# Міністерство освіти і науки України Львівський національний університет імені Івана Франка

**Факультет прикладної математики та інформатики**

Кафедра програмування

Лабораторна робота №11

**Масиви з індексуванням, стек, робота з матрицями**

з курсу “Архітектура обчислюваних систем та схемотехніка”

Виконав: студент групи ПМІ-16 Процай Іван Сергійович

Львів – 2024

**Тема:** Масиви з індексуванням, стек, робота з матрицями

**Мета роботи:** Використовуючи середовище Visual Studio, написати

програму опрацювання масивів даних на мові Assembler.

**Теоретичні відомості:**

1. Робота з масивами в мові Assembler:

* У мові Assembler доступ до елементів масиву зазвичай здійснюється за допомогою адресації пам'яті. Кожен елемент масиву має свою унікальну адресу, і для доступу до конкретного елементу ми завантажуємо його адресу у регістр або використовуємо операції непрямої адресації.
* Індексування масивів у мові Assembler виконується шляхом додавання до базової адреси масиву деякого зміщення, обчисленого на підставі індексу елемента та розміру елемента.

1. Операції множення та сумування:

* Операції множення та сумування виконуються за допомогою спеціальних команд процесора, які оперують над регістрами.
* Для множення двох чисел зазвичай використовуються команди типу mul або imul, які перемножають значення в одному регістрі на значення в іншому регістрі або на константу.
* Операція сумування виконується за допомогою команди add, яка додає значення одного регістра до значення іншого регістра або константи.

1. Робота з регістрами процесора:

* Регістри процесора - це невеликі швидкодіючі області пам'яті, які знаходяться прямо на процесорі.
* Вони використовуються для зберігання тимчасових даних під час виконання програми, включаючи проміжні результати обчислень.
* Операції з регістрами зазвичай виконуються швидше, ніж операції з пам'яттю, тому вони часто використовуються для оптимізації швидкодії програм.

1. Цикли в мові Assembler:

* Цикли у мові Assembler зазвичай реалізуються за допомогою інструкції loop, яка виконується доти, доки лічильник циклу не дорівнює нулю.
* Для ітерації через елементи масивів чи матриць зазвичай використовуються регістри для зберігання значень індексів та адрес елементів.

1. Стек та стекові операції:

* Стек - це структура даних, яка працює за принципом "останній увійшов, перший вийшов" (LIFO).
* У мові Assembler можна використовувати стек для зберігання адрес та проміжних значень під час виконання програми, наприклад, для зберігання адреси повернення під час виклику підпрограми або для зберігання проміжних результатів обчислень.

**Хід роботи:**

1. У середовищі Visual Studio створив простий проект на мові C++.
2. Вибравши з контекстного меню проекту пункт Build Customization та підключив інструмент Microsoft Macro Assembler, активізувавши опцію masm.
3. З використанням асемблерної вставки реалізував програму для програму для обчислення добутку матриць АВ, де А — задана матриця розмірності n x n, а елементи матриці В задаються формулою bij = i + j – 1:

**Програма, для обчислення добутку матриць.**

**Зображення, що містить текст, знімок екрана, програмне забезпечення

Автоматично згенерований опис**

**Принцип роботи:**

Фрагмент коду, що спочатку отримує від користувача розмірність квадратної матриці n, яку він вводить через консоль. Після цього відбувається створення трьох динамічних двовимірних масивів: матриці A, матриці B та результуючої матриці res, кожен розміром n на n. Кожен елемент матриці A вводиться користувачем, використовуючи цикл, тоді як елементи матриці B обчислюються за формулою bij = i + j - 1, де i та j - індекси елемента. Після формування матриць A і B їх значення виводяться на екран для перевірки коректності введених даних. Далі виконується асемблерна вставка, що множить матриці A та B, зберігаючи результат у результуючій матриці res. Нарешті, результуюча матриця res, яка містить добуток матриць A і B, виводиться на екран.

**Завдання варіанту 15 (Задана квадратна матриця A порядку n i вектори x та y розміром n. Отримати вектор A( x^3 + y^2 ).**

Зображення, що містить текст, знімок екрана, програмне забезпечення

Автоматично згенерований опис Зображення, що містить текст, знімок екрана, програмне забезпечення

Автоматично згенерований опис

**Принцип роботи**:

1. Ці функції реалізують операції обчислення куба числа, квадрату числа та суми кубів чисел з масивів, а також множення матриці на вектор, використовуючи асемблерну вставку для оптимізації швидкодії.
2. Функція calculate\_cube отримує вказівник на масив чисел x, обчислює куб кожного елемента цього масиву та зберігає результати у масиві x\_cubed.
3. Функція calculate\_square отримує вказівник на масив чисел y, обчислює квадрат кожного елемента цього масиву та зберігає результати у масиві y\_squared.
4. Функція calculate\_sum отримує вказівники на масиви x\_cubed та y\_squared, обчислює суму елементів на відповідних позиціях цих масивів та зберігає результати у масиві sum\_result.
5. Функція matrix\_vector\_multiply виконує множення матриці на вектор. Для кожного рядка матриці A вона обчислює скалярний добуток цього рядка на вектор result та зберігає результат у векторі final\_result. Це робиться у внутрішньому циклі за допомогою асемблерної вставки для оптимізації швидкодії обчислень.

**Висновок**: У цьому завданні ми вивчали використання асемблерної вставки у мові C++ з використанням середовища Visual Studio для оптимізації швидкодії обчислень. Під час виконання роботи ми розробили функції для обчислення куба числа, квадрату числа, суми кубів чисел з масивів, а також функцію для множення матриці на вектор. Ми дослідили принципи асемблерної вставки для виконання операцій множення, додавання та переміщення даних між регістрами та пам'яттю. Реалізуючи алгоритми у мові Assembler, ми змогли ефективно оптимізувати обчислення та підвищити швидкодію програми. Ця робота демонструє вміння використовувати асемблерну вставку для оптимізації обчислень та виконання складних завдань на рівні мови машинних команд у середовищі програмування C++.