# РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

##### Факультет физико-математических и естественных наук

##### Кафедра прикладной информатики и теории вероятностей

##### ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 14

##### дисциплина: Операционные системы

Студент:

Гаглоев Олег Мелорович

Преподаватель:

Велиева Т.В.

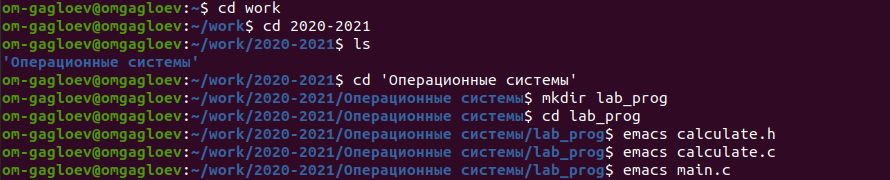
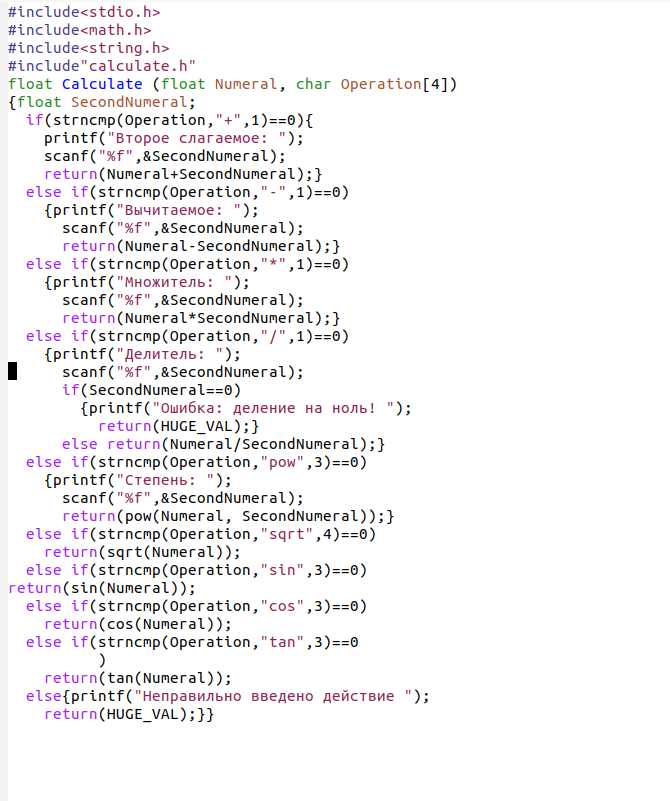
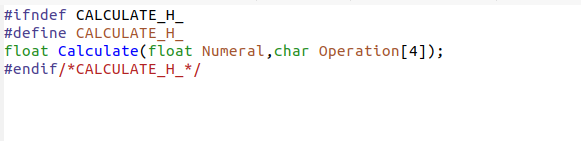
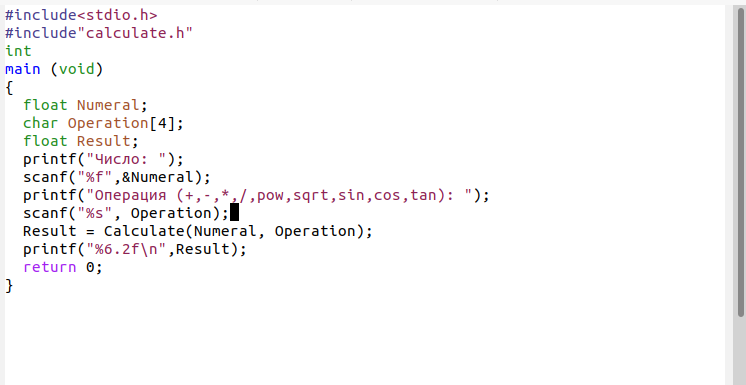
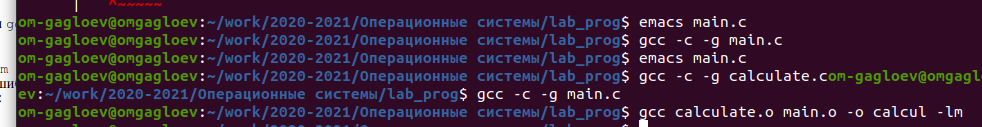
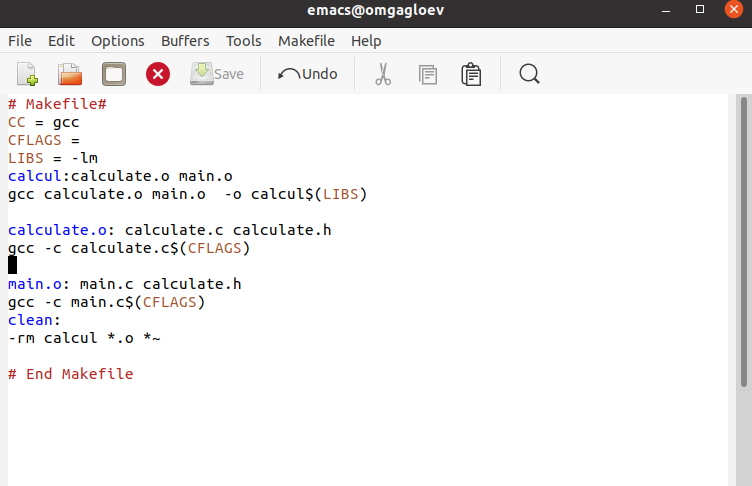
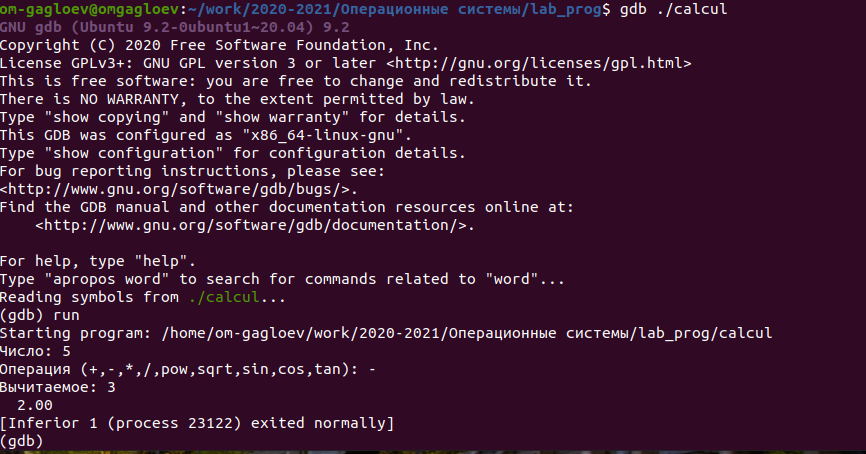
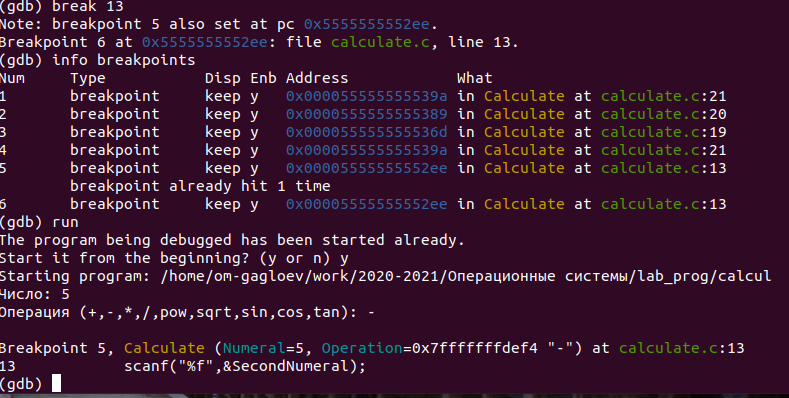
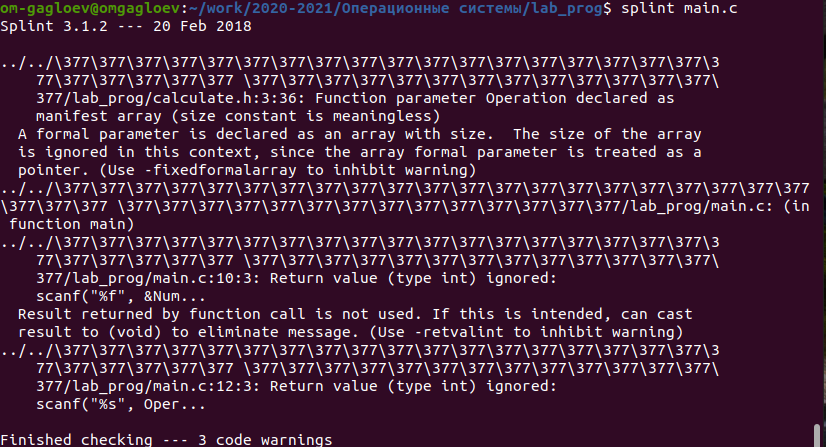
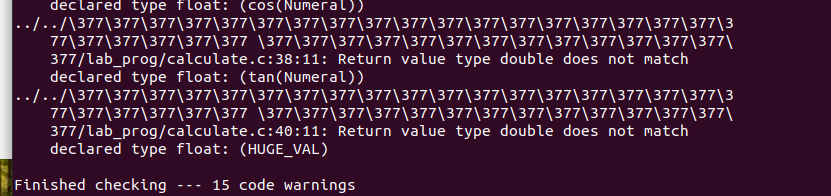
Группа: НПИбд-01-20

МОСКВА 2021 г.

##### 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.

##### Выполнение работы:

1. Создал каталог и три необходимых файла. Проделал это с помощью команды mkdir(каталог) и emacs(файлы) 
2. Скопировал и изменил код, предоставленный в задании  
3. Исправил кучу синтаксических ошибок, после чего всё заработало и выполнил компиляцию программы посредством gcc, добавив опцию -g в противном случае были проблемы с работой gdb 
4. Создал Makefile 
5. Далее с помощью gdb выполнил отладку программы ./calcul с помощью команды gdb ./calcul и для запуска программы внутри отладчика ввел команду run
6. Для постраничного (по 9 строк) просмотра исходного кода использовал команду list Для просмотра строк с 12 по 15 основного файла использовал list 12,15 Для просмотра определённых строк не основного файла использовал list calculate.c:20,29 
7. Установил точку останова в файле calculate.c на строке номер 13 break 13 и убедился в их наличии , выполнив программы и использовав команду info breakpoints  Пришлось немного понаставить точек останова, не сразу дошло, что вычитание стоит на пару строк выше
8. Использовал команду backtrace и увидел, что она выводит .Команда backtrace показал весь стек вызываемых функций от начала программы до текущего места. Далее посмотрел значение Numeral с помощью команды print Numeral после этого сравнил вывод с командой display . Видим, что значения одинаковы. Далее отображаю все точки и удаляю их с помощью команды delete n
9. Скачал утилиты splint 
10. Посмотрел и проанализировал информацию, которую он выдаёт:  Как пример: видим, что в обоих случаях у нас игнорируется значение функции scanf()

##### Вывод :

Я приобрел простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования С калькулятора с простейшими функциями. ##### Контрольные вопросы : 1) C помощью функций info и man. 2)Создание исходного кода программы, которая представляется в виде файла сохранение различных вариантов исходного текста; компиляция исходного текста и построение исполняемого модуля;тестирование и отладка;проверка кода на наличие ошибок сохранение всех изменений, выполняемых при тестировании и отладке. 3)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. [1] 4) Компиляция всей программы в целом и получении исполняемого модуля. 5) Мake-файл содержит последовательность записей (строк), определяющих зависимости между файлами. Первая строка записи представляет собой список целевых (зависимых) файлов, разделенных пробелами, за которыми следует двоеточие и список файлов, от которых зависят целевые. [1] 6) 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) Пошаговая отладка программ заключается в том, что выполняется один оператор программы и, затем контролируются те переменные, на которые должен был воздействовать данный оператор. Если в программе имеются уже отлаженные подпрограммы, то подпрограмму можно рассматривать, как один оператор программы и воспользоваться вторым способом отладки программ. Если в программе существует достаточно большой участок программы, уже отлаженный ранее, то его можно выполнить, не контролируя переменные, на которые он воздействует. Использование точек останова позволяет пропускать уже отлаженную часть программы. Точка останова устанавливается в местах, где необходимо проверить содержимое переменных или просто проконтролировать, передаётся ли управление данному оператору. Практически во всех отладчиках поддерживается это свойство (а также выполнение программы до курсора и выход из подпрограммы). Затем отладка программы продолжается в пошаговом режиме с контролем локальных и глобальных переменных, а также внутренних регистров микроконтроллера и напряжений на выводах этой микросхемы. 8)– 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 которых отображаются каждый раз при остановке программы; • finish – выполняет программу до выхода из текущей функции; отображает 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 передаётся 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) 1) Выполнили компиляцию программы 2) Увидели ошибки в программе 3) Открыли редактор и исправили программу 4) Загрузили программу в отладчик gdb 5) run — отладчик выполнил программу, мы ввели требуемые значения. 6) программа завершена, gdb не видит ошибок. 10) Не возникло – cscope - исследование функций, содержащихся в программе; – splint — критическая проверка программ, написанных на языке Си. 11) 1) Проверка корректности задания аргументов всех использованных в программе функций, а также типов возвращаемых ими значений; 2) Поиск фрагментов исходного текста, корректных с точки зрения синтаксиса языка Си, но малоэффективных с точки зрения их реализации или содержащих в себе семантические ошибки; 3) Общая оценка мобильности пользовательской программы.