Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту

****

**Звіт**

Про виконання

**Лабораторних та практичних робіт №2**

***з дисципліни:*** «Мови та парадигми програмування»

***з розділу***: ««Лінійні та розгалужені алгоритми. Умовні оператори. Константи, змінні»

***Виконав***

студент групи ШІ-13

***Хотієнко Назар Олегович***

**Тема роботи:**

**Тема роботи:**

Знайомство з С/С++. Виконання програм простої структури. Використання основних операторів мови С/С++

**Мета роботи:**

Здобування професійних навичок в роботі з лінійними та розгалужними алгоритмами, умовними операторами, константантами і зміними.

**Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Управління вимогами (розуміти завдання) та проектування .
* Тема №2: Lab# programming: VNS Lab 1 Task 1.
* Тема №3: Lab# programming: VNS Lab 1 Task 2.
* Тема №4: Lab# programming: Algotester Lab 1 Task 2
* Тема №5: Practice# programming: Class Practice Task
* Тема №6: Practice# programming: Self Practice Task

1. Індивідуальний план опрацювання теорії:

* . Тема №1: Управління вимогами (розуміти завдання) та проектування .
  + Джерела Інформації
    - <https://www.visual-paradigm.com/tutorials/flowchart-tutorial/>
    - <https://www.programiz.com/article/flowchart-programming>
  + Що опрацьовано:
    - Створено діаграми для усіх заданих завдань
  + Статус: Ознайомлений
  + Початок опрацювання теми: 14.10.2023
  + Звершення опрацювання теми: 24.11.2023
* Тема №2: Lab# programming: VNS Lab 1 Task 1.
  + Джерела Інформації:
    - [https://chat.openai.com/](https://chat.openai.com/%2520)
    - Лекції
  + Що опрацьовано:
    - Створено код для розвязання Vns Lab 1 Task 1
  + Статус: Ознайомлений
  + Початок опрацювання теми: 14.10.2023
  + Звершення опрацювання теми: 24.11.2023
* Тема №3: Lab# programming: VNS Lab 1 Task 2
  + Джерела Інформації:
  + Що опрацьовано:
    - Створено код для розвязання Vns Lab 1 Task 2
  + Статус: Ознайомлений
  + Початок опрацювання теми: 14.10.2023
  + Звершення опрацювання теми: 24.11.2023
* Тема №4: Lab# programming: Algotester Lab 1 Task 2
  + Джерела Інформації:
    - <https://www.programiz.com/cpp-programming/examples/array-largest-element>
    - <https://chat.openai.com/>
  + Що опрацьовано:
    - Створено код для розвязання Algotester Lab 1 Task 2
  + Статус: Ознайомлений
  + Початок опрацювання теми: 14.10.2023
  + Звершення опрацювання теми: 24.11.2023
* Тема №5: Practice# programming: Class Practice Task
  + Джерела Інформації:
  + Що опрацьовано:
    - Створено код для розвязання Class Practise task
  + Статус: Ознайомлений
  + Початок опрацювання теми: 14.10.2023
  + Звершення опрацювання теми: 24.11.2023
* Тема №6: Practice# programming: Self Practice Task
  + Джерела Інформації:
  + Що опрацьовано:
    - Створено код для розвязання Self Practise Task
  + Статус: Ознайомлений
  + Початок опрацювання теми: 14.10.2023
  + Звершення опрацювання теми: 24.11.2023

**Виконання роботи:**

**1.** Завдання №1 Lab 1v1

У вашого персонажа є H хітпойнтів та M мани.

Персонаж 3 рази використає закляття, кожне з яких може використати хітпойнти та ману одночасно.

Якщо якесь закляття забирає і хітпойнти і ману - ваш персонаж програє, отже для виграшу треба використовувати при одному заклинанні **АБО** хітпойнти, **АБО** ману.

Якщо в кінці персонаж буде мати додатню кількість хітпойнтів та мани

(H,M>0) - він виграє, в іншому випадку програє.

Ваше завдання у випадку виграшу персонажа вивести **YES**, вивести **NO** у іншому випадку.

## Мета Задачі

Навчитися користуватися операторами галуження для структурування логіки програм.

Якщо користувач вводить будь-яку іншу умову, запропонуйте йому ввести дійсну умову.

### Використовуйте таку логіку

*if - якщо* **NO**

*else* - якщо **YES**

### Вимоги:

### **Вхідні дані**

### 2 цілих числа

### H та M

### - хітпойнти та мана персонажа

### 3 рядки по 2 цілих числа,

### hi та mi

### - кількість хітпойнтів та мани, які ваш персонаж потратить за хід на i заклинання

# **Вихідні дані**

### YES

### - якщо ваш персонаж виграє

### NO

### - у всіх інших випадках

# **Обмеження**

### 1≤H≤1012

### 1≤M≤1012

### 0≤hi≤1012

### 0≤mi≤1012

### 

### 

### 

### 

### 

### Приклад коду програми:

#include <iostream>

using namespace std;

int main()

{

long a, b;

cin >> a >> b;

long q1, q2;

cin >> q1 >> q2;

long w1, w2;

cin >> w1 >> w2;

long e1, e2;

cin >> e1 >> e2;

if (a - q1 - w1 - e1 <= 0)

{

cout << "NO";

}

else if (b - q2 - w2 - e2 <= 0)

{

cout << "NO";

}

else if (q1 > 0 and q2 > 0)

{

cout << "NO";

}

else if (w1 > 0 and w2 > 0)

{

cout << "NO";

}

else if (e1 > 0 and e2 > 0)

{

cout << "NO";

}

else

{

cout << "YES";

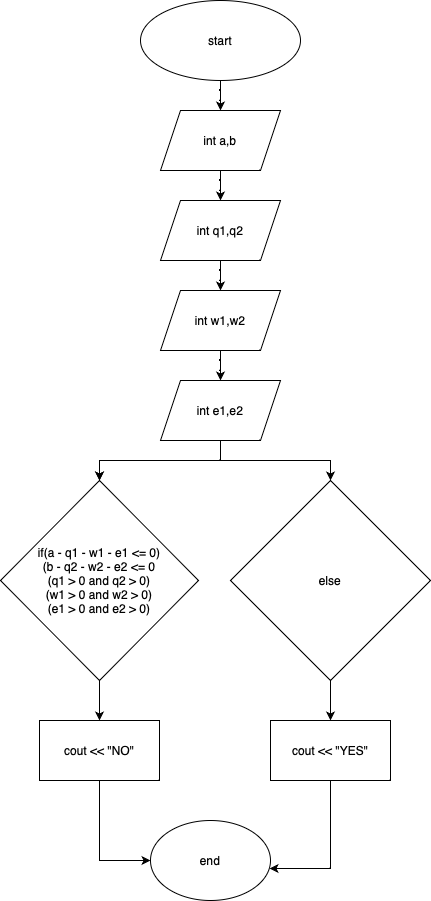
}

return (0);

}

### 

Блок схема програми:



Завдання №2 Lab 1v2

## Задача

Обмеження: 1 сек., 256 МiБ

У вас є стiл, у якого є 4 нiжки, довжини яких вам дано.

Ви хочете зробити нiжки рiвної довжини, для цього ви вiдпиляєте

d

вiд кожної нiжки (тобто

вам буде дано 4 числа, кожне з яких буде означати вiдпилювання вiд вiдповiної нiжки стола).

Якщо пiд час вiдпилювання найдовша нiжка стола буде у 2 рази бiльша-рiвна нiж найменша

нiжка - стiл перевернеться, але вiдпилювати нiжки це вам не завадить.

Тобто якщо

hmax>= 2∗hmin

то стiл перевертається.

Увага

, це може статися i мiж початком

та кiнцем вiдпилювання, наприклад коли вiдпиляють 2, але ще не встигнуть вiдпиляти 3тю нiжку.

Також ми вважаємо що перед вiдпилюванням стiл не перевернеться.

Ваше завдання сказати чи пiсля усiх манiпуляцiй стiл буде цiлий та паралельний пiдлозi.

Якщо довжина, яку вiдрiжуть буде бiльша за довжину ножки - вам треба вивести ERROR.

Увага! Навiть якщо стiл перевернеться - ви все одно вiдпилюєте нiжки i можете отримати

ERROR.

### Вимоги:

Вхiднi данi

4 цiлих числа

h 1,2,3,4 - довжини нiжок стола 4 цiлих числа

d1,2,3,4 - довжина, яку вiдпиляють

вiд вiдповiдної ножки

Вихiднi данi

YES - якщо стiл буде стояти паралельно площинi пiдлоги та довжина найменшої нiжки не

буде рiвна нулю.

ERROR - у випадку якщо ви вiдпиляєте бiльшу довжину нiж має ножка

NO - у iнших випадках

Обмеження

0≤h1,2,3,4≤1012

0≤d1,2,3,4≤1012

Приклад коду програми:

#include <iostream>

#include <algorithm>

using namespace std;

int main()

{

unsigned long a, b, c, d;

cin >> a >> b >> c >> d;

unsigned long q1, q2, q3, q4;

cin >> q1;

cin >> q2;

cin >> q3;

cin >> q4;

if (a < q1 or b < q2 or c < q3 or d < q4)

{

cout << ("ERROR");

}

else if (a - q1 >= 2 \* min(min(b, c), d))

{

cout << "NO";

}

else if (max(max(a - q1, b), max(c, d)) >= 2 \* min(min(b, c), min(a - q1, d)))

{

cout << "NO";

}

else if (b - q2 >= 2 \* min(min(a - q1, c), d))

{

cout << "NO";

}

else if (max(max(a - q1, b - q2), max(c, d)) >= 2 \* min(min(a - q1, c), min(b - q2, d)))

{

cout << "NO";

}

else if (c - q3 >= 2 \* min(min(a - q1, b - q2), d))

{

cout << "NO";

}

else if (max(max(a - q1, b - q2), max(c - q3, d)) >= 2 \* min(min(a - q1, c - q3), min(b - q2, d)))

{

cout << "NO";

}

else if (d - q4 >= 2 \* min(min(a - q1, b - q2), c - q3))

{

cout << "NO";

}

else if (max(max(a - q1, b - q2), max(c - q3, d - q4)) >= 2 \* min(min(a - q1, c - q3), min(b - q2, d - q4)))

{

cout << "NO";

}

else

{

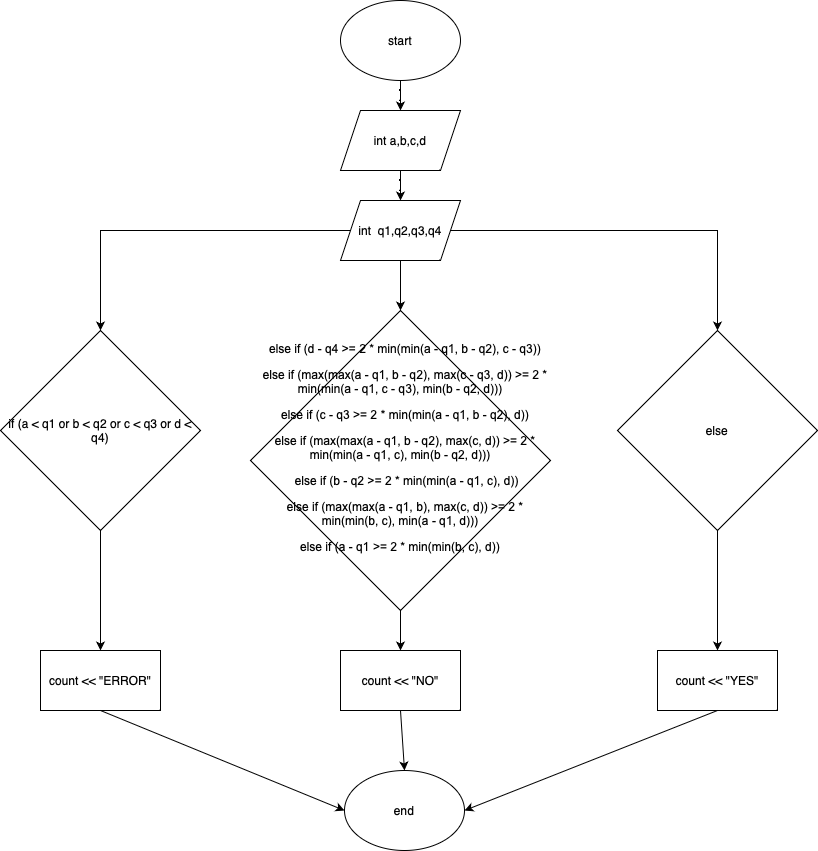
cout << "YES";

}

return (0);

}

Блок схема програми:



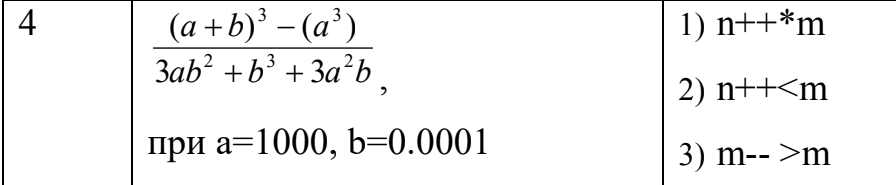
Завдання №1 VNS Lab 1v1 4 Варіант

1) Обчислити значення виразу при різних дійсних типах даних (float й double).

Обчислення варто виконувати з використанням проміжних змінних.

Порівняти й пояснити отримані результати.

2) Обчислити значення виразів. Пояснити отримані результати.



Приклад коду програми:

#include <iostream>

#include <math.h>

using namespace std;

int main()

{

double a;

double b;

cout << ("Введіть a i b з значенням double\n");

cin >> a >> b;

double c = pow(a, 3) + pow(b, 3);

double d = pow(a, 3);

double e = 3 \* a \* pow(b, 2) + 3 \* a \* pow(b, 3);

double g = c - d;

double z = g / e;

cout << (z) << endl;

float a1;

float b1;

cout << ("Введіть a i b з значенням float \n");

cin >> a1 >> b1;

double c1 = pow(a1, 3) + pow(b1, 3);

double d1 = pow(a1, 3);

double e1 = 3 \* a1 \* pow(b1, 2) + 3 \* a \* pow(b1, 3);

double g1 = c1 - d1;

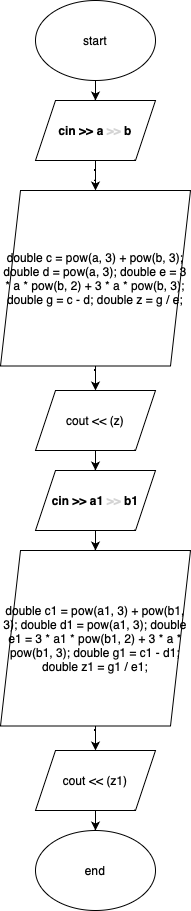
double z1 = g1 / e1;

cout << (z1) << endl;

return 0;

}

Блок схема програми:



Practice Task

John Black - Epic 2 Task 8 - Practice# programming: Class Practice Task

## Задача

Ви створюєте простий порадник щодо погоди. Користувач вводить поточні погодні умови, а програма видає рекомендації щодо активності на основі погоди.

Можливі варіанти погоди:

* sunny;
* rainy;
* cloudy;
* snowy;
* windy;

## Мета Задачі

Навчитися користуватися операторами галуження для структурування логіки програм.

Якщо користувач вводить будь-яку іншу умову, запропонуйте йому ввести дійсну умову.

### Використовуйте таку логіку

*if else* - щоб вирішити, чи повинен користувач взяти куртку чи ні.

*if, else if* - щоб надати рекомендацію щодо активності (прогулянка, футбол, настільні ігри, etc).

*switch case* - для визначення типу рекомендованого взуття.

### Деталі логіки

Рішення чи брати куртку (використовуючи if else)

* Якщо йде сніг або дощ, користувач повинен одягнути куртку.
* В іншому випадку куртка не потрібна.

Рекомендація щодо активності (використання if, else if)

* Якщо сонячно, порекомендуйте «Чудовий день для пікніка!».
* Інакше, якщо буде дощ, рекомендуємо «Ідеальна погода, щоб читати книгу всередині!».
* Інакше, якщо хмарно, рекомендуємо «Може, відвідати музей?».
* Інакше, якщо сніг, порекомендуйте «Як щодо того, щоб зліпити сніговика?».
* Інакше, якщо буде вітер, порекомендуйте «Запустіть повітряного змія, якщо він у вас є!».

Рекомендації щодо взуття (з використанням футляра для вимикача)

* sunny -> "Взуй улюблені кросівки!"
* rainy -> "Дощові чоботи - гарна ідея!"
* cloudy -> "Сьогодні підходить будь-яке взуття."
* snowy -> "Снігові черевики зігріють ваші ноги!"
* windy -> "Одягніть щось міцне!"

### Вимоги:

1. Використати всі згадані в передумові задачі оператори галуження - *if else, if, else if, switch case*;
2. За потреби комбінувати оператори;

### Приклад коду програми:

#include <cstdio>

#include <cmath>

#include <iostream>

#include <string>

int main()

{

std::string weather;

// Ask user for weather input

std::cout << "Please enter the current weather (sunny, rainy, cloudy, snowy, windy): ";

std::cin >> weather;

// Jacket decision

if (weather == "snowy" || weather == "rainy")

{

std::cout << "You should wear a jacket today.\n";

}

else

{

std::cout << "You don't need a jacket today.\n";

}

// Activity recommendation

if (weather == "sunny")

{

std::cout << "It's a great day for a picnic!\n";

}

else if (weather == "rainy")

{

std::cout << "Perfect weather to read a book inside!\n";

}

else if (weather == "cloudy")

{

std::cout << "Maybe visit a museum?\n";

}

else if (weather == "snowy")

{

std::cout << "How about building a snowman?\n";

}

else if (weather == "windy")

{

std::cout << "Fly a kite if you have one!\n";

}

// Footwear recommendation

switch (weather[0])

{ // Using the first character for switch-case

case 's': // Covers sunny and snowy

if (weather == "sunny")

{

std::cout << "Wear your favorite sneakers!\n";

}

else

{

std::cout << "Snow boots will keep your feet warm!\n";

}

break;

case 'r':

std::cout << "Rain boots are a good idea!\n";

break;

case 'c':

std::cout << "Any footwear is fine today.\n";

break;

case 'w':

std::cout << "Wear something sturdy!\n";

break;

default:

std::cout << "Invalid weather condition input.\n";

break;

}

return 0;

}

Self Practice Task

# Завдання взято з Algotester (Lab 1v3. Lab 1v3 | NULP\_LABS)

# **Lab 1v3**

*Limits: 1 sec., 256 MiB*

Персонажу по одному дають сторони 5 кубів a1..5, з яких він будує піраміду.

Коли він отримує куб з ребром ai - він його ставить на існуючий, перший ставить на підлогу (вона безмежна).

Якщо в якийсь момент об’єм куба у руці (який будуть ставити) буде більший ніж у куба на вершині піраміди - персонаж програє і гра закінчується. Розмір усіх наступних кубів після програшу **не враховується**.

Тобто якщо ai−1<ai

- це програш.

Ваше завдання - сказати як закінчиться гра.

# **Input**

5 цілих чисел a1..a5 - сторони кубів

# **Output**

Існуючі варіанти:

**LOSS** - якщо персонаж не зможе поставити куб.

**WIN** - якщо персонаж зможе поставити усі куби

**ERROR** - якщо сторона куба буде мати неможливу довжину, тобто

ai <= 0

# **Constraints**

# −1012≤ai≤1012

### Приклад коду програми:

#include <iostream>

using namespace std;

int main()

{

long q1, q2, q3, q4, q5;

cin >> q1;

cin >> q2;

cin >> q3;

cin >> q4;

cin >> q5;

if (q1 < q2 or q2 < q3 or q3 < q4 or q4 < q5)

{

cout << ("LOSS\n");

}

else if (q1 <= 0 or q2 <= 0 or q3 <= 0 or q4 <= 0 or q5 <= 0)

{

cout << ("ERROR\n");

}

else

{

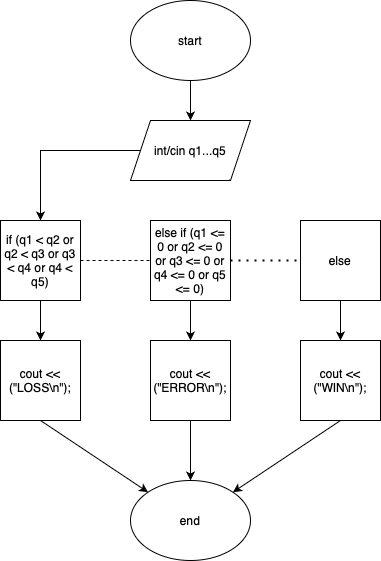
cout << ("WIN\n");

}

return (0);

}

Блок схема програми:



Висновок:

Ознайомився з практичними завданнями, навчився користуватися операторами галуження для структурування логіки програм (*if, else if, else)*, переглянув практичні матеріалию