Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

Звіт

про виконання лабораторних та практичних робіт блоку № 5

На тему: «Файли. Бінарні Файли. Символи і Рядкові Змінні та Текстові Файли. Стандартна бібліотека та деталі/методи роботи з файлами. Створення й використання бібліотек.»

з дисципліни: «Основи програмування»

до:

ВНС Лабораторної Роботи № 6
ВНС Лабораторної Роботи № 8
ВНС Лабораторної Роботи № 9
Алготестер Лабораторної Роботи №4
Алготестер Лабораторної Роботи №6
Практичних Робіт до блоку №5

Виконав:

Студент групи ШІ-11

Мартин М.І

Львів 2024

Тема:

Файли. Бінарні Файли. Символи і Рядкові Змінні та Текстові Файли. Стандартна бібліотека та деталі/методи роботи з файлами. Створення й використання бібліотек.

Мета:

Навчитись працювати з файлами, записувати, приєднувати та читати у різних форматах. Використати їх для практичних застосувань

Теоретичні відомості:

1) Вивчив/знав:
    1. Файли та робота з ними
    2. Бінарний та текстовий запис даних у файли
    3. Створення бібліотек
2) Джерела:

    Всю інформацію до теоретичних відомостей я отримав на лекційних/практичних парах. Додатково використовував сайт

Виконання роботи:

1) Опрацювання завдання та вимог до програми та середовища

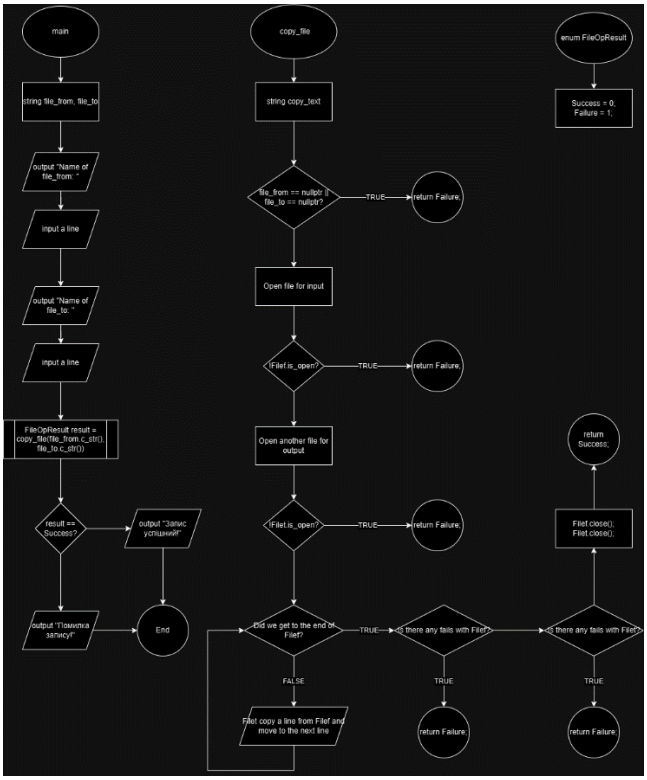
Завдання №1 Епік 5 - Практичне завдання

Реалізувати функцію створення файла і запису в нього даних:
enum FileOpResult { Success, Failure, … };
FileOpResult write_to_file(char *name, char *content);
Умови задачі:

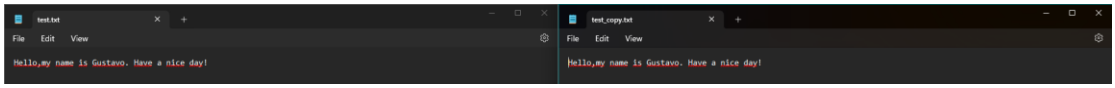
- створити файл із заданим ім’ям; якщо файл існує – перезаписати його вміст
- написати код стійкий до різних варіантів вхідних параметрів
- name – ім’я, може не включати шлях
- записати у файл вміст стрічки content, прочитати content із стандартного вводу
- копіювати вміст файла з ім’ям file_from у файл з ім’ям file_to; написати код стійкий до різних варіантів вхідних параметрів, обробити всі можливі варіанти відсутності одного з файлів
- file_from, file_to – можуть бути повним або відносним шляхом
- повернути статус операції: Success – все пройшло успішно, Failure – файл не вдалося створити, або збій операції відкриття, запису даних, чи закриття файла.

```cpp
#include <iostream>
#include <fstream>
#include <string>
using namespace std;

enum FileOrResult {Success,Failure};

FileOrResult write_to_file (const char* name, const char* content){
  if(name == nullptr || content == nullptr){
    return Failure;
  }

  ofstream outFile(name,ios::out);
  if(!outFile){
    return Failure;
  }

  outFile << content;
  if(!outFile.good()){
    return Failure;
  }

  outFile.close();
  if(!outFile.good()){
    return Failure;
  }

  outFile.close();
  return Success;
}

FileOrResult copy_file (const char* file_from, const char* file_to){
  if(file_from == nullptr || file_to == nullptr){
    cerr << "Error: Invalid parameters for copy_file." << endl;
    return Failure;
  }

  ifstream inFile(file_from,ios::in); //це об'єкт , який відповідає за читання даних з файлу, з якого ми копіюємо.
  if(!inFile){
    cerr << "Error: Unable to open source file: " << file_from << endl;
    return Failure;
  }
  ofstream outFile(file_to,ios::out);//це об'єкт , який відповідає за запис даних з файлу, з якого ми копіюємо.
  if(!outFile){
    cerr << "Error: Unable create source file: " << file_from << endl;
    inFile.close();
    return Failure;
  }

```

```cpp
    string line;
    while (getline(inFile,line)){
      outFile << line << endl;
      if(!outFile.good()){
        cerr << "Error: Failed to write to destination file: " << file_to << endl;
        inFile.close();
        outFile.close();
        return Failure;
      }
    }

    inFile.close();
    outFile.close();

    return Success;
  }

  int main (){
    const char* fileName = "test.txt";
    const char* fileCopy = "test_copy.txt";
    char content[256];

    cout << "Enter content to write to the file: ";
    cin.getline(content, sizeof(content));

    FileOrResult result = write_to_file(fileName,content);
    if (result == Success) {
      cout << "File successfully written to " << fileName << endl;
    } else {
      cout << "Failed to write to file: " << fileName << endl;
    }

    result = copy_file (fileName,fileCopy);
    if(result == Success){
      cout << "File successfully copied to " << fileCopy << endl;
    } else{
      cout << "Failed to copy file to " << fileCopy << endl;
    }

    return 0;
  }
```




```
Enter content to write to the file: Hello,my name is Gustavo. Have a nice day!
File successfully written to test.txt
File successfully copied to test_copy.txt
```



Завдання №2 внс лаб 6 варіант 11

Задано рядок, що складається із символів. Символи поєднуються в слова. Слова одне від одного відокремлюються одним або декількома пробілами. Наприкінці тексту ставиться крапка. Текст містить не більше 255 символів. Виконати ввід рядка, використовуючи функцію gets(s) і здійснити обробку рядка у відповідності зі своїм варіантом.

Перетворити рядок таким чином, щоб всі слова в ньому були надруковані навпаки.

```cpp
#include <iostream>
#include <cstring>
#include <algorithm>

using namespace std;
void reverseWord (string &s , size_t wordStart , size_t wordEnd){
  while (wordStart < wordEnd){
    swap (s[wordStart], s[wordEnd]);
    wordStart++;
    wordEnd--;
  }
}

int main (){
  string s;
  cout << "Enter text (only less than 255 and end with '.' ) :";
  getline(cin,s);

  size_t len = s.length();
  size_t wordStart = 0;

  for (size_t i = 0 ; i <= len;i++){
    if (i == len || s[i] == '.' || s[i] == ' '){
      if(i > wordStart){
      reverseWord (s, wordStart, i-1);
      }
      wordStart = i+1;
    }
  }
  cout << "Result: " << s << endl;
  return 0;
}
```

```
Enter text (only less than 255 and end with '.' ) :lucky
Result: ykcul
PS C:\Users\Maks\Documents\ai_programming_playground_2024\ai_11\maksym_martyn\epic_5>
```

Завдання №3 вис лаб 8 завдання 11

Сформувати двійковий файл із елементів, заданої у варіанті структури, роздрукувати його вмістиме, виконати знищення й додавання елементів у відповідності зі своїм варіантом, використовуючи для пошуку елементів що знищуються чи додаються, функцію. Формування, друк, додавання й знищення елементів оформити у вигляді функцій. Передбачити повідомлення про помилки при відкритті файлу й виконанні операцій вводу/виводу.

11. Структура "Відеокасета":

- назва фільму;

- режисер;

- тривалість;

- ціна.

Знищити всі елементи із ціною вищою заданої, додати 3 елементи в кінець

файлу.

```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
using namespace std;

struct Videotape{
  string title;
  string director;
  int duration;
  double price;
};


bool isValidDouble(const string& s) {
  if (s.empty()) return false;
  char* end = nullptr;
  strtod(s.c_str(), &end);
  return end == s.c_str() + s.size();
}

void addVideotape (const string& filename){
  Videotape tape;
  cout << "Enter film name : ";
  cin.ignore();
  getline(cin,tape.title);

  cout << "Enter director name : ";
  getline(cin,tape.director);
  cout << "Enter duration in minutes: ";
  cin >> tape.duration;
  cout << "Enter price : ";
  cin >> tape.price;

  ofstream file (filename , ios::app);
  if(file){
    file << tape.title << '|' << tape.director << '|' << tape.duration << '|' << tape.price << endl;
    file.close();
    cout << "Videotape was added" << endl;
  } else {
    cerr << "Error with opening file";
  }
}

void printVideotapes (const string& filename){
  ifstream file(filename);

  if(file){
    string line;
    cout << "List of Videotapes: " << endl;

    while(getline(file,line)){
      if (line.empty()) continue;

      size_t position = 0;
      Videotape tape;

      position = line.find("|");
      if (position == string::npos) continue;
      tape.title = line.substr(0,position);
      line.erase(0,position+1);

      position = line.find("|");
      if (position == string::npos) continue;
      tape.director = line.substr(0,position);
      line.erase(0,position+1);

      position = line.find("|");
      if (position == string::npos) continue;
      tape.duration = stoi(line.substr(0,position));
      line.erase(0,position+1);

      if (!isValidDouble(line)) {
        cerr << "Invalid price format. Skipping line: " << line << endl;
        continue;
      }

      tape.price = stod(line);

      cout << "Title: " << tape.title << "\n"
      << "Director: " << tape.director << "\n"
      << "Duration: " << tape.duration << " min\n"
      << "Price: " << tape.price << " UAH\n"
      << "---------------------------------" << endl;
    }
    file.close();
  } else {
    cerr << "Error with opening file for reading.";
  }
}

void deleteExpensiveTapes(const string& filename , double maxPrice){
  ifstream file(filename);
    if(!file){
      cerr << "Error with opening file";
      return;
    }

  vector<Videotape> tapes;
  string line;

  while(getline(file,line)){
    if (line.empty()) continue;

      size_t position = 0;
      Videotape tape;

      position = line.find("|");
      if (position == string::npos) continue;
      tape.title = line.substr(0,position); //стирае
      line.erase(0,position+1);

      position = line.find("|");
      if (position == string::npos) continue;
      tape.director = line.substr(0,position);
      line.erase(0,position+1);

      position = line.find("|");
      if (position == string::npos) continue;
      tape.duration = stoi(line.substr(0,position)); // перетворює текст на цифри
      line.erase(0,position+1);

      if (!isValidDouble(line)) {
        cerr << "Invalid price format. Skipping line: " << line << endl;
        continue;
      }

      tape.price = stod(line);

```

```
    if (tape.price <= maxPrice){
        tapes.push_back(tape);
    }
  }
  file.close();

  ofstream overwritingFile (filename, ios::trunc);
  if(!overwritingFile){
    cerr << "Error with opening file";
    return;
  }

  for (size_t i = 0; i < tapes.size(); ++i) {
    overwritingFile << tapes[i].title << "|" << tapes[i].director << "|" << tapes[i].duration << "|" << tapes[i].price << endl;
  }

  overwritingFile.close();

  cout << "Videotapes priced higher than " << maxPrice << " UAH have been removed.";
}

void processVideotapes(const string& filename, double maxPrice) {
  deleteExpensiveTapes(filename, maxPrice);

  cout << "\nAdding three new videotapes:\n";
  for (int i = 0; i < 3; ++i) {
  cout << "\nEnter details for new videotape #" << (i + 1) << ":\n";
  addVideotape(filename);
 }
}
```

```
int main() {
 const string filename = "videotapes.txt";
 int choice;
 double maxPrice;

 do {
  cout << "\nMenu:\n";
  cout << "1. Add a videotape\n";
  cout << "2. Show all videotapes\n";
  cout << "3. Delete videotapes with price above a specified value and add three new ones\n";
  cout << "4. Exit\n";
  cout << "Your choice: ";
  cin >> choice;
  cin.ignore();

  switch (choice) {
   case 1:
    addVideotape(filename);
    break;

   case 2:
    printVideotapes(filename);
    break;

   case 3:
    cout << "Enter the maximum price to filter videotapes: ";
    cin >> maxPrice;
    processVideotapes(filename, maxPrice);
    break;

   case 4:
    cout << "Exiting the program..." << endl;
    break;

   default:
    cerr << "Invalid choice! Please try again." << endl;
  }
 } while (choice != 4);

 return 0;
}
```

```
Menu:
1. Add a videotape
2. Show all videotapes
3. Delete videotapes with price above a specified value and add three new ones
4. Exit
Your choice: 1
Enter film name : Dexter
Enter director name : Fridon Morgan
Enter duration in minutes: 123
Enter price : 3000
Videotape was added

Menu:
1. Add a videotape
2. Show all videotapes
3. Delete videotapes with price above a specified value and add three new ones
4. Exit
Your choice: 3
Enter the maximum price to filter videotapes: 30
Videotapes priced higher than 30 UAH have been removed.
```

```
Adding three new videotapes:

Enter details for new videotape #1:
Enter film name : Falk
Enter director name : Name
Enter duration in minutes: 100
Enter price : 300
Videotape was added

Enter details for new videotape #2:
Enter film name : test
Enter director name : tester
Enter duration in minutes: 30302
Enter price : 10
Videotape was added

Enter details for new videotape #3:
Enter film name : Negfa
Enter director name : Kano
Enter duration in minutes: 30
Enter price : 1000
Videotape was added
```

```
Menu:
1. Add a videotape
2. Show all videotapes
3. Delete videotapes with price above a specified value and add three new ones
4. Exit
Your choice: 2
List of Videotapes:
Title: Falk
Director: Name
Duration: 100 min
Price: 300 UAH
---------------------------------
Title: test
Director: tester
Duration: 30302 min
Price: 10 UAH
---------------------------------
Title: Negfa
Director: Kano
Duration: 30 min
Price: 1000 UAH
---------------------------------

Menu:
1. Add a videotape
2. Show all videotapes
3. Delete videotapes with price above a specified value and add three new ones
4. Exit
Your choice: 4
Exiting the program...
```

Завдання №4 внс лаб 9 завдання 11

Створити текстовий файл F1 не менше, ніж з 10 рядків і записати в нього інформацію , виконати завдання:

1) Скопіювати з файлу F1 у файл F2 всі рядки, крім того рядка, що містить найкоротше слово.

2) Надрукувати номер цього рядка.

```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <sstream>
#include <climits>
#include <vector>

using namespace std;

size_t findShortestWordLength(const string& line) {
    size_t minLength = INT_MAX;
    size_t currentLength = 0;

    for (char ch : line) {
        if (ch == ' ' || ch == '\n') {
            if (currentLength > 0 && currentLength < minLength) {
                minLength = currentLength;
            }
            currentLength = 0;
        } else {
            currentLength++;
        }
    }

    if (currentLength > 0 && currentLength < minLength) {
        minLength = currentLength;
    }

    return minLength;
}

int main() {
    const string fileF1 = "F1.txt";
    const string fileF2 = "F2.txt";
    vector<string> lines = {
    "Welcome to the programming world.",
    "C++ is powerful programming language.",
    "Short.",
    "This line contains a very small word.",
    "Here is another example of a line.",
    "A quick brown fox jumps over the lazy dog.",
    "This line is bit longer for testing purposes.",
    "An example with mixed word lengths.",
    "Tiny.",
    "This is the last line of this test file."
};

    ofstream outFileF1(fileF1);  // створення файлу F1 і запис даних
    if (!outFileF1) {
        cerr << "Error creating file F1!" << endl;
        return 1;
    }
    for (const auto& line : lines) {
        outFileF1 << line << endl;
    }
    outFileF1.close();

    ifstream inFileF1(fileF1);
    if (!inFileF1) {
        cerr << "Error opening file F1 for reading!" << endl;
        return 1;
    }

    string line;
    size_t minWordLength = INT_MAX;
    int shortestLineNumber = -1;
    int currentLineNumber = 0;

    vector<string> fileContent;
    while (getline(inFileF1, line)) {
        fileContent.push_back(line);
        currentLineNumber++;

        size_t currentMinLength = findShortestWordLength(line);
        if (currentMinLength < minWordLength) {
            minWordLength = currentMinLength;
            shortestLineNumber = currentLineNumber;
        }
    }
    inFileF1.close();

    ofstream outFileF2(fileF2);  //створення файлу F2 і запис даних окрім РЯДКА З НАЙКОРОТШИМ СЛОВОМ
    if (!outFileF2) {
        cerr << "Error creating file F2!" << endl;
        return 1;
    }

    currentLineNumber = 0;
    for (const auto& line : fileContent) {
        currentLineNumber++;
        if (currentLineNumber != shortestLineNumber) {
            outFileF2 << line << endl;
        }
    }
    outFileF2.close();

    cout << "The line number with the shortest word is: " << shortestLineNumber << endl;

    return 0;
}
```

```
The line number with the shortest word is: 4
PS C:\Users\Maks\Documents\ai_programming_playground_2024\ai_11\maksym_martyn\epic_5>
```

Завдання №5 алготестер лаб 4v3

Вам дано масив, який складається з N додатніх цілих чисел.
Ваше завдання - розділити його на три частини, по остачі від ділення на 3, по зростанню остачі (тобто спочатку йдуть числа, у яких остача 0, далі числа з остачею 1 і тоді нарешті числа з остачею 2).

Далі необхідно ті елементи, остача від ділення на 3 яких парна посортувати по зростанню, а ті, у яких остача 1 - по спаданню.

Після цього видаліть усі дублікати з масиву.

Виведіть результуючий масив.

1) Використовуючи stl :

```cpp
// Використання алгоритму STL

#include <iostream>
#include <set>
#include <algorithm>

using namespace std;

int main(){
  int N;
  cin >> N;
  if (N<1 || N>1000){
    cerr << "Invalid N! Please restart program and try again";
    return 0;
  }
  int* arr = new int [N];

  for(int i = 0; i < N; i++){
    cin >> arr[i];
    if(arr[i]<0 || arr[i]>1000){
      cerr << "Invalid ai! Please restart program and try again";
      delete[] arr;
      return 0;
    }
  }

  int* group0 = new int[N];
  int* group1 = new int[N];
  int* group2 = new int[N];
  int size0 = 0, size1 = 0, size2 = 0;

  for (int i = 0; i < N; i++) {
    if (arr[i] % 3 == 0) {
      group0[size0++] = arr[i];
    } else if (arr[i] % 3 == 1) {
      group1[size1++] = arr[i];
    } else {
      group2[size2++] = arr[i];
    }
  }
  sort(group0, group0 + size0); //сортує у порядку зростання
  sort(group1, group1 + size1, greater<int>()); //у порядку спадання
  sort(group2, group2 + size2); //у порядку зростання

```

```cpp
  int* result = new int[N];
  int result_size = 0;

  auto add_unique = [&](int* group, int size) {
    for (int i = 0; i < size; i++) {
      bool exists = false;
      for (int j = 0; j < result_size; j++) {
        if (result[j] == group[i]) {
        exists = true;
        break;
        }
      }
      if (!exists) {
        result[result_size++] = group[i];
      }
    }
  };

  add_unique(group0, size0);
  add_unique(group1, size1);
  add_unique(group2, size2);

  cout << result_size << endl;
  for (int i = 0; i < result_size; i++) {
    cout << result[i] << " ";
  }
  cout << endl;

  delete[] arr;
  delete[] group0;
  delete[] group1;
  delete[] group2;
  delete[] result;

  return 0;
}
```

2) Використання алгоритму Merge Sort

```cpp
// Використання алгоритму Merge Sort

#include <iostream>
#include <set>
#include <algorithm>

using namespace std;

void merge(int* arr, int left, int mid, int right, bool ascending) {
  int n1 = mid - left + 1; // Розмір лівої частини
  int n2 = right - mid; // Розмір правої частини

  int* L = new int[n1];
  int* R = new int[n2];

  for (int i = 0; i < n1; i++){
    L[i] = arr[left + i];
  }
  for (int j = 0; j < n2; j++){
    R[j] = arr[mid + 1 + j];
  }

  int i = 0, j = 0;
  int k = left;

  while(i < n1 && j < n2){ //поки містять елементи
    bool condition = ascending ? (L[i] <= R[j]) : (L[i] >= R[j]);
    if (condition) {
     arr[k++] = L[i++];//arr – це вихідний масив, у який ми записуємо злиті дані.
    } else {
     arr[k++] = R[j++];
    }
  }

  while (i < n1)
  arr[k++] = L[i++];

  while (j < n2)
  arr[k++] = R[j++];

  delete[] L;
  delete[] R;
}

void mergeSort(int* arr, int left, int right, bool ascending) {
  if (left < right) {
    int mid = left + (right - left) / 2;

    mergeSort(arr, left, mid, ascending);          // Сортуємо ліву частину
    mergeSort(arr, mid + 1, right, ascending);     // Сортуємо праву частину
    merge(arr, left, mid, right, ascending);       // Зливаємо частини
  }
}
```

```cpp
void addUnique(int* group, int group_size, int* result, int& result_size) {
  for (int i = 0; i < group_size; i++) {
    bool exists = false;
    for (int j = 0; j < result_size; j++) {
      if (result[j] == group[i]) {
        exists = true;
        break;
      }
    }
    if (!exists) {
      result[result_size++] = group[i];
    }
  }
}
int main(){
  int N;
  cin >> N;

  if (N<1 || N>1000){
    cerr << "Invalid N! Please restart program and try again";
    return 0;
  }

  int* arr = new int [N];

  for(int i = 0; i < N; i++){
    cin >> arr[i];
    if(arr[i]<0 || arr[i]>1000){
      cerr << "Invalid ai! Please restart program and try again";
      delete[] arr;
      return 0;
    }
  }

  int* group0 = new int[N];
  int* group1 = new int[N];
  int* group2 = new int[N];
  int size0 = 0, size1 = 0, size2 = 0;

  for (int i = 0; i < N; i++) {
    if (arr[i] % 3 == 0) {
      group0[size0++] = arr[i];
    } else if (arr[i] % 3 == 1) {
      group1[size1++] = arr[i];
    } else {
      group2[size2++] = arr[i];
    }
  }
```

```
  mergeSort(group0, 0, size0 - 1, true);    // Зростання
  mergeSort(group1, 0, size1 - 1, false);  // Спадання
  mergeSort(group2, 0, size2 - 1, true);   // Зростання

  int* result = new int[N];
  int result_size = 0;

  addUnique(group0, size0, result, result_size);
  addUnique(group1, size1, result, result_size);
  addUnique(group2, size2, result, result_size);

  cout << result_size << endl;
  for (int i = 0; i < result_size; i++) {
    cout << result[i] << " ";
  }
  cout << endl;

  delete[] arr;
  delete[] group0;
  delete[] group1;
  delete[] group2;
  delete[] result;

  return 0;
}
```

```
10
1 33 4 8 6 5 2 7 5 0
9
0 1 2 4 5 6 7 8 33
```

Created	Compiler	Result	Time (sec.)	Memory (MiB)	Actions
a few seconds ago	C++ 23	Accepted	0.003	1.359	View
an hour ago	C++ 23	Accepted	0.003	1.402	View

Завдання №6 алготестер лаб 6v3

У Клінта в черговий раз виключилось світло і йому немає чим зайнятися. Так як навіть це не заставить його подивитися збережені відео про програмування на ютубі - він вирішив придумати свою гру на основі судоку.

Гра виглядає так:

Є поле розміром N×N, в якому частина клітинок заповнена цифрами, а частина клітинок пусті (позначаються нулем). Також у нього є Q пар координат X та Y.
Завданням гри є написати до кожної координати скільки чисел туди можна вписати (якщо вона пуста) і які це числа (обов’язково в посортовані по зростанню!). В клітинку можна вписати лише ті числа, які не зустрічаються в рядку та стовбці, які перетинаються у цій клітинці.

Під час гри поле не міняється!

Також необовязково, щоб це було валідне судоку! Якщо є клітинка, в яку не можна вписати ніяку цифру - виведіть 0.

Також допускаються рядки та стовпці, в яких цифра записана кілька разів.

```cpp
#include <iostream>
#include <vector>
#include <string>

using namespace std;

int main() {
  int N;
  cin >> N;

  if (N < 1 || N > 9) {
    cerr << "Error: N must be between 1 and 9." << endl;
    return 1;
  }

  vector<vector<int>> arr(N, vector<int>(N));

  for (int i = 0; i < N; i++) {
    string str;
    cin >> str;

    if (str.length() != N) {
      cerr << "Error: Each row must have exactly N characters." << endl;
      return 1;
    }

    for (int j = 0; j < N; j++) {
      arr[i][j] = str[j] - '0';

      if (arr[i][j] < 0 || arr[i][j] > 9) {
        cerr << "Error: Each cell must contain a digit between 0 and 9." << endl;
        return 1;
      }
    }
  }

  int Q;
  cin >> Q;

  if (Q < 1 || Q > 1000) {
    cerr << "Error: Q must be between 1 and 1000." << endl;
    return 1;
  }

  vector<pair<int, int>> queries(Q);

  for (int i = 0; i < Q; i++) {
    int x, y;
    cin >> x >> y;

    if (x < 1 || x > N || y < 1 || y > N) {
      cerr << "Error: Query coordinates must be between 1 and N." << endl;
      return 1;
    }

    queries[i] = {x - 1, y - 1};
  }


  for (int i = 0; i < queries.size(); i++) {
    int x = queries[i].first; //
    int y = queries[i].second;

    if (arr[x][y] != 0) {
      cout << "1\n" << arr[x][y] << "\n";
      continue;
    }

    vector<bool> possible(N + 1, true);

    // Видалення чисел, які є в рядку x
    for (int j = 0; j < N; j++) {
      if (arr[x][j] != 0) {
        possible[arr[x][j]] = false;
      }
    }

    // Видалення чисел, які є в стовпці y
    for (int i = 0; i < N; i++) {
      if (arr[i][y] != 0) {
        possible[arr[i][y]] = false;
      }
    }

    vector<int> result;
    for (int i = 1; i <= N; i++) {
      if (possible[i]) {
        result.push_back(i);
      }
    }

    cout << result.size() << "\n";
    for (int num : result) {
      cout << num << " ";
    }
    cout << "\n";
  }

  return 0;
}
```

```
3
123
505
210
3
1 1
2 2
3 3
1
1
1
3
0
```

Завдання №7 self practice

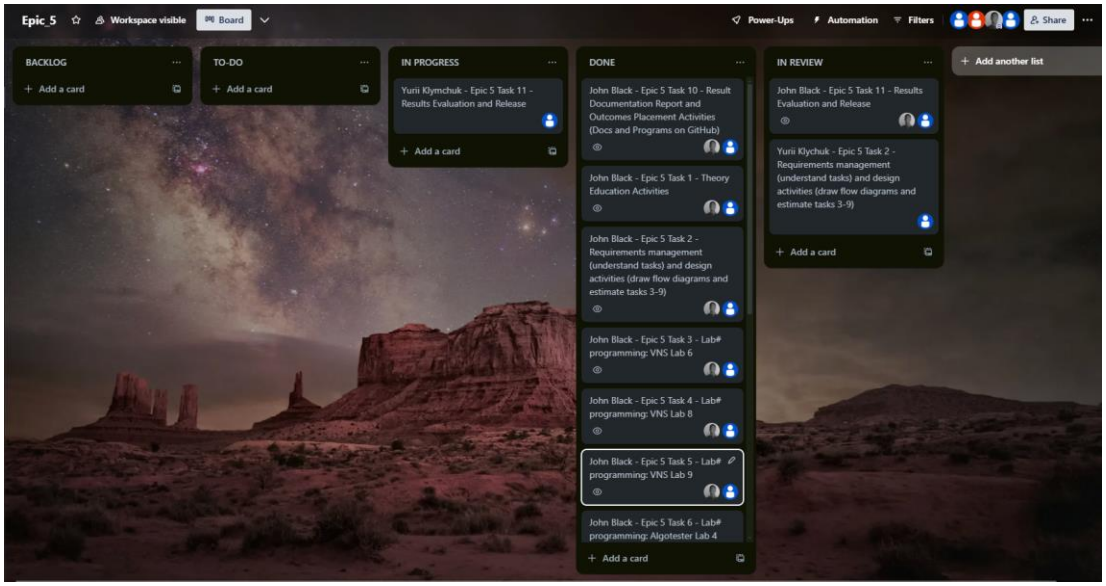
Створити програму, яка визначає добуток елементів заданого рядка

```cpp
void output_matrix(int**matrix,int rows, int columns)
{
  for (size_t i = 0; i < rows; i++)
  {
    for (size_t j = 0; j < columns; j++)
    {
      std::cout<<matrix[i][j]<<'\t';
    }
    std::cout<<'\n';
  }
}

int product_numbers(int**matrix,int columns,int us_row_num)
{
  int pr = 1;
  for (size_t j = 0; j < columns; j++)
  {
    pr *= matrix[us_row_num][j];
  }
  return pr;
}


int main()
{
  int** matrix;
  int r,c,user_number;
  cout<<"Enter number of rows then columns ";
  cin>>r>>c;
  cout<<'\n';
  matrix=create_matrix(r,c);
  input_matrix(matrix,r,c);
  output_matrix(matrix,r,c);
  cout<<"Enter number row: ";
  cin>>user_number;
  cout<<"Product of elements in row "<<user_number<<'='<<product_numbers(matrix,c,user_number-1);
  delete_matrix(matrix,r,c);
  return 0;
}
```

```cpp
//створити функцію яка визначає добуток елементів заданого рядка
#include <iostream>
using namespace std;
int** create_matrix(int rows, int columns)
{
  int**matrix=nullptr;
  matrix = new int*[rows];
  for (size_t i = 0; i < rows; i++)
  {
    matrix[i]=new int[columns];
  }
  return matrix;
}


void delete_matrix(int**matrix, int rows , int columns)
{
  for (size_t i = 0; i < rows; i++)
  {
    delete[] matrix[i];
  }
  delete[] matrix;
}


void input_matrix(int** matrix,int rows, int columns)
{
  for (size_t i = 0; i < rows; i++)
  {
    for (size_t j = 0; j < columns; j++)
    {
      cout<<'['<<i+1<<"]["<<j+1<<"] = ";
      cin>>matrix[i][j];
    }
  }
}
```

```
[1][1] = 1
[1][2] = 2
[1][3] = 3
[2][1] = 4
[2][2] = 5
[2][3] = 6
[3][1] = 7
[3][2] = 8
[3][3] = 9
1       2       3
4       5       6
7       8       9
Enter number row: 2
Product of elements in row 2=120
```

Робота з командою:



Висновок:

Навчився працювати з файлами, записувати, приєднувати та читати у різних форматах. Використати їх для практичних застосувань.