Первая практика:

Задание 1: Исследование устройства памяти (30 минут)

Описание задания:

Напишите небольшую программу на языке C, которая создает массив целых чисел и заполняет его значениями от 1 до 10. Затем выведите эти значения на экран.

Требования:

• Используйте динамическое выделение памяти с помощью malloc.

• Не забудьте освободить выделенную память с помощью free.

Задание 2: Создание многофайлового проекта (30 минут)

Описание задания:

Создайте многофайловый проект, который включает в себя два файла: main.c и utils.c. В utils.c реализуйте функцию, которая принимает два числа и возвращает их сумму.

Требования:

• Создайте заголовочный файл utils.h, в котором будет объявлена функция.

• Используйте makefile для компиляции проекта.

Задание 3: Жизненный цикл программы (30 минут)

Описание задания:

Напишите краткий отчет (1-2 страницы) о жизненном цикле программного обеспечения, включая основные этапы: проектирование, разработка, тестирование, развертывание и поддержка.

Требования:

• Опишите каждый этап, его цели и важность.

• Приведите примеры реальных проектов, где каждый из этапов был критически важен.

Вторая практика:

Задание 1: Основы указателей на функции (30 минут)

1. Создание заголовочного файла function\_pointers.h:

o Определите тип указателя на функцию, который принимает два целых числа и возвращает целое число.

o Объявите функции для выполнения различных арифметических операций (сложение, вычитание, умножение, деление).

2. Создание файла реализации function\_pointers.c:

o Реализуйте функции для каждой арифметической операции.

3. Создание основного файла main.c:

o Используйте указатели на функции для выполнения операций и выводите результаты.

Задание 2: Использование Makefile (30 минут)

1. Создание Makefile:

o Напишите Makefile для компиляции проекта, который включает в себя function\_pointers.c и main.c.

2. Компиляция и запуск программы:

o Используйте команду make для компиляции и ./calculator для запуска программы.

Задание 3: Расширение функциональности (30 минут)

1. Добавление новых операций:

o Добавьте функции для вычисления остатка от деления и возведения в степень.

o Обновите заголовочный файл и файл реализации.

2. Обновление основного файла:

o Обновите массив указателей на функции, чтобы включить новые операции.

3. Тестирование:

o Проверьте, что все операции работают корректно, и результаты выводятся на экран.

Третья практика:  
Задание 1: Создание многофайлового проекта (30 минут)

Цель: Создать простой многофайловый проект на языке C, который включает в себя несколько функций.

Шаги:

• Создайте заголовочный файл math\_functions.h, который будет содержать объявления функций для выполнения арифметических операций (например, сложение, вычитание, умножение и деление).

• Создайте файл реализации math\_functions.c, в котором реализуйте эти функции.

• Создайте основной файл main.c, который будет использовать функции из math\_functions.c для выполнения операций и вывода результатов на экран.

Задание 2: Создание Makefile (30 минут)

Цель: Создать Makefile для автоматизации компиляции многофайлового проекта.

Шаги:

• Создайте файл Makefile в корневом каталоге проекта.

• Определите правила для компиляции объектных файлов и связывания их в исполняемый файл.

Задание 3: Компиляция и запуск проекта (30 минут)

Цель: Скомпилировать проект с помощью Makefile и запустить получившуюся программу.

Шаги:

• Откройте терминал и перейдите в каталог проекта.

• Выполните команду make для компиляции проекта.

• Запустите программу с помощью команды ./my\_program.

• Проверьте, что программа выводит правильные результаты.

Задание 4: Модификация и расширение проекта (30 минут)

Цель: Добавить новые функции в проект и обновить код.

Шаги:

• Добавьте новую функцию modulus для вычисления остатка от деления в math\_functions.h и math\_functions.c.

• Обновите main.c, чтобы протестировать новую функцию.

• Обновите Makefile, если это необходимо.

Четвёртая практика:  
Задание 1: Создание динамической библиотеки (30 минут)

1. Создайте файл math\_functions.c с реализацией следующих функций:

o int add(int a, int b) — возвращает сумму двух чисел.

o int subtract(int a, int b) — возвращает разность двух чисел.

o int multiply(int a, int b) — возвращает произведение двух чисел.

2. Создайте файл заголовка math\_functions.h для объявления этих функций.

3. Скомпилируйте динамическую библиотеку

Задание 2: Использование динамической библиотеки (30 минут)

1. Создайте файл main.c, который будет использовать функции из динамической библиотеки.

2. Создайте Makefile для компиляции проекта

3. Скомпилируйте и запустите проект:

Задание 3: Обсуждение и анализ (30 минут)

1. Отчёт:

o Преимущества и недостатки динамического связывания по сравнению со статическим.

o Ситуации, когда использование динамических библиотек является предпочтительным.

o Как динамическое связывание влияет на размер исполняемого файла и использование памяти.

2. Пункт анализа кода:

o Как можно улучшить код?

o Какие дополнительные функции можно добавить в библиотеку?

Шестая практика:

Задание 1: Про эмулировать на С 5 признаков ООП и тоже самое показать на С++