

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет

«Харківський авіаційний інститут»

Факультет систем управління літальних апаратів

Кафедра систем управління літальних апаратів

Лабораторна робота No2

з дисципліни «Алгоритмізація та програмування»

(назва дисципліни)

на тему: «Математичні обчислення на мові C ++»

XAI.301.312.G7.3.ЛР

Виконав: здобувач 1 курсу, групи 312

напряму підготовки (спеціальності)

G7 «Автоматизація комп'ютерно-інтегрованих
технологій та робототехніка»

ОПП Інженерія мобільних додатків

(шифр і назва напряму підготовки (спеціальності))

С. С. БОЯРСЬКИЙ

(прізвище й ініціали студента)

Прийняв: доцент каф.301, канд. техн. наук,

О. В. ГАВРИЛЕНКО

(посада, науковий ступінь, прізвище й ініціали)

МЕТА РОБОТИ

Вивчити теоретично базові типи даних мови C ++ і реалізувати консольний додаток лінійної структури для введення / виведення і обробки змінних базових типів з використанням вбудованих операцій та бібліотечних функцій на мові програмування C ++.

ПОСТАНОВКА ЗАДАЧІ

Завдання 1. Вирішити задачу з цілочисельними змінними. Всі вхідні і вихідні дані в задачах цієї групи є цілими числами. Всі числа, для яких вказано кількість цифр (двозначне число, тризначне число і т. д.), вважаються додатними.

Завдання 2. Вирішити завдання з логічними змінними. У всіх завданнях даної групи потрібно вивести логічне значення true (1), якщо наведене висловлювання для запропонованих вхідних даних є істинним, і значення false (0) в іншому випадку. Всі числа, для яких вказано кількість цифр (двозначне число, тризначне число і т. д.), вважаються цілими додатними.

Завдання 3. Обчислити математичний вираз зі змінними дійсного типу, використовуючи стандартну бібліотеку cmath. Число π має бути визначено як константа дійсного типу.

Завдання 4. Використовуючи ChatGpt, Gemini або інший засіб генеративного ШІ, провести самоаналіз отриманих знань і навичок за допомогою наступних промптів: «Ти - викладач, що приймає захист моєї роботи. Задай мені 5 тестових питань з 4 варіантами відповіді і 5 відкритих питань - за кодом, що є у файлі звіту і теоретичними відомостями у файлі лекції». «Оціни повноту, правильність та ймовірність використання штучного інтелекту для кожної відповіді. Сформулюй загальну оцінку у 5-бальній шкалі, віднімаючи 50% балів

там, де ймовірність відповіді з засобом ШІ висока».

Проаналізуйте задані питання, коментарі і оцінки, надані ШІ. Додайте 2-3 власних промпта у продовження діалогу для поглиблення розуміння теми.

ВИКОНАННЯ РОБОТИ

ЗАВДАННЯ 1

Вирішення задачі Integer12

Вхідні дані (ім'я, опис, тип, обмеження):

n — трицифрове ціле число (int, обмеження: $|n| \in [100; 999]$)

Вихідні дані (ім'я, опис, тип):

reversed — перевернуте число того ж знаку (int)

Алгоритм вирішення:

1. Вивести запрошення до вводу.
2. Зчитати число n .
3. Перевірити, чи воно трицифрове.
4. Зберегти знак числа.
5. Розбити n на цифри: сотні, десятки, одиниці.
6. Сформувати нове число у зворотному порядку.
7. Вивести результат із врахуванням знаку.

ЗАВДАННЯ 2

Вирішення задачі Boolean37

Вхідні дані (ім'я, опис, тип, обмеження):

$x1, y1$ — координати початкового поля (int, $1 \leq x, y \leq 8$)

$x2, y2$ — координати кінцевого поля (int, $1 \leq x, y \leq 8$)

Вихідні дані (ім'я, опис, тип):

canMove — логічне значення, чи може король перейти за один хід (bool)

Алгоритм вирішення:

1. Вивести запрошення до введення координат.
2. Зчитати $x1, y1, x2, y2$.
3. Перевірити, чи координати в межах 1–8.
4. Переконатися, що початкове та кінцеве поля різні.
5. Обчислити різницю між координатами.
6. Якщо $|x1-x2| \leq 1$ і $|y1-y2| \leq 1 \rightarrow$ король може зробити хід.
7. Вивести результат.

ЗАВДАННЯ 3

Вирішення задачі Math22

Вхідні дані (ім'я, опис, тип, обмеження):

x — дійсне число, аргумент функції (double, $x > 0$)

Вихідні дані (ім'я, опис, тип):

y — результат обчислення математичного виразу (double)

Алгоритм вирішення:

1. Вивести запрошення до введення x .
2. Зчитати x .
3. Перевірити допустимість ($x > 0$, підкореневі вирази ≥ 0).
4. Обчислити чисельник
5. Обчислити знаменник.
6. Переконалися, що знаменник $\neq 0$.
7. Обчислити результат.
8. Вивести y .

Лістинг коду вирішення задач наведено в дод. А (стор. 5).

Екрани роботи програми показано на рис. Б.1., рис. Б.2., рис. Б.3.

ЗАВДАННЯ 4

Діалог у ChatGPT для самоаналізу наведено у дод. В.

ВИСНОВКИ

У ході виконання лабораторної роботи було вивчено базові типи даних мови програмування C++ та реалізовано консольний додаток лінійної структури.

Під час розробки трьох програм були використані різні типи змінних: цілі (int), дійсні (double) та логічні (bool), а також умовні оператори для перевірки коректності введених даних.

Було закріплено навички роботи з операціями ділення, остачі від ділення, піднесення до степеня, обчислення модуля, а також із вбудованими математичними функціями стандартної бібліотеки `<cmath>`.

В результаті роботи було створено програми, які:

виконують обробку числових даних;

реалізують перевірку логічних умов;

здійснюють обчислення складних математичних виразів.

Отже, поставлена мета — вивчити базові типи даних і реалізувати консольний застосунок лінійної структури з використанням вбудованих операцій та бібліотечних функцій C++ — успішно досягнута.

ДОДАТОК А ЛІСТИНГ КОДУ ПРОГРАМИ

```
#include <iostream>
#include <cmath>
using namespace std;

int main() {

    // Завдання Integer12

    cout << "Завдання Integer12. Перевернуте трицифрове число" << endl;

    int n;
    cout << "Введіть трицифрове число: ";
    cin >> n;

    // Перевірка коректності введення
    if (abs(n) < 100 || abs(n) > 999) {
        cout << "Помилка: число не є трицифровим!" << endl;
    } else {
        int sign = (n < 0) ? -1 : 1; // зберігаємо знак, якщо число від'ємне
        n = abs(n);

        // Розбиваємо число на цифри
        int a = n / 100;           // перша цифра (сотні)
        int b = (n / 10) % 10;     // друга цифра (десятки)
        int c = n % 10;           // третя цифра (одиниці)

        int reversed = c * 100 + b * 10 + a;
        cout << "Результат: " << sign * reversed << endl;
    }
}
```

```

// Завдання Boolean37

cout << "\nЗавдання Boolean37. Хід короля на шахівниці" << endl;

int x1, y1, x2, y2;
cout << "Введіть x1, y1: ";
cin >> x1 >> y1;
cout << "Введіть x2, y2: ";
cin >> x2 >> y2;

// Перевірка правильності координат (1..8)
if (x1 < 1 || x1 > 8 || y1 < 1 || y1 > 8 ||
    x2 < 1 || x2 > 8 || y2 < 1 || y2 > 8) {
    cout << "Помилка: координати мають бути в межах 1..8!" << endl;
} else if (x1 == x2 && y1 == y2) {
    cout << "Помилка: поля не повинні збігатися!" << endl;
} else {
    bool canMove = (abs(x1 - x2) <= 1) && (abs(y1 - y2) <= 1);
    cout << boolalpha << "Може перейти за один хід: " << canMove << endl;
}

// Завдання Math22

cout << "\nЗавдання Math22. Обчислення математичного виразу" << endl;

const double pi = 3.141592; // константа пі
double x;
cout << "x = ";
cin >> x;

// Перевірка діапазону допустимих значень
if (x <= 0) {
    cout << "Помилка: для логарифма потрібно x > 0!" << endl;
} else if ((2 * x) < 0 || (5 * x) < 0) {
    cout << "Помилка: підкореневий вираз не може бути від'ємним!" << endl;
} else {
    double numerator = log(fabs(x)) * sqrt(fabs(pow(x, 2) * pow(sin(x), 3) *
sqrt(fabs(cos(x)))));
    double denominator = cos(x) + (1.0 / 5.0) * sqrt(2 * x) + sqrt(5 * x);

    if (denominator == 0) {
        cout << "Помилка: знаменник дорівнює нулю!" << endl;
    } else {
        double y = (1.0 / 4.0) * (numerator / denominator);
        cout << "y = " << y << endl;
    }
}

return 0;
}

```

ДОДАТОК Б

СКРІНШОТИ ВІКНА ВИКОНАННЯ ПРОГРАМИ

```

1 #include <iostream>
2 #include <cmath>
3 using namespace std;
4
5 int main() {
6     // Завдання Integer12
7
8     cout << "Завдання Integer12. Перевернуте трицифрове число" << endl;
9
10    int n;
11    cout << "Введіть трицифрове число: ";
12    cin >> n;
13
14    // Перевірка коректності введення
15    if (abs(n) < 100 || abs(n) > 999) {
16        cout << "Помилка: число не є трицифровим!" << endl;
17    } else {
18        int sign = (n < 0) ? -1 : 1; // зберігаємо знак, якщо число від'ємне
19        n = abs(n);
20
21        // Розбиваємо число на цифри
22        int a = n / 100; // перша цифра (сотні)
23        int b = (n / 10) % 10; // друга цифра (десятки)
24        int c = n % 10; // третя цифра (одиниці)
25
26        int reversed = c * 100 + b * 10 + a;
27        cout << "Результат: " << sign * reversed << endl;
28    }
29
30    // Завдання Boolean37
31
32    cout << "\nЗавдання Boolean37. Хід короля на шахівниці" << endl;
33
34    int x1, y1, x2, y2;
35    cout << "Введіть x1, y1: ";
36    cin >> x1 >> y1;
37    cout << "Введіть x2, y2: ";
38    cin >> x2 >> y2;
39
40    // Перевірка правильності координат (1..8)
41    if (x1 < 1 || x1 > 8 || y1 < 1 || y1 > 8 ||
42        x2 < 1 || x2 > 8 || y2 < 1 || y2 > 8) {
43        cout << "Помилка: координати мають бути в межах 1..8!" << endl;
44    } else if (x1 == x2 && y1 == y2) {
45        cout << "Помилка: поля не повинні збігатися!" << endl;
46    } else {
47

```

Завдання Integer12. Перевернуте трицифрове число
Введіть трицифрове число: 986
Результат: 689

Рисунок - Б.1


```

main.cpp
16- if (abs(n) < 100 || abs(n) > 999) {
17-     cout << "Помилка: число не є трицифровим!" << endl;
18- } else {
19-     int sign = (n < 0) ? -1 : 1; // зберігаємо знак, якщо число від'ємне
20-     n = abs(n);
21-
22-     // Розбиваємо число на цифри
23-     int a = n / 100; // перша цифра (сотні)
24-     int b = (n / 10) % 10; // друга цифра (десятки)
25-     int c = n % 10; // третя цифра (одиниці)
26-
27-     int reversed = c * 100 + b * 10 + a;
28-     cout << "Результат: " << sign * reversed << endl;
29- }
30-
31- // Завдання Boolean37
32-
33- cout << "\nЗавдання Boolean37. Хід короля на шахівниці" << endl;
34-
35- int x1, y1, x2, y2;
36- cout << "Введіть x1, y1: ";
37- cin >> x1 >> y1;
38- cout << "Введіть x2, y2: ";
39- cin >> x2 >> y2;
40-
41- // Перевірка правильності координат (1..8)
42- if (x1 < 1 || x1 > 8 || y1 < 1 || y1 > 8 ||
43-     x2 < 1 || x2 > 8 || y2 < 1 || y2 > 8) {
44-     cout << "Помилка: координати мають бути в межах 1..8!" << endl;
45- } else if (x1 == x2 && y1 == y2) {
46-     cout << "Помилка: поля не повинні збігатися!" << endl;
47- } else {
48-     bool canMove = (abs(x1 - x2) <= 1 && (abs(y1 - y2) <= 1));
49-     cout << boolalpha << "Може перейти за один хід: " << canMove << endl;
50- }
51-
52- // Завдання Math22
53-
54- cout << "\nЗавдання Math22. Обчислення математичного виразу" << endl;
55-
56- const double pi = 3.141592; // константа pi
57- double x;
58- cout << "x = ";
59- cin >> x;
60-
61- // Перевірка діапазону допустимих значень
62- if (x <= 0) {
63-     cout << "Помилка: для логарифма потрібно x > 0!" << endl;
64- } else if ((2 * x) < 0 || (5 * x) < 0) {
65-     cout << "Помилка: підкореневий вираз не може бути від'ємним!" << endl;
66- } else {
67-     double numerator = log(fabs(x)) * sqrt(fabs(cos(x, 2) * sin(sin(x), 3) * sqrt(fabs(cos(x)))));
68-     double denominator = cos(x) + (1.0 / 5.0) * sqrt(2 * x) + sqrt(5 * x);
69-
70-     if (denominator == 0) {
71-         cout << "Помилка: Значення дорівнює нулю!" << endl;
72-     } else {
73-         double y = (1.0 / 4.0) * (numerator / denominator);
74-         cout << "y = " << y << endl;
75-     }
76- }
77-
78- return 0;
79- }

```

Input

Завдання Boolean37. Хід короля на шахівниці
Введіть x1, y1: 2
3
Введіть x2, y2: 2
5
Може перейти за один хід: false

Рисунок - Б.2

```

main.cpp
28- cout << "Результат: " << sign * reversed << endl;
29- }
30-
31- // Завдання Boolean37
32-
33- cout << "\nЗавдання Boolean37. Хід короля на шахівниці" << endl;
34-
35- int x1, y1, x2, y2;
36- cout << "Введіть x1, y1: ";
37- cin >> x1 >> y1;
38- cout << "Введіть x2, y2: ";
39- cin >> x2 >> y2;
40-
41- // Перевірка правильності координат (1..8)
42- if (x1 < 1 || x1 > 8 || y1 < 1 || y1 > 8 ||
43-     x2 < 1 || x2 > 8 || y2 < 1 || y2 > 8) {
44-     cout << "Помилка: координати мають бути в межах 1..8!" << endl;
45- } else if (x1 == x2 && y1 == y2) {
46-     cout << "Помилка: поля не повинні збігатися!" << endl;
47- } else {
48-     bool canMove = (abs(x1 - x2) <= 1 && (abs(y1 - y2) <= 1));
49-     cout << boolalpha << "Може перейти за один хід: " << canMove << endl;
50- }
51-
52- // Завдання Math22
53-
54- cout << "\nЗавдання Math22. Обчислення математичного виразу" << endl;
55-
56- const double pi = 3.141592; // константа pi
57- double x;
58- cout << "x = ";
59- cin >> x;
60-
61- // Перевірка діапазону допустимих значень
62- if (x <= 0) {
63-     cout << "Помилка: для логарифма потрібно x > 0!" << endl;
64- } else if ((2 * x) < 0 || (5 * x) < 0) {
65-     cout << "Помилка: підкореневий вираз не може бути від'ємним!" << endl;
66- } else {
67-     double numerator = log(fabs(x)) * sqrt(fabs(cos(x, 2) * sin(sin(x), 3) * sqrt(fabs(cos(x)))));
68-     double denominator = cos(x) + (1.0 / 5.0) * sqrt(2 * x) + sqrt(5 * x);
69-
70-     if (denominator == 0) {
71-         cout << "Помилка: Значення дорівнює нулю!" << endl;
72-     } else {
73-         double y = (1.0 / 4.0) * (numerator / denominator);
74-         cout << "y = " << y << endl;
75-     }
76- }
77-
78- return 0;
79- }

```

Input

Завдання Math22. Обчислення математичного виразу
x = 55
y = 1.13585

Рисунок - Б.3

ДОДАТОК В ДІАЛОГ З ШІ ДЛЯ САМОАНАЛІЗУ

Добре

Ось приклад 10 контрольних питань (5 тестових + 5 відкритих), які я б поставив як викладач на захисті цієї лабораторної роботи.

Тестові питання (по 4 варіанти відповіді)

1. Що робить оператор `abs(n)` у програмі?

- A) Знаходить квадрат числа
- B) Перетворює число в його від'ємний еквівалент
- C) Обчислює модуль числа (без знаку)
- D) Виводить абсолютне значення на екран

2. Для якої з умов число вважається трицифровим у програмі?

- A) `n > 99 && n < 1000`
- B) `n >= 10 && n <= 9999`
- C) `n > 0 && n < 100`
- D) `abs(n) > 10 && abs(n) < 1000`

3. Що перевіряє логічний вираз

`(abs(x1 - x2) <= 1) && (abs(y1 - y2) <= 1)?`

- A) Чи стоїть тура поруч з ферзем
- B) Чи може кінь перейти на інше поле
- C) Чи може король зробити один хід
- D) Чи рівні координати клітин

4. Яке значення константи `pi` використовується в програмі?

- A) 3.1415
- B) 3.14
- C) 3.141592
- D) 3.14159265

5. У фрагменті

```
double y = (1.0 / 4.0) * (numerator / denominator);
```

число $1.0 / 4.0$ означає:

- A) Ділення змінної `numerator` на 4
- B) Множення результату на чверть
- C) Обчислення залишку від ділення
- D) Перевірку умови ділення на 4

Відкриті питання

1. Поясніть, як у програмі відбувається перевірка правильності введення трицифрового числа.
2. Яким чином реалізовано збереження знаку введеного числа у завданні *Integer12* і для чого це потрібно?
3. Поясніть логіку перевірки правильності координат шахової дошки (1..8) та чому важливо перевіряти, що поля не збігаються.
4. Опишіть алгоритм обчислення виразу у завданні *Math22*: які дії виконуються послідовно та які бібліотечні функції використовуються (`log`, `sqrt`, `fabs`, `pow`, `cos`, `sin`).
5. Яке практичне значення мають базові типи даних у C++ (`int`, `double`, `bool`) і як вони використовуються у вашій програмі?

Ви сказали:

Тестові питання 1. С 2. А 3. С 4. С 5. В

ChatGPT сказав:

1. $\text{abs}(n)$ — повертає модуль числа (тобто беззнакове значення).

Відповідь: С — Вірно

2. Умова трицифрового числа $n > 99 \ \&\& \ n < 1000$ (або $\text{abs}(n) \geq 100 \ \&\& \ \text{abs}(n) \leq 999$) — саме вона використовується.

Відповідь: А — Вірно

3. Вираз $(\text{abs}(x1 - x2) \leq 1) \ \&\& \ (\text{abs}(y1 - y2) \leq 1)$ перевіряє, що король може перейти за один хід.

Відповідь: С — Вірно

4. У програмі `const double pi = 3.141592;`

Відповідь: С — Вірно

5. $(1.0 / 4.0)$ — це множення результату на чверть (0.25), тобто зменшення у 4 рази.

Відповідь: В — Вірно

Підсумок: 5/5 правильних!

Ти чудово розібрався у своїй програмі — видно, що розумієш не лише код, а й математичну логіку за ним

Ви сказали:

Відповіді на відкриті питання:

1. Функція `abs` повертає модуль числа, щоб перевіряти від'ємні числа також..
 2. Спочатку обчислюється з модулем, а потім множиться на `sign` щоб отримати той знак, що був спочатку
 3. Спочатку перевіряється чи всі координати від 1 до 8, а потім чи не збігаються початкова та кінцева клітинка.
 4. Вводиться значення `x`, перевіряється, що воно більше нуля, обчислюються чисельник та знаменник, перевіряється, щоб знаменник не був рівний 0, отримується результат.
 5. `int` зберігає цілі числа
- `double` дійсні числа з плаваючою комою, `bool` це логічні значення правда/неправда

ChatGPT сказав:

Оцінка твоїх відповідей на відкриті питання:

1. “Функція `abs` повертає модуль числа, щоб перевіряти від'ємні числа також.”

Правильно по суті. Коротко, але логічно. Можна було трохи точніше сформулювати, але зміст зрозумілий.

Оцінка: 1 бал із 1

2. “Спочатку обчислюється з модулем, а потім множиться на `sign` щоб отримати той знак, що був спочатку.”

Абсолютно вірно. Правильне розуміння алгоритму та призначення `sign`.

Оцінка: 1 бал із 1

3. “Спочатку перевіряється чи всі координати від 1 до 8, а потім чи не збігаються початкова та кінцева клітинка.”

Повністю правильно. Чітко описана логіка перевірки.

Оцінка: 1 бал із 1

4. “Вводиться значення x , перевіряється, що воно більше нуля, обчислюються чисельник та знаменник, перевіряється, щоб знаменник не був рівний 0, отримується результат.”

Правильно. Послідовно описані основні кроки алгоритму. Можна було додати про використанні функції, але навіть так — зміст коректний.

Оцінка: 0.9 бала із 1

5. “`int` зберігає цілі числа; `double` дійсні числа з плаваючою комою; `bool` це логічні значення правда/неправда.”

Повністю правильно. Коротко, чітко і без помилок.

Оцінка: 1 бал із 1

Загальний результат: 4.9 / 5 балів (відмінно)

Коментар: Відповіді правильні, стислі, по суті. Рекомендується лише трохи розширити пояснення у 4-му пункті (згадати функції `log`, `sqrt`, `fabs` тощо), якщо на захисті попросять деталізувати.

Ви сказали:

чому без модуля не зберігається знак числа?

ChatGPT сказав:

Дуже гарне запитання — і воно показує, що ти вже розумієш суть роботи зі знаком числа.

Поясню докладно й просто:

1. Що таке "модуль" числа

Функція `abs(x)` повертає модуль числа, тобто його величину без знаку.

- `abs(123) → 123`
- `abs(-123) → 123`

Отже, після виклику `abs(n)` інформація про знак числа втрачається — результат завжди додатний.

2. Чому це важливо в твоїй програмі

Коли ти працюєш із трицифровими числами (наприклад, у завданні `Integer12`), ти хочеш обробити тільки цифри, тобто величину без знаку.

Тому:

```
int sign = (n < 0) ? -1 : 1;
```

```
n = abs(n);
```

Тут ти:

1. Спочатку запам'ятовуєш знак (`sign = -1` або `1`),
 2. Потім береш модуль числа, щоб зручно працювати з цифрами (`100`, `10`, `1`),
 3. А після всіх обчислень — повертаєш знак назад:
 4. `result = sign * (c * 100 + b * 10 + a);`
-

3. Що буде, якщо не брати модуль

Припустимо, ти вводиш `-123` і не використовуєш `abs(n)`:

- Поділ на 100, 10 і % дасть не ті цифри, бо арифметика з від'ємними числами поводиться інакше.
 - Наприклад: $-123 \% 10 \rightarrow -3$, а не 3.
 - Тобто ти отримаєш неправильні цифри для подальшої роботи.
-

Висновок

Без `abs()`:

- Втрачається коректність обчислень із цифрами.

З `abs()`:

- Ти працюєш тільки з позитивними цифрами,
- А потім множиш на `sign`, щоб повернути початковий знак.

Ви сказали:

я чув, що в C++ потрібно очищувати пам'ять. Скажи, як це робиться і чи потрібно це робити в моїй роботі?

Дуже слушне запитання — воно показує, що ти розумієш не лише синтаксис, а й роботу з пам'яттю в C++.

Розберімо коротко:

1. Коли потрібно очищувати пам'ять у C++

У C++ пам'ять буває двох основних типів:

- Статична / стекова пам'ять — виділяється автоматично, коли ти оголошуєш змінні типу
- `int n; double x; bool flag;`

або функції з локальними змінними.

Таку пам'ять не потрібно очищувати вручну — вона звільняється автоматично, коли виходиш із блоку коду (функції).

- Динамічна пам'ять — виділяється вручну через оператор `new`, наприклад:
- `int* p = new int;`

У цьому випадку ти сам маєш звільнити пам'ять командою

`delete p;`

Якщо цього не зробити, виникає витік пам'яті — програма поступово займає дедалі більше оперативної пам'яті.

2. Як очищується динамічна пам'ять

- Для одного об'єкта:
- `delete p;`
- Для масиву:
- `delete[] arr;`
- Якщо використовуються сучасні структури (`std::vector`, `std::string`, `std::unique_ptr` тощо), вони очищують пам'ять автоматично, коли виходять із області видимості.

3. Чи потрібно очищувати пам'ять у твоїй роботі

У твоїй програмі немає жодного використання оператора `new` або динамічних структур, усі змінні (`int`, `double`, `bool`) створюються в стековій пам'яті і автоматично звільняються після завершення функції `main()`.

Отже:

Очищення пам'яті вручну у твоїй програмі не потрібне.

Хочеш, я коротко покажу приклад, як би твоя програма виглядала, якби ти виділяв пам'ять динамічно, і як її тоді треба було б очищати?

Вы сказали:

так

ChatGPT сказав:

Добре, подивись приклад 🙌

Я покажу, як аналог твоєї програми міг би виглядати, якби змінні створювалися динамічно (через new), і як потім правильно очистити пам'ять.

Приклад із динамічним виділенням пам'яті

```
#include <iostream>
```

```
#include <cmath>
```

```
using namespace std;
```

```
int main() {
```

```
    // Приклад: перевернуте трицифрове число (динамічний варіант)
```

```
    int* n = new int; // виділяємо пам'ять під змінну n у динамічній області
```

```
    cout << "Введіть трицифрове число: ";
```

```
    cin >> *n; // звертаємось до значення через *
```

```
    if (abs(*n) < 100 || abs(*n) > 999) {
```

```
        cout << "Помилка: число не є трицифровим!" << endl;
```

```

} else {

    int sign = (*n < 0) ? -1 : 1;

    *n = abs(*n);

    int a = *n / 100;

    int b = (*n / 10) % 10;

    int c = *n % 10;

    int reversed = c * 100 + b * 10 + a;

    cout << "Результат: " << sign * reversed << endl;

}

// Після завершення роботи зі змінною обов'язково очищуємо пам'ять
delete n; // Звільняємо пам'ять, виділену через new
n = nullptr; // добрий стиль — обнулити вказівник після видалення

return 0;

}

```

Що відбувається

1. `int* n = new int;` — виділяється пам'ять у кучі (heap).
2. Через `*n` ми працюємо з реальним значенням.
3. Після використання — обов'язково викликається `delete n;`, інакше залишиться “витік пам'яті”.

4. Присвоєння `p = nullptr`; робиться, щоб уникнути “висячих вказівників”.
-

Висновок

- У твоїй лабораторній — усе створено на стеку, тому очищення не потрібно.
- Якщо б ти працював із вказівниками, масивами або динамічними структурами, тоді треба було б:
 - використовувати `delete` або `delete[]`,
 - або застосовувати розумні вказівники (`std::unique_ptr`, `std::shared_ptr`), які роблять це автоматично.