УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе № 7

по предмету «Архитектура компьютерной техники и операционных систем»

Вариант 7

Выполнил:

Панкратьев Е.С.

гр. 251003

Проверил:

Аврамец Д. В.

Минск 2023

# задание

Написать программу нахождения массива значений функции ***y[i]=sin(2\*PI\*i/N) i= [0, N-1]***с использованием ряда Тейлора. Пользователь задаёт значения ***N***и количество ***n***членов ряда Тейлора. Для расчета каждого члена ряда Тейлора запускается отдельный процесс и его результат (член ряда) записывается в файл. Каждый процесс выводит на экран свой ***id*** и рассчитанное значение ряда. Головной процесс суммирует все члены ряда Тейлора, и полученное значение ***y[i]*** записывает в файл. Проверить работу программы для значений ***N.***

# Схема алгоритма для процессов



Рисунок – 1



Рисунок – 2



Рисунок – 3



Рисунок – 4



Рисунок – 5

# Схема алгоритма для потоков



Рисунок – 6



Рисунок – 7



Рисунок – 8



Рисунок – 9



Рисунок – 10

# Код программы для процессов

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/wait.h>

#include <fcntl.h>

#include <errno.h>

#include <math.h>

#define PI 3.14

FILE \*input, \*tailor;

typedef struct {

int i;

int j;

int N;

} TailorElement;

float powBy(float base, int exp) {

float result = 1;

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

result \*= base;

return result;

}

long factorial(int n) {

long int result = 1;

for (int i = 1; i <= n; i++)

result \*= i;

return result;

}

void element(TailorElement \*arg) {

float cur = powBy(-1, arg->j) \* powBy(2 \* PI \* arg->i / arg->N, 2 \* arg->j + 1) / factorial(2 \* arg->j + 1);

flock(fileno(input), LOCK\_EX);

printf("y[%d]: ID = %d, func[%d] = %.25f\n", arg->i, getpid(), arg->j, cur);

fprintf(input, "%.25f\n", cur);

flock(fileno(input), LOCK\_UN);

free(arg);

}

void TailorFunction(int N, int n) {

int status;

pid\_t pid[n \* N];

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

for (int j = 0; j < n; j++) {

TailorElement \*arg = (TailorElement \*)malloc(sizeof(TailorElement));

arg->i = i;

arg->j = j;

arg->N = N;

pid[j] = fork();

if (pid[j] == 0) {

element(arg);

exit(0);

}

}

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

wait(&status);

}

rewind(input);

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

float sum = 0;

for (int j = 0; j < n; j++) {

float el;

fscanf(input, "%f", &el);

sum += el;

}

fprintf(tailor, "y[%d] = %.25f\n", i, sum);

}

}

int main(int argc, char \*argv[]) {

if (argc < 3) {

fprintf(stderr, "Error: Too few arguments!\n");

return -1;

}

input = fopen("input.txt", "w+");

tailor = fopen("tailor.txt", "w+");

TailorFunction(atoi(argv[1]), atoi(argv[2]));

fclose(input);

fclose(tailor);

return 0;

}

# Код программы для потоков

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <sys/wait.h>

#include <fcntl.h>

#include <unistd.h>

#include <pthread.h>

#define PI 3.14

FILE \*output, \*input, \*tailor;

pthread\_mutex\_t mutex = PTHREAD\_MUTEX\_INITIALIZER;

typedef struct tailorElem {

int i;

int j;

int N;

} tailorEl;

float powBy(float base, int exp) {

float result = 1;

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

result \*= base;

return result;

}

long factorial(int n) {

long int result = 1;

for (int i = 1; i <= n; i++)

result \*= i;

return result;

}

void \*element(tailorEl \*arg) {

float cur = powBy(-1, arg->j) \* powBy(2 \* PI \* arg->i / arg->N, 2 \* arg->j + 1) / factorial(2 \* arg->j + 1);

pthread\_mutex\_lock(&mutex);

printf("y[%d]: ID = %d, func[%d] = %.25f\n", arg->i, pthread\_self(), arg->j, cur);

fprintf(input, "%.25f\n", cur);

pthread\_mutex\_unlock(&mutex);

free(arg);

pthread\_exit(0);

}

void TailorFunction(int N, int n) {

int status;

pthread\_t tid[N][n];

pthread\_attr\_t attr[n];

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

pthread\_attr\_init(&attr[j]);

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

for (int j = 0; j < n; j++) {

tailorEl \*arg = (tailorEl \*)malloc(sizeof(tailorEl));

arg->i = i;

arg->j = j;

arg->N = N;

pthread\_create(&tid[i][j], &attr[j], element, arg);

}

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

pthread\_join(tid[i][j], NULL);

}

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

pthread\_attr\_destroy(&attr[j]);

rewind(input);

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

float sum = 0;

for (int j = 0; j < n; j++) {

float el;

fscanf(input, "%f", &el);

sum += el;

}

fprintf(tailor, "y[%d] = %.25f\n", i, sum);

}

}

int main(int argc, char \*argv[]) {

if (argc < 3) {

fprintf(stderr, "Error:Too few arguments!\n");

return 1;

}

input = fopen("input.txt", "w+");

tailor = fopen("tailor.txt", "w+");

TailorFunction(atoi(argv[1]), atoi(argv[2]));

fclose(input);

fclose(tailor);

return 0;

}