|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

**ОТЧЕТ**

|  |  |
| --- | --- |
| **По лабораторной работе №3** |  |

**Название:** Оценка эффективности и качества программ.

**Дисциплина:** Технологии разработки программных систем.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ-42б |  |  | С.В. Астахов |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  |  |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2021

3 вариант

Задача

Написать программу, которая генерирует массив уникальных случайных чисел в диапазоне [-100;+100]

Исходный код программы:

#include <stdio.h>

#include <stdlib.h>

#include <malloc.h>

#include <time.h>

int main()

{

int N = 10, i,j,k=0,f = 0;

double \*a;

double pr = 1;

a = (double\*)malloc(N \* sizeof(int));

srand(time(NULL));

a[0] = - 100 + rand()%(100 + 100 + 1);

for (i=1; i<N; i++){

while (f == 0){

a[i] = - 100 + rand()%(100 + 100 + 1);

for (j =0; j<i;j++){

if (a[i] != a[j]) k++;

}

if (k == i) f = 1;

k=0;

}

f = 0;

}

printf("Исходный массив:\n");

for (i=0; i<N; i++){

printf("%f\n ", a[i]);

}

for (i=0; i<N; i++){

pr \*= a[i];

}

printf("Произведение:%f\n",pr);

return 0;

}

Исходная программа работает некорректно, выдавая ошибки, так как при выделении памяти под массив адресуется память под N элементов типа int (4 байта на элемент), а записываются в массив элементы типа double (8 байт на элемент), следовательно, в ходе цикла программа выходит за пределы выделенной памяти.

Исправим эту ошибку, выделив память под N элементов типа double.

a = (double\*)malloc(N \* sizeof(double));

Программа стала работать корректно. Вывод программы:

Default array:

-98.000000

-22.000000

32.000000

-97.000000

-79.000000

89.000000

26.000000

-6.000000

-10.000000

-50.000000

Product:-3670136101632000.000000

Не меняя кода, реализующего основную логику программы, напишем «обертку», позволяющую считать эффективность программы по времени и использованию памяти при выполнении ее 100.000 раз:

#include <chrono>

#include <windows.h>

#include <Psapi.h>

using namespace std;

using namespace std::chrono;

int main()

{

high\_resolution\_clock::time\_point t1 = high\_resolution\_clock::now();

PROCESS\_MEMORY\_COUNTERS memCounter1;

BOOL result = K32GetProcessMemoryInfo(GetCurrentProcess(), &memCounter1, sizeof(memCounter1));

**for (int cnt = 0; cnt < 100000; cnt++) {**

**// Исходная программа**

**}**

high\_resolution\_clock::time\_point t2 = high\_resolution\_clock::now();

duration<double> time\_span = t2 - t1;

**std::cout << "It took me " << time\_span.count() << " seconds.";**

std::cout << std::endl;

PROCESS\_MEMORY\_COUNTERS memCounter2;

result = K32GetProcessMemoryInfo(GetCurrentProcess(), &memCounter2, sizeof(memCounter2));

**std::cout << "Memory leak: " << memCounter2.WorkingSetSize - memCounter1.WorkingSetSize << std::endl;**

std::cout << "\n(press any key to exit)\n";

std::getchar();

return 0;

}

Проведем первичное измерение характеристик программы:

Затраченное время