг або трапилася нічия, якщо proverka () = 1, то переміг нулик, якщо 2 - хрестик.

При перевірці ми перевіряємо в циклі for всі рядки і шукаємо є в будь-якої рядку всі нулики

// пробігаємо по всіх рядках

for (i = 1; i <= 3; i ++)

{

// перевірка чи в рядку нулики

if (a [i] [1] == 1 && a [i] [2] == 1 && a [i] [3] == 1) {res = 1;

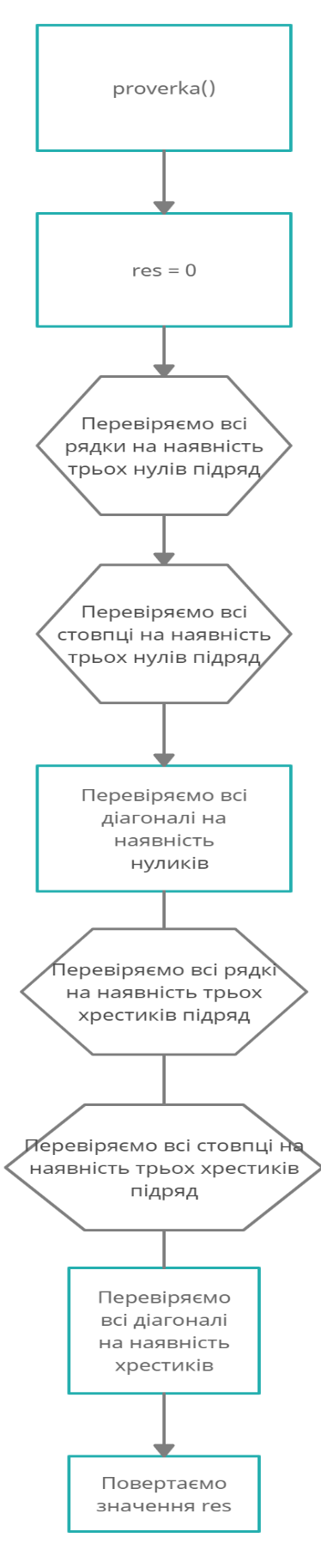


Рисунок Проверка

5. Аналогічно йде проверрка по всіх стовпцях і перевірка діагоналей

if (a[1][1]==1 && a[2][2]==1 && a[3][3]==1 )  {res=1;}

if (a[1][3]==1 && a[2][2]==1 && a[3][1]==1 )  {res=1;}

# **Результати роботи програми**

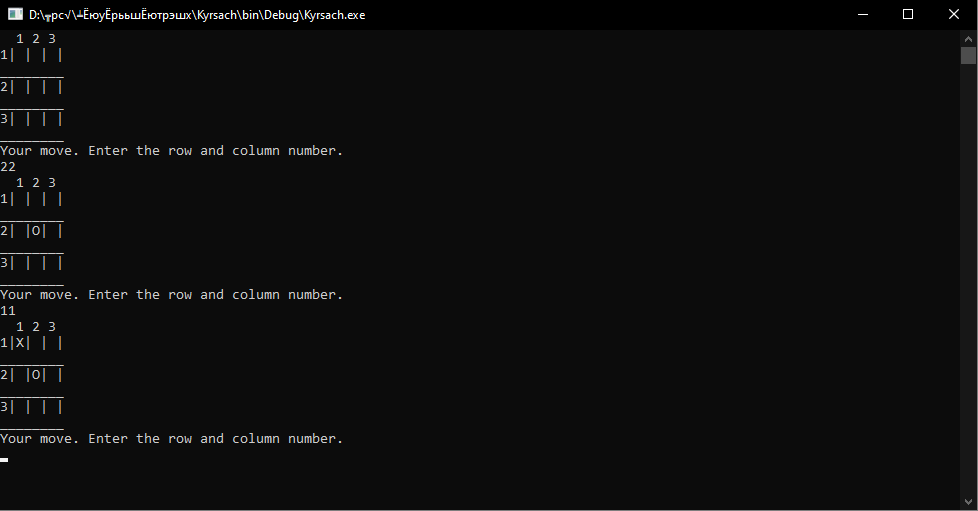


Рисунок 7 перший хід

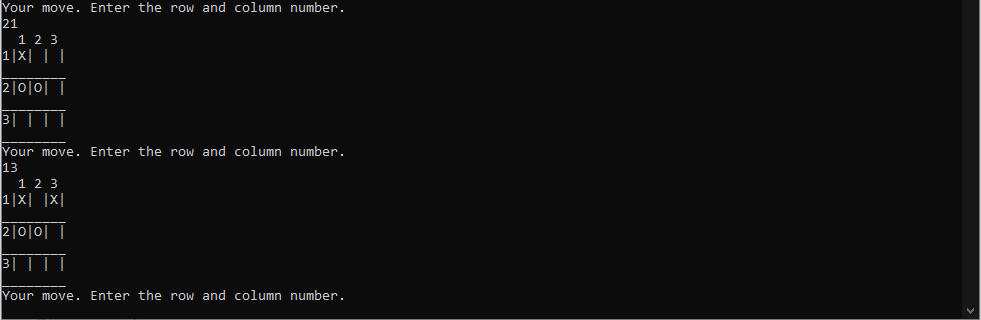


Рисунок 8 Другій хід обох гравців

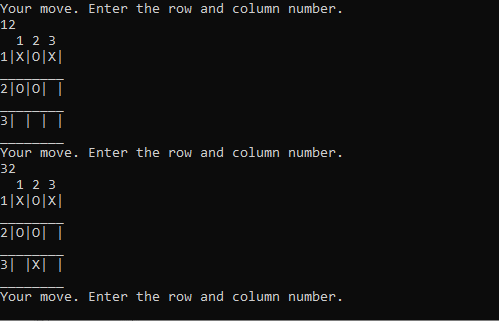


Рисунок 9 Третій хід обох гравців

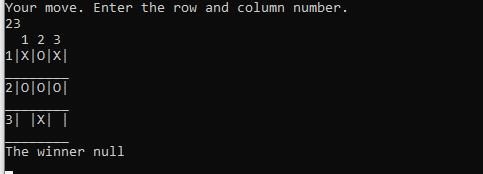


Рисунок 10 Об'явлення переможця

# 

# **Висновки**

Створена гра хрестики-нулики на основі знань, получений у процесі навчання на першому курсі, спеціальності комп’ютерна інженерія.

Програм провірена на роботоздатність, розроблене игрове поле для користувача.

На мою думку, гра реалізована сиро, хоча є можливість пришвидшити її роботу, оптимізувавши код. Також можливо до неї додати комп’ютер який буде грати замість другого гравця.

Використані теми з програмування:

* Робота з циклами
* Робота з масивами
* Робота з процедурами
* Робота з власними функціями

# **Літературні джерела**

1. Simple and Fast Multimedia Library *SFML* : веб-сайт. URL : [(sfml-dev.org)](https://www.sfml-dev.org/).
2. C Уроки \ Разработка игр : веб-сайт. URL : <https://kychka-pc.ru>
3. Форум программістів StackOverflow. URL: Stack Overflow - Where Developers Learn, Share, & Build Careers.

# Додаток А. Текст програми

#include <stdio.h>

#include <conio.h>

int a[3][3];

void nul()

{

int i,j;

for(i=1; i<=3; i++)

{

for (j=1;j<=3;j++)

{

a[i][j]=0;

}

}

}

void visual()

{

int i,j;

printf (" 1 2 3\n");

for(i=1; i<=3; i++)

{

printf ("%d", i);

for (j=1;j<=3;j++)

{

if (a[i][j]==0){printf("| ");};

if (a[i][j]==1){printf("|O");};

if (a[i][j]==2){printf("|X");};

}

printf ("|\n");

printf ("\_\_\_\_\_\_\_\_\n");

}

}

void hod (int n)

{

int hi;

int sto;

int str;

int ver=0;

while (ver==0)

{

printf ("Your move. Enter the row and column number. \n");

scanf("%d", &hi);

sto=hi%10;

str=(hi-sto)/10;

if (a[str][sto]==0 && (hi==11 || hi==12 || hi==13 || hi==21 || hi==22 || hi==23 || hi==31 || hi==32 || hi==33 )) (ver=1);

}

if ( n==1) {a[str][sto]=1;}

if ( n==2) {a[str][sto]=2;}

}

int proverka()

{

int i;

int res;

res=0;

for (i=1; i<=3; i++)

{

if (a[i][1]==1 && a[i][2]==1 && a[i][3]==1 ) {res=1;}

}

for (i=1; i<=3; i++)

{

if (a[1][i]==1 && a[2][i]==1 && a[3][i]==1 ) {res=1;}

}

if (a[1][1]==1 && a[2][2]==1 && a[3][3]==1 ) {res=1;}

if (a[1][3]==1 && a[2][2]==1 && a[3][1]==1 ) {res=1;}

for (i=1; i<=3; i++)

{

if (a[i][1]==2 && a[i][2]==2 && a[i][3]==2 ) {res=2;}

}

for (i=1; i<=3; i++)

{

if (a[1][i]==2 && a[2][i]==2 && a[3][i]==2 ) {res=2;}

}

if (a[1][1]==2 && a[2][2]==2 && a[3][3]==2 ) {res=2;}

if (a[1][3]==2 && a[2][2]==2 && a[3][1]==2 ) {res=2;}

return res;

}

main()

{

int igra;

nul();

visual();

igra=0;

while (igra==0)

{

hod(1);

visual();

igra=proverka();

if (igra==0)

{

hod(2);

visual();

igra=proverka();

}

}

if (igra==1){printf ("The winner null\n");};

if (igra==2){printf ("THe winner cross\n");};

getch();

}