**Российский университит дружбы народов**

**Факультет физико-математических и естественных наук**

**Лабораторная работа № 13. Средства, применяемые при разработке программного обеспечения в ОС типа UNIX/Linux**

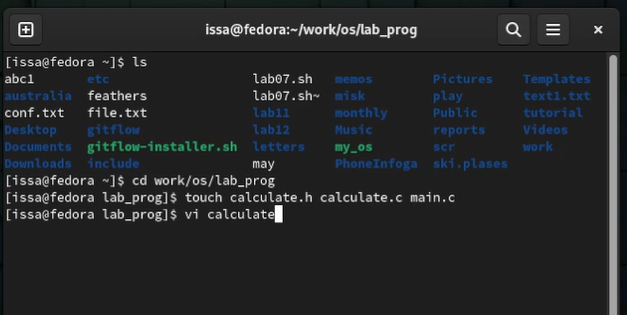
* Имя : исса гадир
* Студенческий билет : 1032218267
* Группа : нфибд-01-21

**Цель работы**

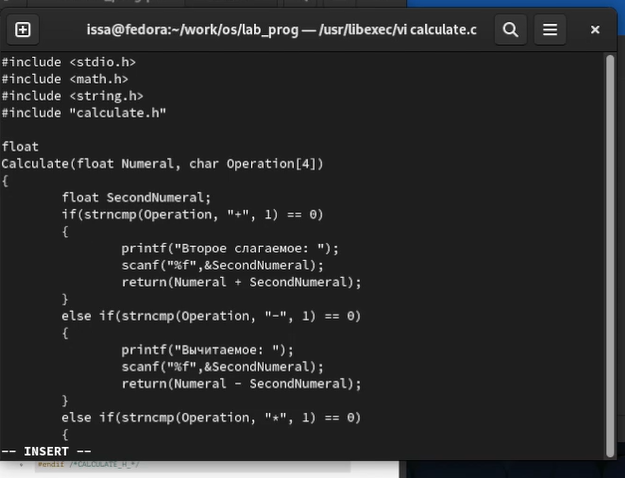
*Приобрести простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования С калькулятора с простейшими функциями*

**выполнения работы**

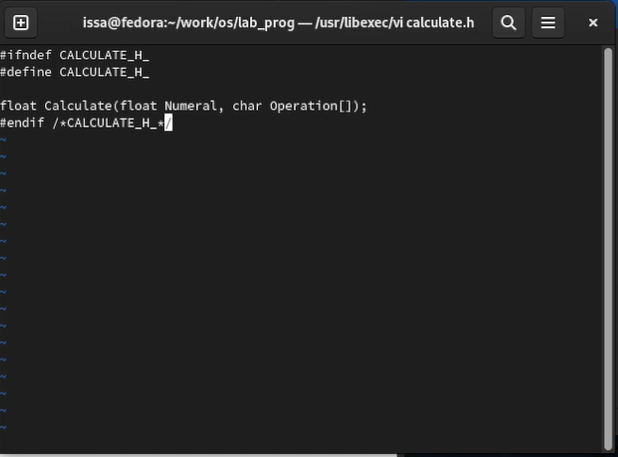
1. **В домашнем каталоге создайте подкаталог ~/work/os/lab\_prog.**

**[](https://imgbb.com/)**

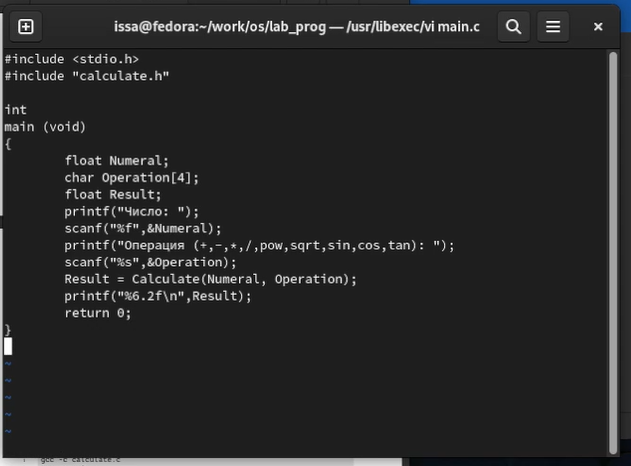
1. **Создайте в нём файлы: calculate.h, calculate.c, main.c. Это будет примитивнейший калькулятор, способный складывать, вычитать, умножать и делить, возводить число в степень, брать квадратный корень, вычислять sin, cos, tan. При запуске он будет запрашивать первое число, операцию, второе число. После этого программа выведет результат и остановится. Реализация функций калькулятора в файле calculate.h:**

**[](https://imgbb.com/)**

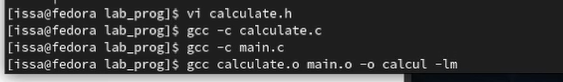
Интерфейсный файл calculate.h, описывающий формат вызова функции- калькулятора:

**[](https://imgbb.com/)**

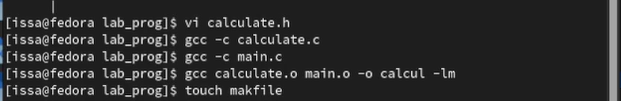
Основной файл main.c, реализующий интерфейс пользователя к калькулятору:

**[](https://imgbb.com/)**

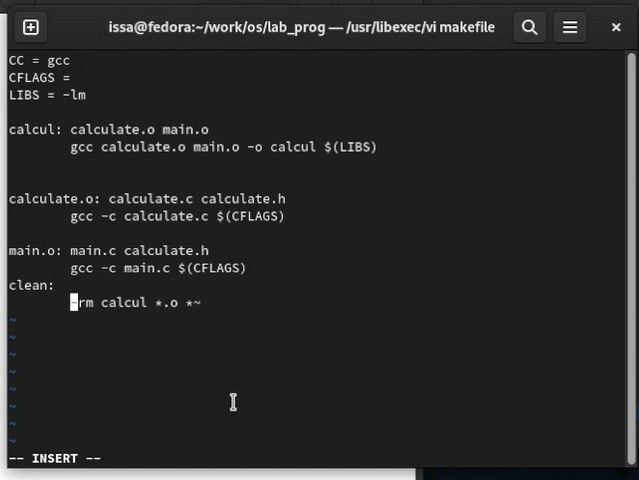
1. **Выполните компиляцию программы посредством gcc:**

**[](https://imgbb.com/)**

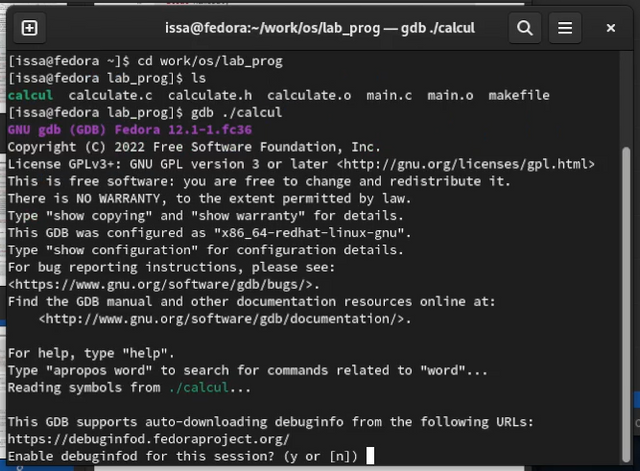
1. **При необходимости исправьте синтаксические ошибки**.

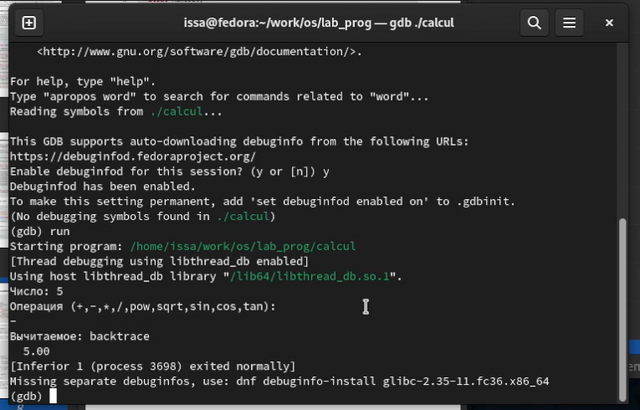
**[](https://imgbb.com/)**

1. **Создайте Makefile со следующим содержанием:**

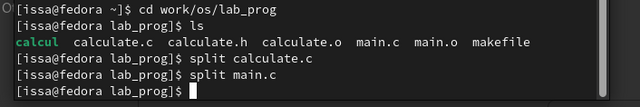
**[](https://ibb.co/Xsvt2qP)**

1. **С помощью gdb выполните отладку программы calcul (перед использованием gdb исправьте Makefile): – Запустите отладчик GDB, загрузив в него программу для отладки**

**[](https://ibb.co/z8Dtjnc)**

**[](https://ibb.co/ZK5BSq4)**

1. **С помощью утилиты splint попробуйте проанализировать коды файлов calculate.c и main.c.**

**[](https://ibb.co/ryw3JcP)**

**Контрольные вопросы**

**1.Как получить информацию о возможностях программ gcc, make, gdb и др.?**

--help

**2.Назовите и дайте краткую характеристику основным этапам разработки приложений в UNIX.**

Unix поддерживает следующие основные этапы разработки приложений: -создание исходного кода программы;- представляется в виде файла -сохранение различных вариантов исходного текста; -анализ исходного текста; Необходимо отслеживать изменения исходного кода, а также при работе более двух программистов над проектом программы нужно, чтобы они не делали изменений кода в одно время. -компиляция исходного текста и построение исполняемого модуля; -тестирование и отладка; - Проверка кода на наличие ошибок +-сохранение всех изменений, выполняемых при тестировании и отладке.

**3.Что такое суффикс в контексте языка программирования? Приведите примеры использования.**

Использование суффикса ".с" для имени файла с программой на языке Си отражает удобное и полезное соглашение, принятое в ОС UNIX. Для любого имени входного файла суффикс определяет какая компиляция требуется. Суффиксы и префиксы указывают тип объекта. Одно из полезных свойств компилятора Си — его способность по суффиксам определять типы файлов. По суффиксу .c компилятор распознает, что файл abcd.c должен компилироваться, а по суффиксу .o, что файл abcd.о является объектным модулем и для получения исполняемой программы необходимо выполнить редактирование связей. Простейший пример командной строки для компиляции программы abcd.c и построения исполняемого модуля abcd имеет вид: gcc -o abcd abcd.c. +Некоторые проекты предпочитают показывать префиксы в начале текста изменений для старых (old) и новых (new) файлов. Опция – prefix может быть использована для установки такого префикса. Плюс к этому команда bzr diff -p1 выводит префиксы в форме которая подходит для команды patch -p1.

**4. Каково основное назначение компилятора языка С в UNIX?**

Скомпилируйту файл

**5. Для чего предназначена утилита make?**

Получите файл выполнения

**6. Приведите пример структуры Makefile. Дайте характеристику основным элементам этого файла.**

Calcul: calculate.o main.o Gcc calculate.o main.o -o

элементам этого файла. Текст, следующий за точкой с запятой, и все последующие строки, начинающиеся с литеры табуляции, являются командами OC UNIX, которые необходимо выполнить для обновления целевого файла. Таким образом, спецификация взаимосвязей имеет формат: target1 [ target2...]: [:] [dependment1...] [(tab)commands] [#commentary] [(tab)commands] [#commentary], +где # — специфицирует начало комментария, так как содержимое строки, начиная с # и до конца строки, не будет обрабатываться командой make; : — последовательность команд ОС UNIX должна содержаться в одной строке make-файла (файла описаний), есть возможность переноса команд (), но она считается как одна строка; :: — последовательность команд ОС UNIX может содержаться в нескольких последовательных строках файла описаний.

**7. Назовите основное свойство, присущее всем программам отладки. Что необходимо сделать, чтобы его можно было использовать?**

‘Main’ это оден из значений его атрибута

**8. Назовите и дайте основную характеристику основным командам отладчика gdb.**

– backtrace – выводит весь путь к текущей точке останова, то есть названия всех функций, начиная от main(); иными словами, выводит весь стек функций; – break – устанавливает точку останова; параметром может быть номер строки или название функции; – clear – удаляет все точки останова на текущем уровне стека (то есть в текущей функции); – continue – продолжает выполнение программы от текущей точки до конца; – delete – удаляет точку останова или контрольное выражение; – display – добавляет выражение в список выражений, значения кото- рых отображаются каждый раз при остановке программы; – finish – выполняет программу до выхода из текущей функции; отоб- ражает возвращаемое значение,если такое имеется; – info breakpoints – выводит список всех имеющихся точек останова; – info watchpoints – выводит список всех имеющихся контрольных выражений; – list – выводит исходный код; в качестве параметра передаются название файла исходного кода, затем, через двоеточие, номер начальной и конечной строки; – next – пошаговое выполнение программы, но, в отличие от команды step, не выполняет пошагово вызываемые функции; – print – выводит значение какого-либо выражения (выражение пере- даётся в качестве параметра); – run – запускает программу на выполнение; – set – устанавливает новое значение переменной – step – пошаговое выполнение программы; +– watch – устанавливает контрольное выражение, программа остановится, как только значение контрольного выражения изменится;

**9. Опишите по шагам схему отладки программы, которую Вы использовали при выполнении лабораторной работы.**

Backtrace: вывод на экран пути к еекущей точле щстанова Break: устанавить точку останова Clear: удалить все точки останова

**10. Прокомментируйте реакцию компилятора на синтаксические ошибки в программе при его первом запуске.**

**11. Назовите основные средства, повышающие понимание исходного кода программы.**

Программисты могут использовать текстовый редактор, инструмент визуального программирования или интегрированную среду разработки (IDE), такую как комплект разработки программного обеспечения (SDK), для создания исходного кода. В больших средах разработки программ часто используются системы управления, которые помогают программистам отделять и отслеживать различные состояния и уровни файлов исходного кода.

**12. Каковы основные задачи, решаемые программой splint?**

Splint особенно хорош для проверки типов назначений переменных и функций, эффективности, неиспользуемых переменных и идентификаторов функций, недостижимого кода и возможных утечек памяти. Существует множество полезных опций, помогающих контролировать шину (см. Шина человека).