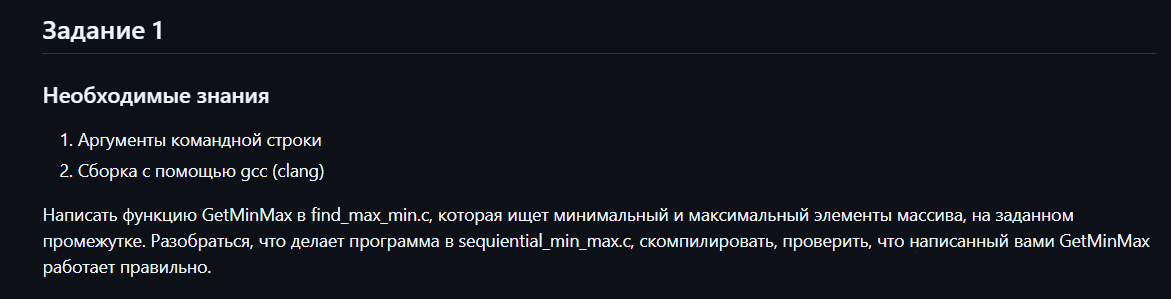
ЛР3



**1. Аргументы командной строки**

Ответ на этот вопрос есть в прошлой лабе

2. Сборка с помощью GCC/Clang

Основные команды:

Команда Описание

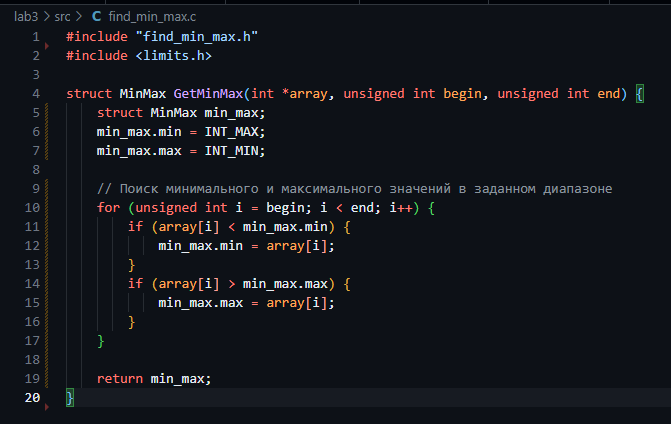
gcc main.c -o program Сборка в исполняемый файл program

gcc -c file.c Компиляция в объектный файл (file.o)

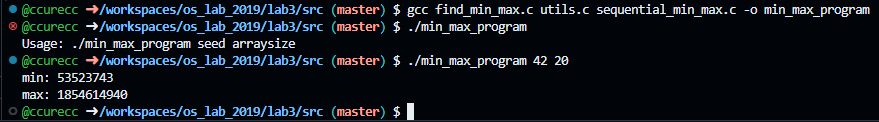
gcc file1.o file2.o -o program Линковка объектных файлов

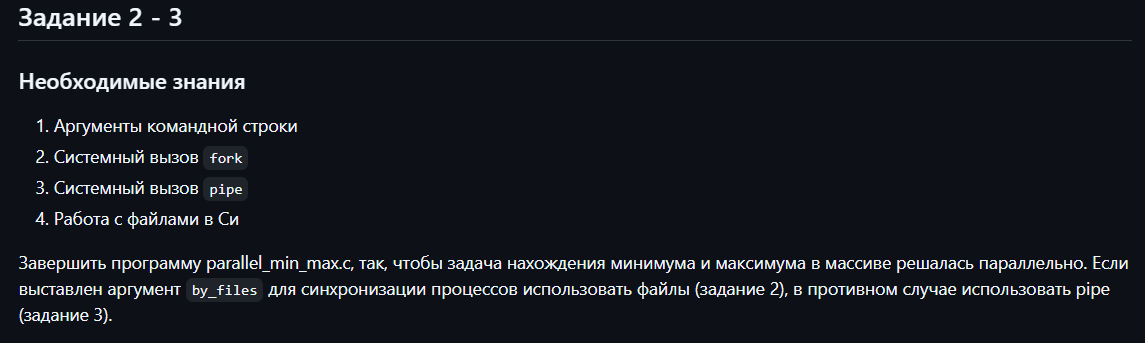
gcc -Wall -Wextra main.c Включение всех предупреждений

clang Аналогично GCC (более строгий анализ кода)



C файлом разобрался, написал подробные комментарии.





**1. Аргументы командной строки**

**Этот вопрос тоже был**

**2. Системный вызов fork()**

**Что делает:  
Создает копию процесса (дочерний процесс).**

#include <unistd.h>

#include <stdio.h>

int main() {

pid\_t pid = fork(); // Делим процесс на два

if (pid == 0) {

printf("Дочерний процесс (PID: %d)\n", getpid());

} else {

printf("Родительский процесс (PID: %d)\n", getpid());

}

}

Вывод

Родительский процесс (PID: 123)

Дочерний процесс (PID: 124)

**3. Системный вызов pipe()**

**Что делает:**  
Создает канал для обмена данными между процессами.

#include <unistd.h>

#include <stdio.h>

int main() {

int fd[2];

pipe(fd); // fd[0] — чтение, fd[1] — запись

if (fork() == 0) {

close(fd[0]); // Закрываем чтение

write(fd[1], "Hello", 6); // Пишем в канал

} else {

close(fd[1]);

char buf[6];

read(fd[0], buf, 6); // Читаем из канала

printf("Получено: %s\n", buf); // "Hello"

}

}

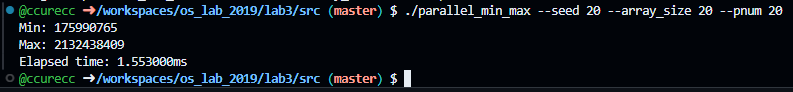
**4. Работа с файлами в C**

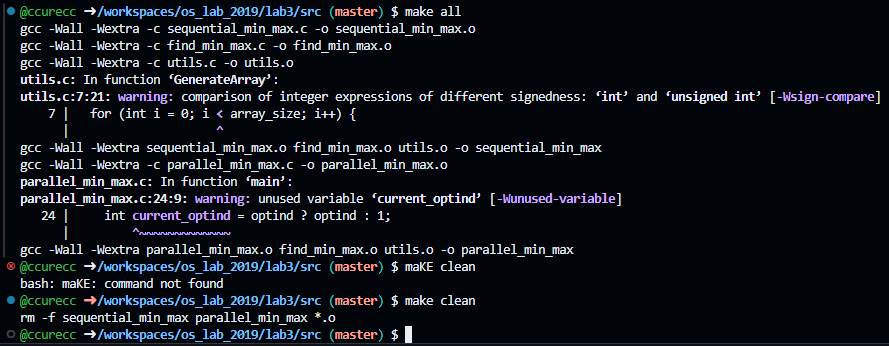
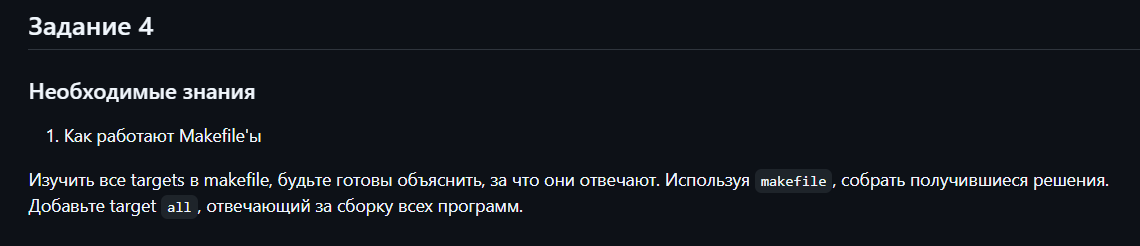
**Основные функции:**

* fopen() — открытие файла.
* fclose() — закрытие.
* fread()/fwrite() — чтение/запись.



Код написал





**1. Как работают Makefile'ы**

**Makefile** — это скрипт для автоматизации сборки проектов, который определяет:

* **Зависимости** между файлами.
* **Команды** для компиляции и линковки.
* **Правила** для очистки, тестирования и других задач.

**Зачем это нужно?**

Автоматизация — не нужно вручную компилировать каждый файл.

Инкрементная сборка — пересобираются только измененные файлы.

Управление зависимостями — например, пересборка при изменении .h-файлов.

**Автоматические переменные**

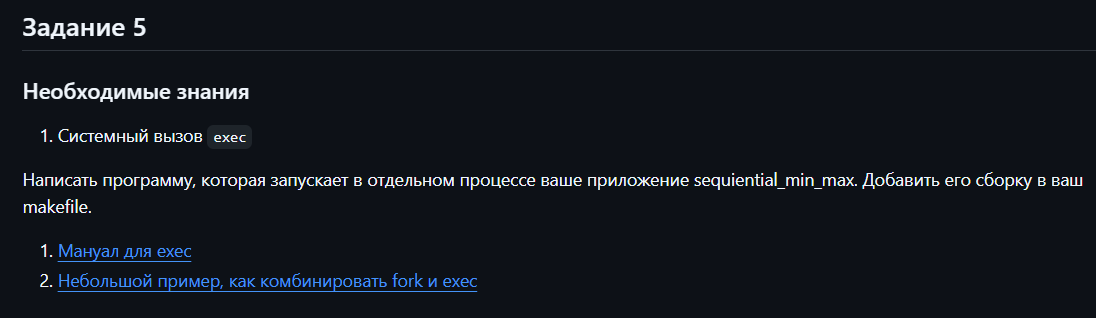
* $@ — имя цели.
* $^ — все зависимости.
* $< — первая зависимость.



Написал код для ранера

Отредактировал мейк файл

Сто раз его чинил



1. Семейство функций exec

Включает несколько вариантов:

Функция Описание

execl Принимает список аргументов (завершается NULL)

execv Принимает массив аргументов (char \*argv[])

execlp, execvp Ищут программу в PATH (не нужно указывать полный путь)

execle, execvpe Передают переменные окружения (дополнительный аргумент char \*envp[])

