Министерство образования и науки Российской Федерации

Новосибирский государственный технический университет

Кафедра теоретической и прикладной информатики

Лабораторная работа №2 по дисциплине

«Управление ресурсами в вычислительных системах»

Факультет: ПМИ

Группа: ПМИ-12

Студенты: Швадченко А. В.

Михайловский М. А.

Вариант: 4

Преподаватели: Стасышин В. М., Сивак М. А.

Новосибирск

2024

1. ***Цель работы***

Изучить программные средства создания процессов, получить навыки управления и синхронизации процессов, а также простейшие способы обмена данными между процессами. Ознакомиться со средствами динамического запуска программ в рамках порожденного процесса, изучить механизм сигналов ОС UNIX, позволяющий процессам реагировать на различные события, и каналы, как одно из средств обмена информацией между процессами.

1. ***Задание***

Разработать программу, вычисляющую плотность распределения Рэлея с положительным параметром sigma в точке х по формуле f(x)=(x/sigma^2)\*exp(-x^2/(2\*sigma^2)). Для нахождения exp(-x^2/(2\*sigma^2)) программа должна породить параллельный процесс, вычисляющий эту величину путём разложения в ряд по формулам вычислительной математики.

1. ***Описание используемых структур***

**int fork();** - системный вызов fork() используется для порождения нового процесса.

**int wait(int \*status);** - ожидание завершения процесса-потомка родительским процессом.

**int exit(int status);** - завершение функционирования процесса

Аргумент status является статусом завершения, который передается отцу процесса, если он выполнял системный вызов wait().

1. ***Описание алгоритма***
2. родительский процесс создает временный файл для передачи данных;
3. родительский процесс порождает дочерний процесс, запомнив его идентификатор;
4. родительский процесс постоянно проверяет, не изменился ли временный файл;
5. обнаружив в файле новые данные, родительский процесс считывает их;
6. вычисляет значение функции и удаляет временный файл

Для вычисления использовали ряд Тейлора:

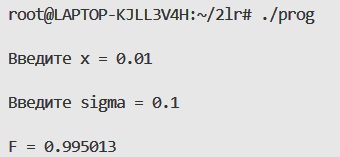
1. ***Спецификация***

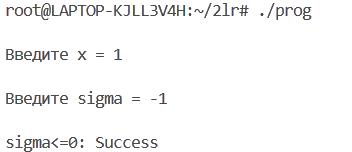
Программа находится в каталоге /root/2lr и называется prog.c.

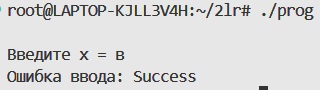
Чтобы получить исполняемый файл необходимо выполнить команду: gcc –o [имя\_исполняемого\_файла] si.c. Далее можно запустить исполняемый файл командой: ./[ имя\_исполняемого\_файла]. Далее необходимо ввести ***x*** и ***sigma.***

В результате получим значение функции плотности для заданных параметров.

1. ***Тесты***
2. Стандартный тест



1. Параметр Sigma неположителен
2. Введён недопустимый символ



1. ***Текст программы***

#include <stdio.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <fcntl.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/wait.h>

int main()

{

struct storage

{

int pid; // идентификатор потомка

double data; // значение, полученное от потомка

} st;

int i, n, fd, pid\_exp; // fd - файловый дескриптор .tmp

int a = 1, b = 1;

double f, sigma, x, exp, s = 1.0;

char file\_name[] = "file.tmp"; // файл для хранения данных

struct stat tmp\_stat;

if (!scanf("%lf", &x))

{

perror("Ошибка ввода");

exit(EXIT\_FAILURE);

}

printf("\nВведите sigma = ");

if (!scanf("%le", &sigma))

{

perror("Ошибка ввода");

exit(EXIT\_FAILURE);

}

printf("\n");

if (sigma<=0){

perror("sigma<=0");

exit(EXIT\_FAILURE);

}

fd = open(file\_name, O\_RDWR | O\_CREAT, S\_IRWXU); // открытие временного файла или создание, если его нет

printf("\nВведите x = "); // проверка входных данных

f = x / (sigma \* sigma); // вычисление первой части функции

pid\_exp = fork();

if (pid\_exp == 0) // потомок процесса

{

st.pid = getpid();

st.data = 1.0;

x = (-1)\*(x\*x)/(2\*sigma\*sigma);

for (i = 1; i < 5; i++) //вычисление второй части функции

{

st.data += x / a;

if (i % 2 == 0)

x = x \* x \* b;

else

x = x \* x;

a = a \* (a+1);

}

write(fd, &st, sizeof(st)); // запись даных в Файл

exit(0);

}

do

{

fstat(fd, &tmp\_stat);

} while (tmp\_stat.st\_size != sizeof(st));

lseek(fd, 0, SEEK\_SET); // переходим в начало файла

read(fd, &st, sizeof(st)); // чтение данных

if (st.pid == pid\_exp)

exp = st.data;

f = f \* exp; // вычисляем функцию

printf("F = %f\n", f);

close(fd); // закрываем

remove(file\_name); // удаляем

waitpid(pid\_exp, NULL, 0);

sleep(2);

return 0;

}