Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

**(ПНИПУ)**

Факультет: Электротехнический

Кафедра: Информационные технологии и автоматизированные системы.

# Направление подготовки: Разработка программно-информационных систем.

**Лабораторная работа № 1**

Сумма положительных элементов. Умножение матриц.

Выполнил студент гр. РИС-18-1б

Гилязов Р.А.

(Фамилия, имя, отчество)

###### Проверил:

Старший преподаватель кафедры ИТАС, Щапов В.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность, Ф.И.О. руководителя от кафедры)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

**Пермь, 2020**

**Постановка задачи:**

1. Показать, в каком массиве отсортированном или не отсортированном происходит быстрее расчет суммы всех положительных элементов.
2. Показать, какой из вариантов перемножения матриц является наиболее эффективным.
3. Реализовать 1 и 2 в конфигурации Debug и Release, измерить время выполнения в этих конфигурациях.

**Код программы:**

#include <iostream>

#include <cstdlib>

#include <chrono>

#include <algorithm>

#include <ctime>

using namespace std;

typedef long long llong;

#define SIZE 10'000'000

llong\*\* create(int, int);

void runtime\_sum\_array(llong\* arr, long size)

{

llong res = 0;

chrono::time\_point<chrono::high\_resolution\_clock> beg = chrono::high\_resolution\_clock::now();

for (int i = 0; i < size; i++)

{

if (arr[i] > 0)

{

res += arr[i];

}

}

chrono::time\_point<chrono::high\_resolution\_clock> end = chrono::high\_resolution\_clock::now();

cout << "runtime == " << chrono::duration\_cast<chrono::microseconds>(end - beg).count() << " res ==" << res << endl;

}

llong\*\* random(llong\*\* arr, int str, int col)

{

if (arr == nullptr) arr = create(str, col);

srand(time(NULL));

for (int i = 0; i < str; i++)

{

for (int j = 0; j < col; j++)

{

arr[i][j] = rand() % 11;

}

}

return arr;

}

llong\*\* zero(llong\*\* arr, int str, int col)

{

if (arr == nullptr) arr = create(str, col);

for (int i = 0; i < str; i++)

{

for (int j = 0; j < col; j++)

{

arr[i][j] = 0;

}

}

return arr;

}

llong\*\* create(int str, int col)

{

llong\*\* a = new llong \* [str];

for (int i = 0; i < str; i++)

{

a[i] = new llong[col];

}

return a;

}

void multy\_ijk(llong\*\* a, int str1, int col1,

llong\*\* b, int str2, int col2)

{

if (str2 != col1) return;

llong\*\* res = nullptr;

res = zero(res, str1, col2);

chrono::time\_point<chrono::high\_resolution\_clock> beg = chrono::high\_resolution\_clock::now();

for (int i = 0; i < str1; i++)

{

for (int j = 0; j < col2; j++)

{

for (int k = 0; k < str2; k++)

{

res[i][j] += a[i][k] \* b[k][j];

}

}

}

chrono::time\_point<chrono::high\_resolution\_clock> end = chrono::high\_resolution\_clock::now();

cout << "runtime (ijk)== " << chrono::duration\_cast<chrono::microseconds>(end - beg).count() << endl;

//for (int i = 0; i < str1; i++)

//{

//

// cout << res[0][i] << " ";

//}

}

void multy\_ikj(llong\*\* a, int str1, int col1,

llong\*\* b, int str2, int col2)

{

if (str2 != col1) return;

llong\*\* res = nullptr;

res = zero(res, str1, col2);

chrono::time\_point<chrono::high\_resolution\_clock> beg = chrono::high\_resolution\_clock::now();

for (int i = 0; i < str1; i++)

{

for (int k = 0; k < str2; k++)

{

for (int j = 0; j < col2; j++)

{

res[i][j] += a[i][k] \* b[k][j];

}

}

}

chrono::time\_point<chrono::high\_resolution\_clock> end = chrono::high\_resolution\_clock::now();

cout << "runtime (ikj)== " << chrono::duration\_cast<chrono::microseconds>(end - beg).count() << endl;

//for (int i = 0; i < str1; i++)

//{

//

// cout << res[0][i] << " ";

//}

}

void multy\_kji(llong\*\* a, int str1, int col1,

llong\*\* b, int str2, int col2)

{

if (str2 != col1) return;

llong\*\* res = nullptr;

res = zero(res, str1, col2);

chrono::time\_point<chrono::high\_resolution\_clock> beg = chrono::high\_resolution\_clock::now();

for (int k = 0; k < str2; k++)

{

for (int j = 0; j < col2; j++)

{

for (int i = 0; i < str1; i++)

{

res[i][j] += a[i][k] \* b[k][j];

}

}

}

chrono::time\_point<chrono::high\_resolution\_clock> end = chrono::high\_resolution\_clock::now();

cout << "runtime (kji)== " << chrono::duration\_cast<chrono::microseconds>(end - beg).count() << endl;

//for (int i = 0; i < str1; i++)

//{

//

// cout << res[0][i] << " ";

//}

}

void multy\_kij(llong\*\* a, int str1, int col1,

llong\*\* b, int str2, int col2)

{

if (str2 != col1) return;

llong\*\* res = nullptr;

res = zero(res, str1, col2);

chrono::time\_point<chrono::high\_resolution\_clock> beg = chrono::high\_resolution\_clock::now();

for (int k = 0; k < str2; k++)

{

for (int i = 0; i < str1; i++)

{

for (int j = 0; j < col2; j++)

{

res[i][j] += a[i][k] \* b[k][j];

}

}

}

chrono::time\_point<chrono::high\_resolution\_clock> end = chrono::high\_resolution\_clock::now();

cout << "runtime (kij)== " << chrono::duration\_cast<chrono::microseconds>(end - beg).count() << endl;

//for (int i = 0; i < str1; i++)

//{

//

// cout << res[0][i] << " ";

//}

}

void multy\_jik(llong\*\* a, int str1, int col1,

llong\*\* b, int str2, int col2)

{

if (str2 != col1) return;

llong\*\* res = nullptr;

res = zero(res, str1, col2);

chrono::time\_point<chrono::high\_resolution\_clock> beg = chrono::high\_resolution\_clock::now();

for (int j = 0; j < col2; j++)

{

for (int i = 0; i < str1; i++)

{

for (int k = 0; k < str2; k++)

{

res[i][j] += a[i][k] \* b[k][j];

}

}

}

chrono::time\_point<chrono::high\_resolution\_clock> end = chrono::high\_resolution\_clock::now();

cout << "runtime (jik)== " << chrono::duration\_cast<chrono::microseconds>(end - beg).count() << endl;

//for (int i = 0; i < str1; i++)

//{

//

// cout << res[0][i] << " ";

//}

}

void multy\_jki(llong\*\* a, int str1, int col1,

llong\*\* b, int str2, int col2)

{

if (str2 != col1) return;

llong\*\* res = nullptr;

res = zero(res, str1, col2);

chrono::time\_point<chrono::high\_resolution\_clock> beg = chrono::high\_resolution\_clock::now();

for (int j = 0; j < col2; j++)

{

for (int k = 0; k < str2; k++)

{

for (int i = 0; i < str1; i++)

{

res[i][j] += a[i][k] \* b[k][j];

}

}

}

chrono::time\_point<chrono::high\_resolution\_clock> end = chrono::high\_resolution\_clock::now();

cout << "runtime (jki)== " << chrono::duration\_cast<chrono::microseconds>(end - beg).count() << endl;

//for (int i = 0; i < str1; i++)

//{

//

// cout << res[0][i] << " ";

//}

}

void part\_one() {

cout << "Part 1" << endl;

srand(time(NULL));

llong\* arr = new llong[SIZE];

for (int i = 0; i < SIZE; i++)

{

arr[i] = (llong)rand() % 101 - 50;

}

runtime\_sum\_array(arr, SIZE);

sort(arr, arr + SIZE);

runtime\_sum\_array(arr, SIZE);

}

void part\_two()

{

cout << endl << endl << "Part 2";

int str1 = 2000;

int col1 = 500;

int str2 = col1;

int col2 = 2000;

llong\*\* a = create(str1, col1);

llong\*\* b = create(str2, col2);

a = random(a, str1, col1);

b = random(b, str2, col2); // [0...10]

multy\_ijk(a, str1, col1, b, str2, col2);

multy\_ikj(a, str1, col1, b, str2, col2);

multy\_kji(a, str1, col1, b, str2, col2);

multy\_kij(a, str1, col1, b, str2, col2);

multy\_jik(a, str1, col1, b, str2, col2);

multy\_jki(a, str1, col1, b, str2, col2);

}

int main()

{

part\_one();

part\_two();

}

**Результаты тестов:**

**Таблица 1: Измерение суммы положительных элементов в массиве в конфигурации Debug.**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Время без сортировки, мкс** | **Время с сортировкой\*, мкс** | **Результат** |
| **1** | 109411 | 63481 | 126028552 |
| **2** | 95583 | 31889 | 126085733 |
| **3** | 112710 | 31817 | 126038182 |
| **4** | 96064 | 60806 | 125972891 |
| **5** | 199659 | 61217 | 125994272 |
| **6** | 201524 | 29675 | 126069913 |
| **7** | 91915 | 30508 | 126019065 |
| **8** | 92900 | 29735 | 126078815 |
| **9** | 93024 | 29733 | 126046155 |
| **10** | 92865 | 30449 | 126014473 |

**Таблица 2: Измерение времени умножения матриц (2000x500 & 500x2000) различных функций в конфигурации Debug.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Время измерения, мкс** | | | | | |
| **ijk** | **ikj** | **kji** | **kij** | **jik** | **jki** | |
| **1** | 42854934 | 22578651 | 69796417 | 24918022 | 40812177 | 49637510 | |
| **2** | 49970464 | 24477757 | 65458410 | 24808799 | 42263056 | 62153778 | |
| **3** | 50260262 | 24690857 | 67593800 | 24624268 | 42280735 | 62907217 | |
| **4** | 49980093 | 24520697 | 67962601 | 25062092 | 42318633 | 66120497 | |
| **5** | 48674861 | 16415990 | 62227313 | 24800691 | 42440217 | 63747510 | |
| **6** | 50773259 | 24628994 | 66192496 | 22406992 | 37483350 | 60243858 | |
| **7** | 46813636 | 24055648 | 63613781 | 24539228 | 40850574 | 60092280 | |
| **8** | 46994276 | 24270360 | 63787393 | 23776430 | 28172700 | 55729596 | |
| **9** | 48881576 | 23191328 | 55104787 | 11170516 | 18558027 | 32261331 | |
| **10** | 28863101 | 10837987 | 38414998 | 11197927 | 18613565 | 33548004 | |

**Таблица 3: Измерение суммы положительных элементов в массиве в конфигурации Release.**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Время без сортировки, мкс** | **Время с сортировкой\*, мкс** | **Результат** |
| **1** | 6557 | 6104 | 125979870 |
| **2** | 7608 | 6574 | 126096340 |
| **3** | 10715 | 11209 | 126049973 |
| **4** | 10796 | 10471 | 126083602 |
| **5** | 9889 | 9898 | 126089041 |
| **6** | 9951 | 10192 | 126051623 |
| **7** | 10437 | 10437 | 126093434 |
| **8** | 10636 | 9886 | 126077710 |
| **9** | 10609 | 10104 | 126093799 |
| **10** | 10099 | 10421 | 126020653 |

**Таблица 4: Измерение времени умножения матриц (2000x500 & 500x2000) различных функций в конфигурации Release.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Время измерения, мкс** | | | | | |
| **ijk** | **ikj** | **kji** | **kij** | **jik** | **jki** | |
| **1** | 18182149 | 7680960 | 30868009 | 7651896 | 12101058 | 28722129 | |
| **2** | 17635640 | 7583230 | 31074504 | 7987495 | 12501189 | 29016185 | |
| **3** | 18764230 | 7771201 | 32150042 | 7777499 | 12652875 | 28615514 | |
| **4** | 18268065 | 7763025 | 31698177 | 7853011 | 12493130 | 28238996 | |
| **5** | 18403358 | 7699516 | 31853687 | 7780404 | 12424167 | 28674208 | |
| **6** | 18377762 | 7693862 | 41150056 | 16426770 | 22141341 | 47551910 | |
| **7** | 30920855 | 16234810 | 53280684 | 16356888 | 25453983 | 51751355 | |
| **8** | 31158250 | 16468885 | 50817652 | 16757777 | 25898772 | 51513886 | |