### Лабораторная работа №13

Архитектура ОС

Демидович. Н. М.

6 мая 2023

Российский университет дружбы народов, Москва, Россия

### Докладчик

#### Докладчик

- Демидович Никита Михайлович
- Студент группы НКАбд-01-22
- Кафедра прикладной информатики и теории вероятностей
- Российский университет дружбы народов
- · 1132221550@pfur.ru
- https://github.com/nikdem1



Цели и задачи

#### Цель лабораторной 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.

Задачи

#### Задачи

Последовательность выполнения работы:

- 1. В домашнем каталоге создайте подкаталог ~/work/os/lab\_prog.
- 2. 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.

- 3. Выполните компиляцию программы посредством дсс.
- 4. При необходимости исправьте синтаксические ошибки.
- 5. Создайте Makefile.

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

#### gdb ./calcul

- Для запуска программы внутри отладчика введите команду run.
- Для постраничного (по 9 строк) просмотра исходного код используйте команду:

#### 1 list

• Для просмотра строк с 12 по 15 основного файла используйте list с параметрами:

#### list 12,15

• Для просмотра определённых строк не основного файла используйте list с параметрами:

#### list calculate.c:20,29

• Установите точку останова в файле calculate.c на строке номер 21:

#### list calculate.c:20,27 break 21

• Выведите информацию об имеющихся в проекте точка останова:

#### info breakpoints

- Запустите программу внутри отладчика и убедитесь, что программа остановится в момент прохождения точки останова.
- Отладчик выдаст следующую информацию:

```
#0 Calculate (Numeral=5, Operation=0x7fffffffd280 "-") at
calculate.c:21 #1 0x000000000400b2b in main () at main.c:17
```

а команда backtrace покажет весь стек вызываемых функций от начала программы до текущего места.

#### Задачи

• Посмотрите, чему равно на этом этапе значение переменной Numeral, введя:

#### print Numeral

На экран должно быть выведено число 5.

• Сравните с результатом вывода на экран после использования команды:

#### display Numeral

- Уберите точки останова.
- 7. С помощью утилиты splint попробуйте проанализировать коды файлов calculate.c и main.c.

# Теоретическое введение

#### Теоретическое введение

Процесс разработки программного обеспечения обычно разделяется на следующие этапы:

- планирование, включающее сбор и анализ требований к функционалу и другим характеристикам разрабатываемого приложения;
- проектирование, включающее в себя разработку базовых алгоритмов и спецификаций, определение языка программирования;
- непосредственная разработка приложения;
- кодирование по сути создание исходного текста программы (возможно в нескольких вариантах);
- анализ разработанного кода;
- сборка, компиляция и разработка исполняемого модуля;

#### Теоретическое введение

- тестирование и отладка, сохранение произведённых изменений;
- документирование.

Для создания исходного текста программы разработчик может воспользоваться любым удобным для него редактором текста: vi, vim, mceditor, emacs, geany и др. После завершения написания исходного кода программы (возможно состоящей из нескольких файлов), необходимо её скомпилировать и получить исполняемый модуль.

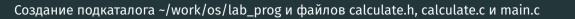
Стандартным средством для компиляции программ в ОС типа UNIX является GCC (GNU Compiler Collection). Это набор компиляторов для разного рода языков программирования (C, C++, Java, Фортран и др.). Работа с GCC производится при помощи одноимённой управляющей программы gcc, которая интерпретирует аргументы командной строки, определяет и осуществляет запуск нужного компилятора для входного файла. Файлы с расширением (суффиксом) .c воспринимаются gcc как программы на языке C, файлы с расширением .cc

 $\mu_{\Pi}\mu_{\Pi} = \mu_{\Pi}\mu_{\Pi} + \mu_{$ 

Выполнение лабораторной работы



Далее описан ход выполнения данной лабораторной работы.



На первом этапе выполнения работы я создал подкаталог ~/work/os/lab\_prog и все необходимые файлы, после чего написал программы и сделал эти файлы исполняемыми.

#### Исполняемый файл calculate.c

```
#include <stdio.h>
 #include <math.h>
 #include <string.h>
 #include "calculate.h"
float
Calculate(float Numeral, char Operation[4])
 float SecondNumeral:
if(strncmp(Operation, "+", 1) == 0)
printf("Второе слагаемое: ");
 scanf("%f",&SecondNumeral);
 return(Numeral + SecondNumeral);
```

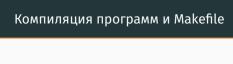
#### Исполняемый файл calculate.h

```
#ifndef CALCULATE_H_
#define CALCULATE_H_

float Calculate(float Numeral, char Operation[4]);
#endif /*CALCULATE_H_*/
```

#### Исполняемый файл main.h

```
#include <stdio.h>
#include "calculate.h"
int
main (void)
float Numeral:
char Operation[4]:
float Result;
printf("Число: ");
scanf("%f",&Numeral);
printf("Операция (+,-,*,/,pow,sqrt,sin,cos,tan): ");
scanf("%s",&Operation);
Result = Calculate(Numeral, Operation);
printf("%6.2f\n".Result):
```



Затем я выполнил компиляцию программы посредством gcc и создал Makefile.

После этого я с помощью gdb выполнил отладку программы calcul (перед использованием gdb исправил Makefile) и проверил её работу, построчно (по 9 строк). Далее просмотрел код программы, просмотрел определённые строки не основного файла и установил точку останова в файле calculate.c на строке номер 21. После этого я вывел информацию об имеющихся в проекте точках останова, программу внутри отладчика и убедился, что программа остановится в момент прохождения точки останова. Затем я посмотрел, чему равно на этом этапе значение переменной Numeral, далее сравнил с результатом вывода на экран после использования команды, убрал точки останова и с помощью утилиты splint попробуйте проанализировать коды файлов calculate.c и main.c.

## Результаты

#### Результаты

В результате выполнения данной лабораторной работы 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.

Список источников

#### Список источников

Лабораторная работа №13 (Архитектура ОС).