НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО»

ЗВІТ

З лабораторної роботи №1

з навчальної дисципліни

«Програмцвання алгоритмічниї структур»

Варіант № 7

Тема:

СКЛАДНІСТЬ АЛГОРИТМУ

Студентки 1-го курсу НН ІАТЕ гр. ТР-22

Грищук Анжеліка

Лабораторну роботу захищено

З оцінкою\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КИЇВ 2023

**Мета:** Ознайомлення з оцінкою складності алгоритмів та її характеристики.

**Теоретична частина**

**Алгоритм** – це скінченна послідовність команд, які треба виконати над вхідними даними для отримання результату.

Основними мірами обчислювальною складності алгоритмів є:

Складність алгоритму описується функцією f(n), де n – розмір вхідних даних. Важливе теоретичне ш практичне значення має класифікація алгоритмів, яка бере до уваги швидкість зростання цієї функції.

* часова складність, яка характеризує час, необхідний для виконання алгоритму на даній машині; цей час, як правило, визначається кількістю операцій, які потрібно виконати для реалізації алгоритму;
* ємнісна складність, яка характеризує пам’ять, необхідну для виконання алгоритму.

Часова та ємнісна складність тісно пов’язані між собою. Обидві є функціями від розміру вхідних даних.

**Код програми:**

//Варіант №7

#include <cstdlib> // містить "srand()" і "rand()"

#include <iostream>

#include <chrono> // Для вирахування часу

#include <windows.h> // бібліотека для зміни кольору тексту в консолі

using namespace std;

int main() {

srand(time(0)); //генератор рандомних чисел

int task;

std::cout << "Enter a number of the task you want to see(1-2)-->\n\t";

cin >> task;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if (task == 1) {

int number;

std::cout << "Enter the array size: \n\t\t 1. 10x10\n\t\t 2. 50x50\n\t\t 3. 100x100\n\t\t 4. 500x500 \n\t";

cin >> number;

int n1, n2;

switch (number) {

case 1:

n1 = 10;

n2 = 10;

std::cout << "Generating 10x10 array for task 1...\n" << endl;

break;

case 2:

n1 = 50;

n2 = 50;

std::cout << "Generating 50x50 array for task 1...\n" << endl;

break;

case 3:

n1 = 100;

n2 = 100;

std::cout << "Generating 100x100 array for task 1...\n" << endl;

break;

case 4:

n1 = 500;

n2 = 500;

std::cout << "Generating 500x500 array for task 1...\n" << endl;

break;

default:

std::cout << "you entered the wrong number, enter it again !" << endl;

return 0;

}

// Створення двовимірного масиву

int sum = 0;

int row\_sum = 0;

int\*\* array = new int\* [n1];

for (int i = 0; i < n1; i++)

{

array[i] = new int[n2];

for (int j = 0; j < n2; j++)

{

array[i][j] = rand() % 31 - 10;

std::cout.width(3); // Структурування масиву

std::cout << array[i][j] << " ";

}

std::cout << endl;

}

// Пошук суми елементів першого стовпчика

for (int i = 0; i < n1; i++)

{

sum += array[i][0]; //рахуємо суму елементів 1-го стовпчика

}

// Збільшення елементів на 5 відповідно до умови

for (int i = 0; i < n1; i++)

{

for (int j = 0; j < n2; j++)

{

row\_sum += array[i][j];

}

if (row\_sum < sum)

{

for (int j = 0; j < n2; j++)

{

array[i][j] += 5;

row\_sum++;

}

std::cout << "lene " << i + 1 << " it was noticed" << endl;

}

}

std::cout << endl << "Redacted...\n" << endl;

// Виведення масиву після модифікації

auto start = std::chrono::high\_resolution\_clock::now();

int iterations = 0;

/\*clock\_t start\_time2 = clock();\*/

for (int i = 0; i < n1; i++)

{

for (int j = 0; j < n2; j++)

{

std::cout.width(3); // Структурування масиву

std::cout << array[i][j] << " ";

iterations++;

}

std::cout << endl;

}

std::cout << "\nNumber of iterations: " << iterations << endl;

auto end = std::chrono::high\_resolution\_clock::now();

auto duration = std::chrono::duration\_cast<std::chrono::nanoseconds>(end - start).count();

std::cout << "Execution time: " << duration << " ns" << std::endl;

// Звільняємо пам'ять

for (int i = 0; i < n1; i++) {

delete[] array[i];

}

delete[] array;

return 0;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if (task == 2) {

int number;

std::cout << "Enter the array size: \n\t\t 1. 10x10\n\t\t 2. 50x50\n\t\t 3. 100x100\n\t\t 4. 500x500 \n\t";

cin >> number;

int n3, n4;

switch (number) {

case 1:

n3 = 10;

n4 = 10;

std::cout << "Generating 10x10 array for task 1...\n" << endl;

break;

case 2:

n3 = 50;

n4 = 50;

std::cout << "Generating 50x50 array for task 1...\n" << endl;

break;

case 3:

n3 = 100;

n4 = 100;

std::cout << "Generating 100x100 array for task 1...\n" << endl;

break;

case 4:

n3 = 500;

n4 = 500;

std::cout << "Generating 500x500 array for task 1...\n" << endl;

break;

default:

std::cout << "you entered the wrong number, enter it again !" << endl;

return 0;

}

// Створення двовимірного масиву

int\*\* array1 = new int\* [n3];

for (int i = 0; i < n3; i++)

{

array1[i] = new int[n4];

for (int j = 0; j < n4; j++)

{

if ((j < n3 / 2) && (i < n3) && ((j + 1 + i) / n3) || (j > i - 1) && (j < n4 - j - 1)) {

cout << "\033[38;2;255;20;147m"; // рожевий колір

}

if ((j < n3 - i) && (j > n4 - j - 1) || (j < i + 1) && (j > n4 - j - 1)) {

cout << "\033[32m"; // зелений колір

}

array1[i][j] = rand() % 31 - 10;

std::cout.width(3); // Структурування масиву

std::cout << array1[i][j] << " ";

cout << "\033[0m"; // відновлюємо колір за замовчуванням

}

std::cout << endl;

}

// обмін верхніх елементів масиву місцями

for (int i = 0; i < n3; i++) {

for (int j = 0; j < n4; j++) {

if ((j > i - 1) && (j < n4 - j - 1)) {

int temp = array1[i][j];

array1[i][j] = array1[i][n3 - j - 1];

array1[i][n3 - j - 1] = temp;

}

}

}

// обмін нижніх елементів масиву місцями

for (int i = 0; i < n3; i++) {

for (int j = 0; j < n4; j++) {

if ((j < n3 / 2) && (i < n3) && ((j + 1 + i) / n3)) {

int temp1 = array1[i][j];

array1[i][j] = array1[i][n3 - j - 1];

array1[i][n3 - j - 1] = temp1;

}

}

}

//Відредагований масив

std::cout << endl << "Redacted...\n" << endl;

// Виведення масиву після модифікації

int iterations1 = 0;

auto start = std::chrono::high\_resolution\_clock::now();

for (int i = 0; i < n3; i++)

{

for (int j = 0; j < n4; j++)

{

if ((j < n3 / 2) && (i < n3) && ((j + 1 + i) / n3) || (j > i - 1) && (j < n4 - j - 1)) {

cout << "\033[32m"; // зелений колір

}

if ((j < n3 - i) && (j > n4 - j - 1) || (j < i + 1) && (j > n4 - j - 0.5)) {

cout << "\033[38;2;255;20;147m"; // рожевий колір

}

std::cout.width(3); // Структурування масиву

std::cout << array1[i][j] << " ";

iterations1++;

cout << "\033[0m"; // відновлюємо колір за замовчуванням

}

std::cout << endl;

}

auto end = std::chrono::high\_resolution\_clock::now();

std::cout << "\nNumber of iterations: " << iterations1 << endl;

// Замір часу

auto duration = std::chrono::duration\_cast<std::chrono::nanoseconds>(end - start).count();

std::cout << "Execution time: " << duration << " ns" << std::endl;

// Звільняємо пам'ять

for (int i = 0; i < n3; i++) {

delete[] array1[i];

}

delete[] array1;

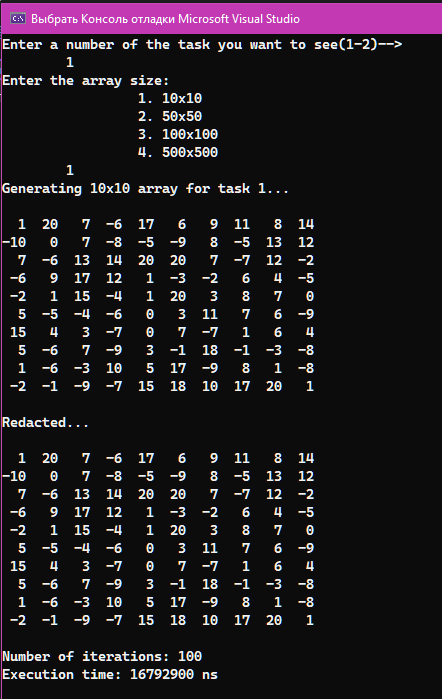
return 0;

}

}

**Результати виконання**

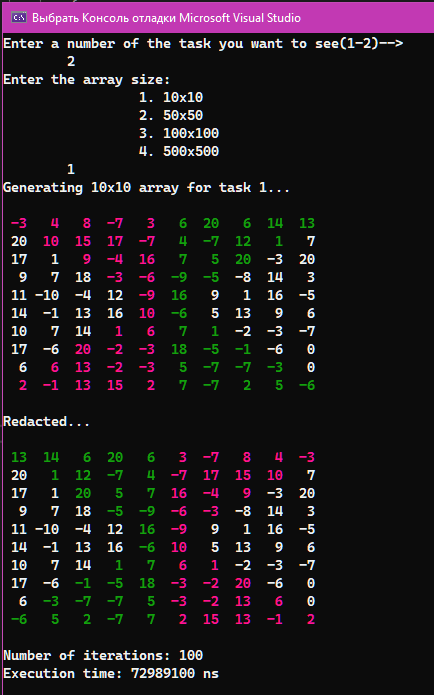
**Скріншоти виконання 1 завдання**



**Складність алгоритму в О-нотації**

|  |  |  |
| --- | --- | --- |
| **Розмір матриці** | **Кількість кроків** | **Час виконання (нс)** |
| **10** | **100** | **16792900** |
| **50** | **2500** | **303437600** |
| **100** | **10000** | **1094294400** |
| **500** | **250000** | **26796494800** |

**Скріншоти виконання 2 завдання**



**Складність алгоритму в О-нотації**

|  |  |  |
| --- | --- | --- |
| **Розмір матриці** | **Кількість кроків** | **Час виконання (нс)** |
| **10** | **100** | **72989100** |
| **50** | **2500** | **441229400** |
| **100** | **10000** | **1442295100** |
| **500** | **250000** | **34839878000** |

**Висновок.** Під час виконання лабораторної роботи, я отримала практичні навички визначення часової складності алгоритму, дізналася про основні класи алгоритмів, та навчилася будувати алгоритми з мінімальною часовою складністю для вирішення поставлених задач.