

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний аерокосмічний університет
«Харківський авіаційний інститут»

Факультет систем управління літальних апаратів
Кафедра систем управління літальних апаратів

Лабораторна робота № 2

з дисципліни «Алгоритмізація та програмування»
на тему «Математичні обчислення на мові C ++
»

ХАІ.301. G6 Інформаційно-вимірювальні технології. Інформаційно-
вимірювальні технології забезпечення якості продукції . 318. 21 ЛР

Виконав студент гр. _____318_____

_____ Анастасія Шльомка _____
(підпис, дата) 03.11.25 (П.І.Б.) Шльомка

Анастасія Андріївна

Перевірів

_____ к.т.н., доц. Олена ГАВРИЛЕНКО
(підпис, дата) (П.І.Б.)

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. Прошло N секунд від початку доби. Кожна хвилина має 60 секунд.

Щоб дізнатися, скільки секунд пройшло в поточній хвилині, потрібно взяти остачу від ділення на 60.

формула: $r = N \% 60$

Завдання 2. Boolean 21 Умова: Дано тризначне число. Перевірити істинність висловлювання: «Цифри даного числа утворюють зростаючу послідовність».

Завдання 3. Math 21: Обчислити значення виразу
$$\left[y = \frac{\sin(x) + \ln(1+x^2)}{e^x + \cos^2(x)} \right]$$

ВИКОНАННЯ РОБОТИ

Завдання 1. Integer. 21

Вирішення задачі Цілочисельні операції, Integer 21

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

Ім'я змінної: n

Опис: Кількість секунд з початку останньої хвилини

Тип: int

Обмеження: $N \geq 0$

Вхідні данні:

Ім'я змінної: seconds

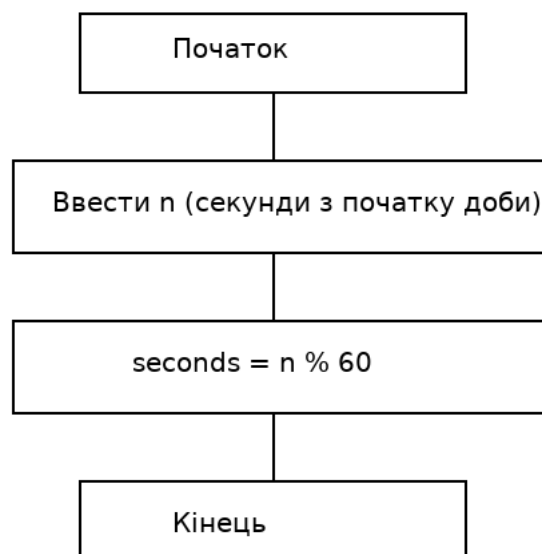
Опис: Кількість секунд з початку останньої хвилини мінної

Тип: Int

Алгоритм вирішення:

1. Ввести *n*.
2. Знайти сотні, десятки, одиниці.
3. Перевірити умову `hundreds < tens && tens < ones`.
4. Вивести true або false.

Малюнок:



Малюнок 1 – Алгоритм визначення кількості секунд, що пройшли з початку останньої хвилини

Код до завдання 1

```
#include <iostream>
#include <cmath>      // підключення бібліотеки математичних функцій: sqrt, log
using namespace std;

int main() {
    // Integer21. З початку доби минуло N секунд (N – ціле).
    // Знайти кількість секунд, що пройшли з початку останньої хвилини.
    cout << "Integer21\n";
    int N;
    cout << "N (s) = ";
    cin >> N;
    // операція взяття залишку від ділення на 60 дає секунди останньої хвилини
    int sec_last_min = N % 60;
    cout << "Seconds from start of last minute: "
         << sec_last_min << "\n\n";
}
```

Вхідні дані:

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

Ім'я змінної: n

Опис: Тризначне число

Тип: `int`

Обмеження: $N \geq 0$

Вихідні дані:

Ім'я : `is_increasing`

Опис: Істинність висловлювання

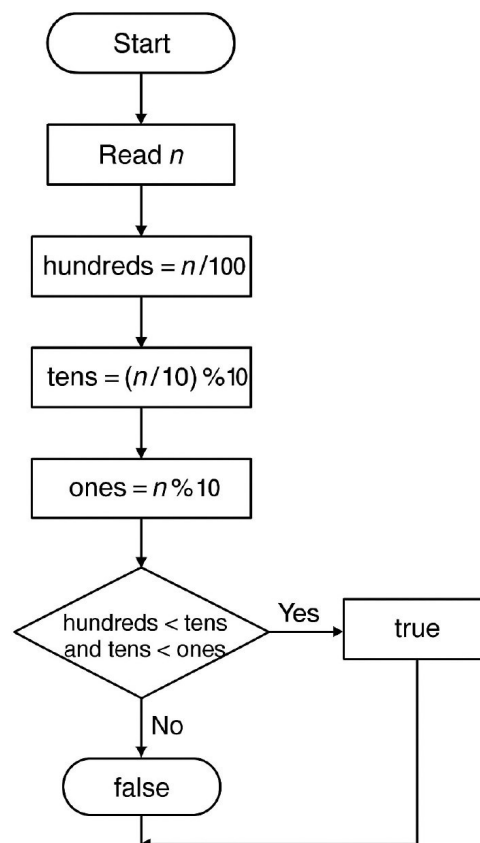
Тип: `bool`

Алгоритм:

1. Ввести `*n*`.
2. Знайти сотні, десятки, одиниці.
3. Перевірити умову `hundreds < tens && tens < ones`.
4. Вивести `true` або `false`.

Малюнок:

Малюнок 2



Код до завдання 2

```
// Boolean21. Дано тризначне число A.  
// Перевірити істинність висловлювання:  
// «Цифри даного числа утворюють зростаючу послідовність».  
cout << "Boolean21\n";  
int A;  
cout << "Three-digit A = ";  
cin >> A;  
// дістаємо цифри сотень, десятків і одиниць  
int hundreds = A / 100;      // перша цифра (сотні)  
int tens      = (A / 10) % 10; // друга цифра (десятки)  
int ones      = A % 10;       // третя цифра (одиниці)  
// перевіряємо, що сотні < десятки < одиниці  
bool is_strictly_increasing = (hundreds < tens) && (tens < ones);  
cout << boolalpha  
      << "Digits form strictly increasing sequence: "  
      << is_strictly_increasing << "\n\n";
```

Завдання 3. Math 21

Вхідні данні:

Ім'я: x

Опис: Аргумент функції

Тип: double

Ім'я: y

Опис: Значення функції.

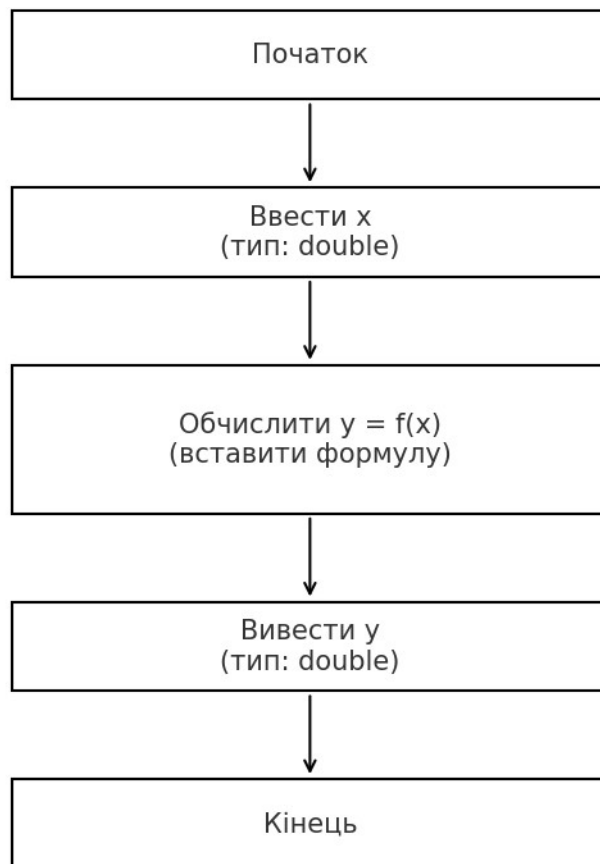
Тип: double

Алгоритм:

1.Ввести *x*.

2.Обчислити значення функції за формулою.

3.Вивести результат.



Малюнок 3 – Алгоритм обчислювання математичного виразу.

```

// Math21. Обчислити за формулою:
//  $y = \sqrt[3]{\left| x^2 - 2 \cdot |\sin x| \cdot 3 \cdot \operatorname{tg} x \right| \cdot 5^{\cos(x - 12)}} \cdot 0.6 + 4 \cdot \log_2(x + 15)$ 
// -----
//
// 1)  $\sqrt[3]{\dots}$  – кубічний корінь, у C++ – функція cbrt(z).
// 2)  $|\sin x|$  – fabs(sin(x))
// 3)  $\operatorname{tg} x$  – тангенс: tan(x)
// 4)  $5^{\dots}$  – pow(5, ...)
// 5)  $\log_2(\dots)$  – двійковий логарифм: log2(...)
cout << "Math21\n";
double x;
cout << "Real x (> -15): ";
cin >> x;

// Перевірка область визначення:  $x + 15 > 0$ 
if (x <= -15) {
    cout << "Error: x + 15 must be positive.\n";
    return 1;
}

// Обчислюємо окремі частини:
double sinx      = sin(x);
double abssinx   = fabs(sinx);
double tgx       = tan(x);
double expr_inside = x*x - 2 * abssinx * 3 * tgx;
double abs_expr   = fabs(expr_inside);
double pow5       = pow(5.0, cos(x - 12.0));
double numerator  = abs_expr * pow5;

double denom = 0.6 + 4.0 * log2(x + 15.0);

// Кубічний корінь із чисельника
double y = cbrt(numerator) / denom;

cout << "y = " << y << "\n";

return 0;
}

```

Код до завдання 3

Лістинг коду вирішення задачі варіанту 21 завдань 1-3:

```

#include <iostream>
#include <cmath>          // підключення бібліотеки математичних функцій: sqrt,
log
using namespace std;

int main() {
    // Integer21. З початку доби минуло N секунд (N – ціле).
    // Знайти кількість секунд, що пройшли з початку останньої хвилини.
    cout << "Integer21\n";
    int N;
    cout << "N (s) = ";
    cin >> N;
    // операція взяття залишку від ділення на 60 дає секунди останньої
хвилини
    int sec_last_min = N % 60;
    cout << "Seconds from start of last minute: "
        << sec_last_min << "\n\n";

    // Boolean21. Дано тризначне число A.
    // Перевірити істинність висловлювання:
    // «Цифри даного числа утворюють зростаючу послідовність».
    cout << "Boolean21\n";
    int A;
    cout << "Three-digit A = ";
    cin >> A;
    // дістаємо цифри сотень, десятків і одиниць
    int hundreds = A / 100;          // перша цифра (сотні)
    int tens      = (A / 10) % 10;    // друга цифра (десятки)
    int ones      = A % 10;           // третя цифра (одиниці)
    // перевіряємо, що сотні < десятки < одиниці
    bool is_strictly_increasing = (hundreds < tens) && (tens < ones);
    cout << boolalpha
        << "Digits form strictly increasing sequence: "
        << is_strictly_increasing << "\n\n";

    // Math21. Обчислити за формулою:
    // 
$$y = \sqrt[3]{\sqrt{|x^2 - 2 \cdot |\sin x| \cdot 3 \cdot \operatorname{tg} x|} \cdot 5^{\cos(x - 12)}} \cdot 0.6 + 4 \cdot \log_2(x + 15)$$

    //
    // 1)  $\sqrt[3]{...}$  – кубічний корінь, у C++ – функція cbrt(z).
    // 2)  $|\sin x|$  – fabs(sin(x))
    // 3)  $\operatorname{tg} x$  – тангенс: tan(x)
    // 4)  $5^{(...)}$  – pow(5, ...)
    // 5)  $\log_2(...)$  – двійковий логарифм: log2(...)

```

```

cout << "Math21\n";
double x;
cout << "Real x (> -15): ";
cin >> x;

// Перевірка область визначення:  $x + 15 > 0$ 
if (x <= -15) {
    cout << "Error: x + 15 must be positive.\n";
    return 1;
}

// Обчислюємо окремі частини:
double sinx      = sin(x);
double abssinx   = fabs(sinx);
double tgx       = tan(x);
double expr_inside = x*x - 2 * abssinx * 3 * tgx;
double abs_expr   = fabs(expr_inside);
double pow5       = pow(5.0, cos(x - 12.0));
double numerator  = abs_expr * pow5;

double denom = 0.6 + 4.0 * log2(x + 15.0);

// Кубічний корінь із чисельника
double y = cbrt(numerator) / denom;

cout << "y = " << y << "\n";

return 0;
}

```

ВИСНОВКИ

Було вивчено принципи роботи з цілими, логічними та дійсними типами даних у C++. Закріплено на практиці операції ділення, взяття залишку, логічні вирази й обчислення з використанням бібліотеки `cmath`.

ДОДАТОК Варіант 21. Завдання 1.

Лістинг коду програми

```
#include <iostream>
#include <cmath>          // підключення бібліотеки математичних функцій: sqrt, log
using namespace std;

int main() {
    // Integer21. З початку доби минуло N секунд (N - ціле).
    // Знайти кількість секунд, що пройшли з початку останньої хвилини.
    cout << "Integer21\n";
    int N;
    cout << "N (s) = ";
    cin >> N;
    // операція взяття залишку від ділення на 60 дає секунди останньої хвилини
    int sec_last_min = N % 60;
    cout << "Seconds from start of last minute: "
         << sec_last_min << "\n\n";
```

Варіант 21. Завдання 2.

Лістинг коду програми

```
// Boolean21. Дано тризначне число A.
// Перевірити істинність висловлювання:
// «Цифри даного числа утворюють зростаючу послідовність».
cout << "Boolean21\n";
int A;
cout << "Three-digit A = ";
cin >> A;
// дістаємо цифри сотень, десятків і одиниць
int hundreds = A / 100;          // перша цифра (сотні)
int tens      = (A / 10) % 10;    // друга цифра (десятки)
int ones      = A % 10;          // третя цифра (одиниці)
// перевіряємо, що сотні < десятки < одиниці
bool is_strictly_increasing = (hundreds < tens) && (tens < ones);
cout << boolalpha
      << "Digits form strictly increasing sequence: "
      << is_strictly_increasing << "\n\n";
```

Варіант 21. Завдання 3.

Лістинг коду програми

```
// Math21. Обчислити за формулою:
//      
$$y = \frac{\sqrt[3]{|x^2 - 2 \cdot |\sin x| \cdot 3 \cdot \operatorname{tg} x|} \cdot 5^{\cos(x - 12)}}{0.6 + 4 \cdot \log_2(x + 15)}$$

//      -----
//      0.6 + 4 · log2(x + 15)
//
// 1)  $\sqrt[3]{\dots}$  – кубічний корінь, у C++ – функція cbrt(z).
// 2) |sin x| – fabs(sin(x))
// 3) tg x – тангенс: tan(x)
// 4) 5^(...) – pow(5, ...)
// 5) log2(...) – двійковий логарифм: log2(...)
cout << "Math21\n";
double x;
cout << "Real x (> -15): ";
cin >> x;

// Перевірка область визначення: x + 15 > 0
if (x <= -15) {
    cout << "Error: x + 15 must be positive.\n";
    return 1;
}

// Обчислюємо окремі частини:
double sinx      = sin(x);
double abssinx   = fabs(sinx);
double tgx       = tan(x);
double expr_inside = x*x - 2 * abssinx * 3 * tgx;
double abs_expr  = fabs(expr_inside);
double pow5      = pow(5.0, cos(x - 12.0));
double numerator = abs_expr * pow5;

double denom = 0.6 + 4.0 * log2(x + 15.0);

// Кубічний корінь із чисельника
double y = cbrt(numerator) / denom;

cout << "y = " << y << "\n";

return 0;
}
```

ДОДАТОК

Варіант 21. Завдання 1/

Скрін-шоти вікна виконання програми

```
#include <iostream>
#include <cmath>      // підключення бібліотеки математичних функцій: sqrt, log
using namespace std;

int main() {
    // Integer21. З початку доби минуло N секунд (N – ціле).
    // Знайти кількість секунд, що пройшли з початку останньої хвилини.
    cout << "Integer21\n";
    int N;
    cout << "N (s) = ";
    cin >> N;
    // операція взяття залишку від ділення на 60 дає секунди останньої хвилини
    int sec_last_min = N % 60;
    cout << "Seconds from start of last minute: "
         << sec_last_min << "\n\n";
}
```

Варіант 21. Завдання 2/

Скрін-шоти вікна виконання програми

```
// Boolean21. Дано тризначне число A.
// Перевірити істинність висловлювання:
// «Цифри даного числа утворюють зростаючу послідовність».
cout << "Boolean21\n";
int A;
cout << "Three-digit A = ";
cin >> A;
// дістаємо цифри сотень, десятків і одиниць
int hundreds = A / 100;      // перша цифра (сотні)
int tens      = (A / 10) % 10; // друга цифра (десятки)
int ones      = A % 10;       // третя цифра (одиниці)
// перевіряємо, що сотні < десятки < одиниці
bool is_strictly_increasing = (hundreds < tens) && (tens < ones);
cout << boolalpha
     << "Digits form strictly increasing sequence: "
     << is_strictly_increasing << "\n\n";
```

Варіант 21. Завдання 3/
Скрін-шоти вікна виконання програми

```
// Math21. Обчислити за формулою:  
//  $y = \sqrt[3]{|x^2 - 2 \cdot |\sin x| \cdot 3 \cdot \tan x| \cdot 5^{\cos(x - 12)}} \cdot 0.6 + 4 \cdot \log_2(x + 15)$   
// -----  
//  
// 1)  $\sqrt[3]{...}$  – кубічний корінь, у C++ – функція cbrt(z).  
// 2)  $|\sin x|$  – fabs(sin(x))  
// 3)  $\tan x$  – тангенс: tan(x)  
// 4)  $5^{(...)}$  – pow(5, ...)  
// 5)  $\log_2(...)$  – двійковий логарифм: log2(...)  
cout << "Math21\n";  
double x;  
cout << "Real x (> -15): ";  
cin >> x;  
  
// Перевірка область визначення:  $x + 15 > 0$   
if (x <= -15) {  
    cout << "Error: x + 15 must be positive.\n";  
    return 1;  
}  
  
// Обчислюємо окремі частини:  
double sinx      = sin(x);  
double abssinx   = fabs(sinx);  
double tgx       = tan(x);  
double expr_inside = x*x - 2 * abssinx * 3 * tgx;  
double abs_expr   = fabs(expr_inside);  
double pow5       = pow(5.0, cos(x - 12.0));  
double numerator  = abs_expr * pow5;  
  
double denom = 0.6 + 4.0 * log2(x + 15.0);  
  
// Кубічний корінь із чисельника  
double y = cbrt(numerator) / denom;  
  
cout << "y = " << y << "\n";  
  
return 0;  
}
```

ДОДАТОК В

Діалог з ІІІ для самоаналізу

Студент: Мені потрібно розв'язати задачу: «З початку доби минуло N секунд. Знайти, скільки секунд пройшло з початку останньої хвилини». Але я не зовсім розумію, як це робиться.

ІІІ: Тут все просто. Треба знайти залишок від ділення числа N на 60, тому що одна хвилина містить 60 секунд.

Формула:

$$S = N \% 60$$

Студент: Це означає, що я беру число секунд і просто знаходжу залишок?

ІІІ: Так. Наприклад, якщо $N = 130$, то:

$$130 \% 60 = 10$$

Отже, пройшло 10 секунд з початку хвилини.

Студент: Добре, а в коді як це буде виглядати?

ІІІ: Наприклад на C++:

$$S = N \% 60;$$

Студент: Все, зрозуміла. І в блок-схемі буде: початок \rightarrow введення $N \rightarrow$ обчислення \rightarrow виведення \rightarrow кінець?

ІІІ: Так, саме так.

Студент: Дякую, тепер можу оформити звіт.

ІІІ: Завжди радий допомогти.