

Средства, применяемые при разработке программного обеспечения в ОС типа UNIX/Linux

Лабораторная работа №13

Паулу А. Ж.

Российский университет дружбы народов, Москва, Россия

Информация

- Паулу Антонью Жоау
- студент 1 курса, группа НММбд-02-22
- Российский университет дружбы народов



Вводная часть

- Командный процессор ОС UNIX
- Командные файлы

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


- Ознакомиться с теоретическим материалом.
- Выполнить упражнения.
- Ответить на контрольные вопросы.

Выполнение лабораторной работы №13

Создание файлов

```
[azpaulu@fedora ~]$ mkdir ~/work/os/lab_prog  
[azpaulu@fedora ~]$ ls ~/work/os  
lab08  lab_prog  
[azpaulu@fedora ~]$
```


```
[azpaulu@fedora ~]$ cd ~/work/os/lab_prog  
[azpaulu@fedora lab_prog]$ touch calculate.h calculate.c main.c  
[azpaulu@fedora lab_prog]$ ls  
calculate.c  calculate.h  main.c  
[azpaulu@fedora lab_prog]$
```

```
Открыть ▾  • calculate.c
~/work/ks/lab_prog

////////////////////////////////////
// calculate.c
////////////////////////////////////

#include <stdio.h>
#include <math.h>
#include <string.h>
#include "calculate.h"

float
Calculate(float Numeral, char Operation[4])
{
    float SecondNumeral;
    if(strcmp(Operation, "+") == 0)
    {
        printf("Введите второе число: ");
        scanf("%d", &SecondNumeral);
        return(Numeral + SecondNumeral);
    }
    else if(strcmp(Operation, "-") == 0)
    {
        printf("Введите второе число: ");
        scanf("%d", &SecondNumeral);
        return(Numeral - SecondNumeral);
    }
    else if(strcmp(Operation, "*") == 0)
    {
        printf("Введите второе число: ");
        scanf("%d", &SecondNumeral);
        return(Numeral * SecondNumeral);
    }
    else if(strcmp(Operation, "/") == 0)
    {
```

```
Открыть ▾  • calculate.h
~/work/ks/lab_prog

////////////////////////////////////
// calculate.h
////////////////////////////////////

#ifndef CALCULATE_H
#define CALCULATE_H

float Calculate(float Numeral, char Operation[4]);

#endif /*CALCULATE_H*/
```

```
Открыть ▾  main.c
~/work/ks/lab_prog


////////////////////////////////////
// main.c
////////////////////////////////////

#include <stdio.h>
#include "calculate.h"

int
main(void)
{
    float Numeral;
    char Operation[4];
    float Result;
    printf("Введите число: ");
    scanf("%f", &Numeral);
    printf("Введите операцию (+, -, *, /, pow, sqrt, sin, cos, tan): ");
    scanf("%s", &Operation);
    Result = Calculate(Numeral, Operation);
    printf("%0.2f\n", Result);
    return 0;
}
```

```
[azpaulu@fedora lab_prog]$ gcc -c calculate.c
[azpaulu@fedora lab_prog]$ gcc -c main.c
[azpaulu@fedora lab_prog]$ gcc calculate.o main.o -o calcul -lm
```

```
[azpaulu@fedora lab_prog]$ touch Makefile
```

Открыть  Makefile
~/work/os/lab_prog

```
#
# Makefile
#

CC = gcc
CFLAGS =
LIBS = -lm

calcul: calculate.o main.o
gcc calculate.o main.o -o calcul $(LIBS)

calculate.o: calculate.c calculate.h
gcc -c calculate.c $(CFLAGS)

main.o: main.c calculate.h
gcc -c main.c $(CFLAGS)

clean:
rm calcul *.o *~

# End Makefile
```

Работа с отладчиком

```
(gdb) run
Starting program: /home/azpaulu/work/os/lab_prog/calcul
Downloading separate debug info for /home/azpaulu/work/os/lab_prog/system-sup...
00 000 00 0000000000000000...

Downloading separate debug info for /lib64/libc.so.6...
Downloading separate debug info for /lib64/libc.so.6...
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
Начало: 2
intexpawn (*,*,*,/,pow,sqrt,sin,cos,tan): ~
Bowersecoe: 1
1.00
[Inferior 1 (process 191487) exited normally]
(gdb)
```

```
(gdb) list
Downloading source file /usr/src/debug/glibc-2.35-4.fc36.x86_64/elf/sofini.c...
1
2 /* Terminate the frame unwind info section with a 4byte 0 as a sentinel;
3    this would be the 'length' field in a real FDE. */
4
5 typedef unsigned int u132 __attribute__((mode(SI)));
6 static const u132 __FRAME_END__ =
7   __attribute__((used, section(".eh_frame")))
8   0;
(gdb)
```

```
(gdb) list 1, 4
1
2 /* Terminate the frame unwind info section with a 4byte 0 as a sentinel;
3    this would be the 'length' field in a real FDE. */
4
5 typedef unsigned int u132 __attribute__((mode(SI)));
(gdb)
```

```
breakpoint 1 (21) pending.  
(gdb) i b  
Num      Type      Disp Enb Address  What  
1        breakpoint keep y   <PENDING> 21
```

```
(gdb) run  
Starting program: /home/azpaulu/work/os/lab_prog/calcul  
Downloading separate debug info for /home/azpaulu/work/os/lab_prog/system-  
ed DSO at 0x7ffff7fc4000...  
  
Downloading separate debug info for /lib64/libc.so.6...  
Downloading separate debug info for /lib64/libc.so.6...  
[Thread debugging using libthread_db enabled]  
Using host libthread_db library "/lib64/libthread_db.so.1".
```

```
(gdb) i b  
Num      Type      Disp Enb Address  What  
1        breakpoint keep y   <PENDING> 21  
(gdb) delete 1  
(gdb) i b  
No breakpoints or watchpoints.
```

Анализ с помощью утилиты splint

```
* Ожидание в очереди...
* Ожидание аутентификации...
* Ожидание в очереди...
* Загрузка пакетов...
* Запрос данных...
* Проверка изменений...
* Установка пакетов...
```

Splint 3.1.2 --- 22 Jan 2022

calculate.h:7:37: Function parameter Operation declared as manifest array (size constant is meaningless)

A formal parameter is declared as an array with size. The size of the array is ignored in this context, since the array formal parameter is treated as a pointer. (Use -fixedformalarray to inhibit warning)

calculate.c:10:31: Function parameter Operation declared as manifest array (size constant is meaningless)

calculate.c: (in function Calculate)

calculate.c:16:5: Return value (type int) ignored: scanf("%f", &Sec...

Result returned by function call is not used. If this is intended, can cast result to (void) to eliminate message. (Use -retvalint to inhibit warning)

calculate.c:22:5: Return value (type int) ignored: scanf("%f", &Sec...

```
[azpaulu@fedora lab_prog]$ splint main.c
```

Splint 3.1.2 --- 22 Jan 2022

calculate.h:7:37: Function parameter Operation declared as manifest array (size constant is meaningless)

A formal parameter is declared as an array with size. The size of the array is ignored in this context, since the array formal parameter is treated as a pointer. (Use -fixedformalarray to inhibit warning)

main.c: (in function main)

main.c:13:3: Return value (type int) ignored: scanf("%f", &Num...

Result returned by function call is not used. If this is intended, can cast result to (void) to eliminate message. (Use -retvalint to inhibit warning)

main.c:15:14: Format argument 1 to scanf (%s) expects char * gets char [4] *: &Operation

Type of parameter is not consistent with corresponding code in format string. (Use -formattype to inhibit warning)

main.c:15:11: Corresponding format code

main.c:15:3: Return value (type int) ignored: scanf("%s", &ope...

Finished checking --- 4 code warnings

7. Во время работы над кодом программы программист неизбежно сталкивается с появлением ошибок в ней. Использование отладчика для поиска и устранения ошибок в программе существенно облегчает жизнь программиста. В комплект программ GNU для ОС типа UNIX входит отладчик GDB (GNU Debugger). Для использования GDB необходимо скомпилировать анализируемый код программы таким образом, чтобы отладочная информация содержалась в результирующем бинарном файле. Для этого следует воспользоваться опцией `-g` компилятора `gcc`: `gcc -g file.c` После этого для начала работы с `gdb` необходимо в командной строке ввести одноимённую команду, указав в качестве аргумента анализируемый бинарный файл: `gdb file.o`

6. Для работы с утилитой `make` необходимо в корне рабочего каталога с Вашим проектом создать файл с названием `makefile` или `Makefile`, в котором будут описаны правила обработки файлов Вашего программного комплекса. В самом простом случае `Makefile` имеет следующий синтаксис: `... : ... <команда 1> ...` Сначала задаётся список целей, разделённых пробелами, за которым идёт двоеточие и список зависимостей. Затем в следующих строках указываются команды. Строки с командами обязательно должны начинаться с табуляции. В качестве цели в `Makefile` может выступать имя файла или название какого-то действия. Зависимость задаёт исходные параметры (условия) для достижения указанной цели. Зависимость также может быть названием какого-то действия. Команды – собственно действия, которые необходимо выполнить для достижения цели. Общий синтаксис

Результаты

В ходе выполнения были приобретены простейшие навыки разработки, анализа, тестирования и отладки приложений в ОС типа UNIX/Linux на примере создания на языке программирования С калькулятора с простейшими функциями.