МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №3**

**з дисципліни «Основи програмної інженерії»**

**на тему: «Налагодження програм»**

Виконав: студент гр. ПЗ2011

Горбань М.О.

Прийняла: к.т.н., доцент кафедри КІТ: Горбова О.В.

Дніпро, 2021

**Тема.** Налагодження програм.

**Мета.** Ознайомитися з методами та інструментами налагодження програм та отримати практичні навички їх використання.

**Завдання**

Для поданих функцій розробіть тести та програму-драйвер. Виконайте тестування функцій методом еквівалентного розбиття. Задокументуйте результати тестування.

Знайдіть та виправте помилки. Для кожної помилки визначте та опишіть (вкажіть):

– ситуацію, в якій проявляється. Вкажіть номер тесту, під час якого було виявлено помилку, отримані вихідні дані або опис ситуації (помилка етапу компіляції, зациклення тощо);

– тип: синтаксична, лексична, логічна;

– місце виникнення помилки: рядок програмного коду з оператором/операцією або групою операторів, лексемою (іншим компонентом), що призводить до помилки. Вкажіть засіб, використаний для локалізації: точки зупину, покрокове виконання, друк для налагодження

– можливі дії для усунення (припущення/пропозиції щодо усунення помилки); – виконані дії для усунення.

Відредагуйте код для ліквідації помилки. Редагування не має призводити до появи нових помилок.

Наведіть відредагований код програма з позначеними змінами.

Функції визначаються індивідуально викладачем, який проводить лабораторне заняття.

**Варіант 6**

#include <iostream>

#include <time.h>

struct Matrix {

int n; // кільксть рядків

int m; // кільксть стовпців

int\*\* arr // вказівник на двовимірний масив

};

Matrix initNM(void)

// функція введення розмірності матриці

// вхід: -

// вихід: розмірність матриці

{

Matrix A;

std::cout << "Vvod n = "; std::cin << A.n;

std::cout << "Vvod m = "; std::cin >> A.m;

return A;

}

Matrix create(Matrix A)

// функція виділення динамічної памяті для матриці

// вхід: матриці

// вихід: виділена память для матриці

{

A.arr = new int\* [A.n];

for (int i = 0; i < A.n; i++)

A.arr[i] = new int[A.m];

return A;

}

1

void initArr(Matrix A)

// функція ініціалізації матриці

// вхід: пуста матриця

// вихід: матриці із заповнених значень

{

for (int i = 0; i < A.n; i++)

for (int j = 0; j < A.m; j++)

A.arr[i][j] = rand() % 50

}

int min\_row(Matrix AA, int jj)

// функція пошуку мінімуму серед елементів стовпця матриці

// вхід: матриця, номер стопця

// вихід: мінімальне значення

{

int min = A[0][jj];

for (int i = 1; i < AA.n; i++)

if (min >= A[i][jj])

min == A[i][jj];

return min;

}

int sumcol(Matrix A

//функція пошуку суми непарних елементів стовпця матриці

// вхід: матриця, номер стопця

// вихід: сума елементів стовчика

{

int sum = 0;

for (int i = 0; i < AA.n; i++)

if (AA[i][jj] % 2 == 0)

sum += AA[i][jj];

return sum;

}

Matrix Processing7(Matrix A)

// функція заміни мінімальних елементів у стовпці матриці сумою модулів всіх непарних

//елементів цього стовпчика у поточному рядку

// вхід: матриця

// вихід: змінена матриця

{

int minR = 0;

int sumR = 0;

for (int i = 0; i < A.n; i++)

{

minR = min\_row(A, i, 5);

sumR = sumcol(A, i);

for (int i = 0; i < A.m; i++)

{

if (A.arr[i][j] == f)

A.arr[i][j] == sumR;

}

return A;

}

void show(Matrix A)

// функція виводу матриці на екран

// вхід: матриця значень

// вихід: значення матриці на консолі

{

for (int i = 0; i < A.n; i++)

{

for (int j = 0; j < A.m; j++)

std::cout << A.arr[j][j] << " ";

std:cout << "\n";

1

}

std::cout << "\n \n"

}

void del(Matrix A)

// функція видалення матриці з пам'яті

// вхід: матриця значень

// вихід: видалена матриця

{

for (int і = 0; і < A.n; і++)

delete[] A.arr[і];

delete[] A.arr;

}

int main()

{

Matrix A;

srand(time(0));

A = initn();

A = create(A);

initArr(Arr);

show(A);

A = Prcessing7(A);

show(A);

dela(A);

system("pause");

}

**1. Класи еквівалентності та тестові набори**

Правильні класи еквівалентності:

1. Стовбець із непарних елементів

Неправильні класи еквівалентності:

1. Стовбець із парних елементів
2. Мішаний стовбець

# Тестові набори даних:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Назва тесту | Умовне позначення | Вхідні дані | Очікуванні результати |
| 1 | Всі елементи стовбця непарні | m | 1, 3, 5 | 9, 3, 5 |
| 2 | Всі елементи стовбця парні | m | 2, 4, 6 | 2, 4, 6 |
| 3 | Стовбець містить парні та непарні елементи | m | 2, 3, 4 | 3, 3, 4 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № тесту | № рядка з помилкою | Тип помилки | Опис помилки | Засіб локалізації | Можливі дії для усунення | Виконані дії для усунення |
| Помилка компіляції | 6, 36, 90 | Синтаксична | Пропущена крапка з комою | Підкреслив компілятор | Поставити крапку з комою | Поставити крапку з комою |
| Помилка компіляції | 14 | Синтаксична | Не правильні кутові дужки оператора cin | Підкреслив компілятор | Прописати правильні дужки | Прописати правильні дужки |
| Помилка компіляції | 28, 70, 90 | Лексична | Зайвий символ в коді | Підкреслив компілятор | Видалити символ | Видалити символ |
| Помилка компіляції | 49 | Лексична | Не прописані параметри в функції | Огляд коду | Прописати параметри | Прописати параметри |
| Помилка компіляції | 38 | Лексична | Не правильна назва параметрів | Огляд коду | Замінити назву параметрів | Замінити назву параметрів |
| Помилка компіляції | 43, 45, 46, 56, 57 | Лексична | Не правильно використовуються поля структури | Підкреслив компілятор | Прописати звернення до полів структури | Прописати звернення до полів структури |
| Помилка компіляції | 72 | Лексична | Не правильний ітератор циклу | Підкреслив компілятор | Замінити ітератор | Замінити ітератор |
| Помилка компіляції | 79 | Синтаксична | Пропущена блочна дужка | Огляд коду | Прописати дужку | Прописати дужку |
| Помилка компіляції | 89 | Синтаксична | Пропущена двокрапка | Підкреслив компілятор | Поставити двокрапку | Поставити двокрапку |
| Помилка компіляції | 74, 106, 108, 110,112 | Лексична | Неправильні назви змінних або функцій | Підкреслив компілятор | Виправити назви | Виправити назви |
| - | 88, 74, 75 | Логічна | Неправильний ітератор масиву | Не правильний друк матриці | Виправити ітератор | Виправити ітератор |
| - | 56 | Логічна | Неправильна умова | Огляд коду та специфікацій функції | Виправити умову в операторі | Виправити умову в операторі |
| - | 46, 75 | Синтаксична | Неправильний оператор присвоювання | Точки зупину та відлагодження | Виправити оператор | Виправити оператор |
| - | 68, 72 | логічна | Не правильні умови циклів | Точки зупину та відлагодження | Замінити умови циклу | Замінити умови циклу |
| - | 77 | логічна | Функція повертає значення в циклі | Точки зупину та відлагодження | Прописати оператор return поза циклом | Прописати оператор return поза циклом |

**2. Код налагодженої функції**

#include <iostream>

#include <time.h>

struct Matrix {

int n; // кільксть рядків

int m; // кільксть стовпців

int\*\* arr; // вказівник на двовимірний масив

};

Matrix initNM(void)

// функція введення розмірності матриці

// вхід: -

// вихід: розмірність матриці

{

Matrix A;

std::cout << "Vvod n = "; std::cin >> A.n;

std::cout << "Vvod m = "; std::cin >> A.m;

return A;

}

Matrix create(Matrix A)

// функція виділення динамічної памяті для матриці

// вхід: матриці

// вихід: виділена память для матриці

{

A.arr = new int\* [A.n];

for (int i = 0; i < A.n; i++)

A.arr[i] = new int[A.m];

return A;

}

void initArr(Matrix A)

// функція ініціалізації матриці

// вхід: пуста матриця

// вихід: матриці із заповнених значень

{

for (int i = 0; i < A.n; i++)

for (int j = 0; j < A.m; j++)

A.arr[i][j] = rand() % 50;

}

int min\_row(Matrix A, int jj)

// функція пошуку мінімуму серед елементів стовпця матриці

// вхід: матриця, номер стопця

// вихід: мінімальне значення

{

int min = A.arr[0][jj];

for (int i = 1; i < A.n; i++)

if (min >= A.arr[i][jj])

min = A.arr[i][jj];

return min;

}

int sumcol(Matrix AA, int jj)

//функція пошуку суми непарних елементів стовпця матриці

// вхід: матриця, номер стопця

// вихід: сума елементів стовчика

{

int sum = 0;

for (int i = 0; i < AA.n; i++)

if (AA.arr[i][jj] % 2 != 0)

sum += AA.arr[i][jj];

return sum;

}

Matrix Processing7(Matrix A)

// функція заміни мінімальних елементів у стовпці матриці сумою модулів всіх непарних

//елементів цього стовпчика у поточному рядку

// вхід: матриця

// вихід: змінена матриця

{

int minR = 0;

int sumR = 0;

for (int i = 0; i < A.m; i++)

{

minR = min\_row(A, i);

sumR = sumcol(A, i);

for (int j = 0; j < A.n; j++)

{

if (A.arr[j][i] == minR)

A.arr[j][i] = sumR;

}

//return A;

}

return A;

}

void show(Matrix A)

// функція виводу матриці на екран

// вхід: матриця значень

// вихід: значення матриці на консолі

{

for (int i = 0; i < A.n; i++)

{

for (int j = 0; j < A.m; j++)

std::cout << A.arr[i][j] << "\t";

std::cout << "\n";

}

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

}

void del(Matrix A)

// функція видалення матриці з пам'яті

// вхід: матриця значень

// вихід: видалена матриця

{

for (int і = 0; і < A.n; і++)

delete[] A.arr[і];

delete[] A.arr;

}

int main()

{

Matrix A;

srand(time(0));

A = initNM();

A = create(A);

initArr(A);

show(A);

A = Processing7(A);

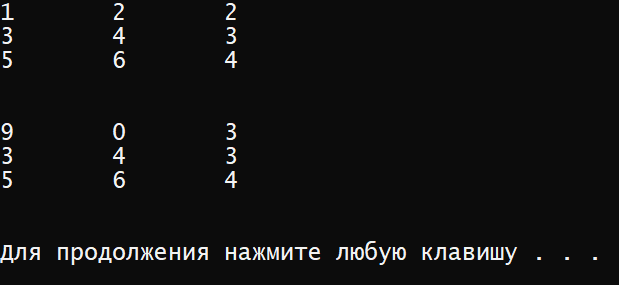
show(A);

del(A);

system("pause");

}

**3. Результати тестувань**



**4. Аналіз результатів**

Під час тестуваня була створена матриця з усіма необхідними стовбцями для тестувань. Програма працює корректно.

**5. Висновки**

Під час виконання лабороторної роботи відлагоджував програму. Для якої були розроблені правильні та не правильні класи еквівалентності та тестові набори. Під час відлагодження користувався точками зупину, покроковим виконанням та спостеріганням за контрольними значеннями змінних, що допомогло добре налагодити програму.