Лабораторная работа 4

Задание 1

Необходимые знания

1. Функция `kill`
2. Неблокирующий wait с `WNOHANG`
3. Функция `alarm`, сигнал `SIGALRM`, функция `signal`.

Дополнить программу parallel_min_max.c из ***лабораторной работы №3***, так чтобы после заданного таймаута родительский процесс посылал дочерним сигнал SIGKILL. Таймаут должен быть задан, как именной необязательный параметр командной строки (`--timeout 10`). Если таймаут не задан, то выполнение программы не должно меняться.

```
ubuntu@ubuntu:~/osis/lab3/src$ gcc -o parallel_min_max parallel_min_max.c find_min_max.c utils.c
ubuntu@ubuntu:~/osis/lab3/src$ ./parallel_min_max --seed 123 --array_size 100000 --pnum 4
Min: 36536
Max: 2147481872
Elapsed time: 106.840000ms
ubuntu@ubuntu:~/osis/lab3/src$ ./parallel_min_max --seed 123 --array_size 100000000 --pnum 4 --timeout 2
Min: 26
Max: 2147483635
Elapsed time: 123.430000ms
ubuntu@ubuntu:~/osis/lab3/src$
```

Задание 2

Необходимые знания

1. Что такое зомби процессы, как появляются, как исчезают.

Создать программу, с помощью которой можно продемонстрировать зомби процессы. Необходимо объяснить, как появляются зомби процессы, чем они опасны, и как можно от них избавиться.

```
ubuntu@ubuntu:~/osis/lab4/src$ gcc zombi_demo.c -o zombi_demo
ubuntu@ubuntu:~/osis/lab4/src$ ./zombi_demo
Parent process (PID: 36176) is waiting for child to exit...
Child process (PID: 36177) is sleeping for 2 seconds...
Child process (PID: 36177) is exiting...
Parent process (PID: 36176) is now calling wait().
Zombie process has been reaped.
ubuntu@ubuntu:~/osis/lab4/src$
```

Задание 3

Необходимые знания

1. Работа виртуальной памяти.

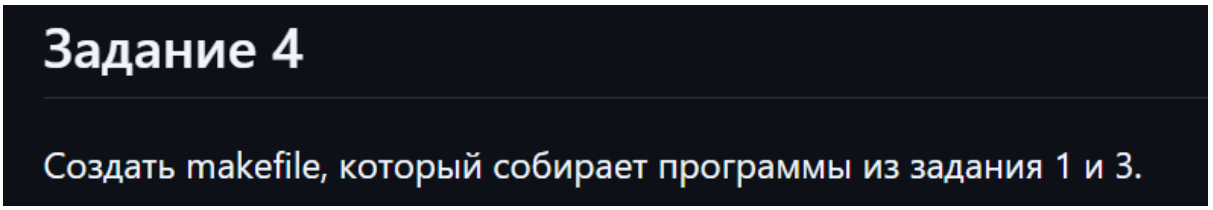
Скомпилировать process_memory.c. Объяснить, за что отвечают переменные `etext`, `edata`, `end`.

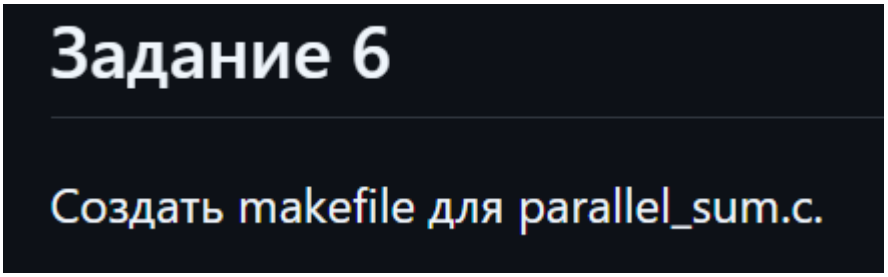
Виртуальная память — это механизм ОС, позволяющий каждому процессу работать с непрерывным адресным пространством, независимо от фрагментации физической памяти.

В process_memory.c используются специальные переменные:\

- **etext** - конец сегмента кода (где заканчивается исполняемый код).
- **edata -** конец сегмента инициализированных данных.
- **end -** конец сегмента неинициализированных данных (BSS) и начало кучи (heap).

Они помогают понять, как программа размещается в памяти.





Эти задания я объединил.

makefile

```
CC = gcc
CFLAGS = -I. -Wall -Wextra -pthread

all: parallel_min_max process_memory parallel_sum

parallel_min_max: parallel_min_max.o find_min_max.o utils.o
	$(CC) -o parallel_min_max parallel_min_max.o find_min_max.o utils.o $(CFLAGS)

process_memory: process_memory.o
	$(CC) -o process_memory process_memory.o $(CFLAGS)

parallel_sum: parallel_sum.o
	$(CC) -o parallel_sum parallel_sum.o $(CFLAGS)

parallel_min_max.o: parallel_min_max.c find_min_max.h utils.h
	$(CC) -c parallel_min_max.c $(CFLAGS)

find_min_max.o: find_min_max.c find_min_max.h utils.h
	$(CC) -c find_min_max.c $(CFLAGS)

utils.o: utils.c utils.h
	$(CC) -c utils.c $(CFLAGS)

process_memory.o: process_memory.c
	$(CC) -c process_memory.c $(CFLAGS)

parallel_sum.o: parallel_sum.c
	$(CC) -c parallel_sum.c $(CFLAGS)

clean:
	rm -f *.o parallel_min_max process_memory parallel_sum
```

Задание 5

Необходимые знания

1. POSIX threads: как создавать, как дожидаться завершения.
2. Как линковаться на бибилотеку `pthread`

Доработать parallel_sum.c так, чтобы:

- Сумма массива высчитывалась параллельно.
- Массив генерировался с помощью функции `GenerateArray` из ***лабораторной работы №3***.
- Программа должна принимать входные аргументы: количество потоков, seed для генерирования массива, размер массива (`./psum --threads_num "num" --seed "num" --array_size "num"`).
- Вместе с ответом программа должна выводить время подсчета суммы (генерация массива не должна попадать в замер времени).
- Вынести функцию, которая считает сумму в отдельную библиотеку.

```
ubuntu@ubuntu:~/osis/lab4/src$ gcc -c array_random.c -o array_random.o
ubuntu@ubuntu:~/osis/lab4/src$ gcc -c sum.c -o sum.o
ubuntu@ubuntu:~/osis/lab4/src$ gcc -c parallel_sum.c -o parallel_sum.o
ubuntu@ubuntu:~/osis/lab4/src$ gcc parallel_sum.o sum.o array_random.o -o parallel_sum -lpthread
ubuntu@ubuntu:~/osis/lab4/src$ ./parallel_sum --threads_num 4 --seed 42 --array_size 10000
Total: 497358
Elapsed time: 0.50 ms
ubuntu@ubuntu:~/osis/lab4/src$
```