|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра прикладной математики | | |
| Практическое задание № 1 | | |
| по дисциплине «Введение в архитектуру компьютера» | | |
| **Определение времени работы прикладных программ** | | |
|  | | |
|  | Бригада 3 | пм-23 Нос алексей |
| ПМ-23-24-25 | пм-24 параскун иван |
| Вариант 4 | пм-24 царева виктория |
|  | пм-25 таджибаев завкиддин |
|  | пм-23 лужанский александр |
|  | пм-24 кузнецова алиса |
| Преподаватели | Тракимус Юрий Викторович |
|  |  |
| Новосибирск, 2023 | | |

1. **Цель работы**

1. Изучение методики измерения времени работы подпрограммы.

2. Изучение приёмов повышения точности измерения времени работы подпрограммы.

3. Изучение способов измерения времени работы подпрограммы.

4. Получение практических навыков измерения времени работы программы.

1. **Вариант задания**

Алгоритм вычисления функции с помощью разложения в ряд по первым членам этого ряда:

Область сходимости ряда:

1. **Текст программы**

#include <math.h>

#include <stdio.h>

#include <time.h>

double \_sin(unsigned N, double x);

int main()

{

double x, sin\_x;

unsigned N;

scanf("%u %lf", &N, &x);

clock\_t stime = clock();

sin\_x = \_sin(N, x);

clock\_t etime = clock();

printf("sin(x) = %.31lf\nTime = %lfs.", sin\_x,

(double)(etime - stime) / CLOCKS\_PER\_SEC);

return 0;

}

int cast(double \*x)

{

int sgn = x < 0 ? -1 : 1;

\*x = fmod(fabs(\*x), M\_PI \* 2);

if (\*x > M\_PI)

{

sgn \*= -1;

\*x -= M\_PI;

}

\*x = \*x > M\_PI\_2 ? M\_PI - \*x : \*x;

return sgn;

}

double fact(unsigned n)

{

double r = 1;

for (unsigned i = 2; i <= n; i++)

r \*= i;

return r;

}

double \_sin(unsigned N, double x)

{

int sgn = cast(&x);

double sin = x;

for (unsigned n = 2; n <= N; n++)

{

int s = n % 2 == 0 ? -1 : 1;

sin += s \* pow(x, 2 \* n - 1) / fact(2 \* n - 1);

}

return sin \* sgn;

}

1. **Текст программы с алгоритмической оптимизацией**

#include <math.h>

#include <stdio.h>

#include <time.h>

double \_sin(unsigned N, double x);

int main()

{

double x, sin\_x;

unsigned N;

scanf("%u %lf", &N, &x);

clock\_t stime = clock();

sin\_x = \_sin(N, x);

clock\_t etime = clock();

printf("sin(x) = %.31lf\nTime = %lfs.\n", sin\_x,

(double)(etime - stime) / CLOCKS\_PER\_SEC);

return 0;

}

int cast(double \*x)

{

int sgn = x < 0 ? -1 : 1;

\*x = fmod(fabs(\*x), M\_PI \* 2);

if (\*x > M\_PI)

{

sgn \*= -1;

\*x -= M\_PI;

}

\*x = \*x > M\_PI\_2 ? M\_PI - \*x : \*x;

return sgn;

}

inline double \_sin(unsigned N, double x)

{

int sgn = cast(&x);

double sin = x, ds = x;

for (unsigned n = 2; n <= N; n++)

{

ds \*= -x \* x;

ds /= 2 \* (n - 1) \* (2 \* n - 1);

sin += ds;

}

return sin \* sgn;

}

1. **Точность вычисления числа**

|  |  |  |
| --- | --- | --- |
| **Кол-во членов ряда** |  | |
| **Без алгоритмической оптимизации** | **С алгоритмической оптимизацией** |
| 2 | **0.8**333333333333333703407674875052 | **0.8**333333333333333703407674875052 |
| 4 | **0.8414**682539682539763603585925011 | **0.8414**682539682539763603585925011 |
| 8 | **0.84147098480789**37293180956658034 | **0.84147098480789**37293180956658034 |
| 16 | **0.84147098480789650**48756572286948 | **0.84147098480789650**48756572286948 |
| калькулятор  Windows | 0.8414709848078965066525023216303 | |

1. **График зависимости времени работы программы от типа оптимизации на домашнем ПК1**
2. **Запуск программы на разных архитектурах (с алгоритмической оптимизацией + О2)**
3. **Вывод**

Измерив время работы программы и построив графики на основе тестов, можно выделить несколько особенностей:

1. Лучший показатель по времени работы программы – у программы с алгоритмической оптимизацией и оптимизацией O2 был на 99% быстрее худшего показателя по времени работы программы - без алгоритмической оптимизации и с отсутствием оптимизации при компилировании (Od).
2. При тестировании программы с алгоритмической оптимизацией и применением различных видов параметров оптимизации кода (O1, O2) различие по времени между тестами крайне мало.
3. При тестировании программы без алгоритмической оптимизации и применением различных видов параметров оптимизации кода (O1, O2) различие по времени между тестами оказалось также крайне мало.
4. Различие по времени между тестами с алгоритмической оптимизацией и параметром оптимизации Od, и между тестами с алгоритмической оптимизацией и параметрами оптимизации O1, O2, составляло примерно 0,001 секунд.
5. Время работы обеих программ линейно от количества членов ряда Фурье для вычисления значения синуса.
6. **Архитектура компьютера**

|  |  |  |  |
| --- | --- | --- | --- |
| **Характеристика** | **Домашний ПК1** | **Домашний ПК2** | **Домашний ПК3** |
| **Компилятор** | g++ (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0 | g++ (Debian 12.2.0-14) 12.2.0 | Apple clang version 14.0.3 (clang-1403.0.22.14.1) |
| **Операционная система** | Ubuntu 22.04.2 LTS (Linux 6.2.0-34-generic) | Debian GNU/Linux 12 (Linux 6.1.0-13-amd64) | macOS 13.4.1 (22F82) (Darwin 22.5.0) |
| **Процессор** | AMD Ryzen 5 2600, 3400 MHz, x86\_64 | AMD Ryzen 5 3500U, 2100 MHz, x86\_64 | Apple M2, 2,4 GHz |
| **Число ядер** | 6 | 4 | 8 (4 efficiency + 4 energy ) |
| **Число логический процессоров** | 12 | 8 | 8 |
| **Объем кэша** | L1d: 192 KiB (6 inst) L1i: 384 KiB (6 inst) L2: 3 MiB (6 inst) L3: 16 MiB (2 inst) | L1d: 128 KiB (4 inst) L1i: 256 KiB (4 inst) L2: 2 MiB (4 inst) L3: 4 MiB (1 inst) | L1d: 192 KiB + 128 KiB L1i: 128 KiB + 64 KiB L2: 16 MiB + 4 MiB L3: 8 MiB |
| **Установленная ОЗУ** | 2x: 4 GB, DIMM-DDR4, 2400 MT/s | 8 GB, SODIMM-DDR4, 2400 MT/s | 16 GB, LPDDR5 |
| **stress-ng - bogo ops/s (double, 180s)** | 97446.38 | 5542.89 | 10914.73 |
| **stress-ng - bogo ops/s (matrixprod, 180s)** | 1228.13 | 2736.06 | 15657.22 |