

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

Лабораторная работа №13

Азимов М.

4 мая 2023

Российский университет дружбы народов, Москва, Россия

Информация

- Азимов Миразим
- студент 1 курса, группа НММбд-01-22
- 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

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

```
[mazimov@fedora lab_prog]$ touch calculate.h calculate.c main.c  
[mazimov@fedora lab_prog]$ ls  
calculate.c calculate.h main.c
```

```
Открыть ▾  calculate.c
~/work/oclab_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(strncmp(Operation, "+", 1) == 0)
    {
        printf("Второе слагаемое: ");
        scanf("%f", &SecondNumeral);
        return(Numeral + SecondNumeral);
    }
    else if(strncmp(Operation, "-", 1) == 0)
    {
        printf("Вычитаемое: ");
```

```
Открыть ▾  calculate.h
~/work/oclab_prog

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

#ifndef CALCULATE_H_
#define CALCULATE_H_

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

#endif /*CALCULATE_H_*/
```

```
Открыть ▾  main.c
~/work/oclab_prog

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


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

int
main (void)
{
    float Numeral;
    char Operation[4];
    float Result;
    printf("Введите: ");
    scanf("%f", &Numeral);
    printf("Операция (+, -, *, /, pow, sqrt, sin, cos, tan): ");
    scanf("%s", &Operation);
    Result = Calculate(Numeral, Operation);
    printf("N6.2f\n", Result);
    return 0;
}
```

Компиляция и Makefile

```
(mazinov@fedora lab_prog)$ gcc -c calculate.c
(mazinov@fedora lab_prog)$ gcc -c main.c
(mazinov@fedora lab_prog)$ gcc calculate.o main.o -o calcul -lm
```

```
(mazinov@fedora lab_prog)$ touch Makefile
```

Открыть ▾  **makefile**
~/work/os/lab_prog

```
#
# Makefile
#

CC = gcc
CFLAGS = -g
LIBS = -lm

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

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

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

clean:
-rm calcul *.o *~

# End Makefile
```

Работа с отладчиком

```
[mazimov@fedora lab_prog]$ gdb ./calcul
GNU gdb (GDB) Fedora 11.2-3.fc36
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software; you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./calcul...

This GDB supports auto-downloading debuginfo from the following URLs:
https://debuginfod.fedoraproject.org/
Enable debuginfod for this session? (y or [n]) y
debuginfod has been enabled.
```

```
(gdb) run
Starting program: /home/mazimov/work/os/lab_prog/calcul
Downloading separate debug info for /home/mazimov/work/os/lab_prog/system-suppl
ed DSO at 0x7ffff7fc4898...

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".
Hecho: 5
Shaparea (*,*,*,./pow,sqrt,sin,cos,tan): =
Buenamano: 4
1.98
[Inferior 1 (process 7584) exited normally]
```

```
(gdb) list
Downloading source file /usr/src/debug/glibc-2.35-4.fc36.x86_64/glibc/nfmini.c...
1 /* Terminate the frame unwind info section with a 4byte 0 as a sentinel;
2 this would be the 'length' field in a real FDE. */
3
4 typedef unsigned int u32 __attribute__((mode(SI)));
5 static const u32 __FRAME_END__[1]
6 __attribute__((used, section(".eh_frame")))
7 = { 0 }
```

Работа с отладчиком

```
(gdb) list 2,5
2      this would be the 'length' field in a real FDE.  */
3
4      typedef unsigned int u132 __attribute__((mode(SI)));
5      static const u132  FRAME_END  = {}
```

```
(gdb) break 21
No line 21 in the current file.
Make breakpoint pending on future shared library load? (y or [n]) y
Breakpoint 1 (21) pending.
```

```
16 (gdb) backtrace
#0  _dl_call_libc_early_init (libc_map=0x7ffff7faf500,
    initial=initial_entry+true) at dl-call-libc-early-init.c:27
#1  0x00007ffff7f02000 in dl_main (phdr=<optimized out>,
    phnum=<optimized out>, user_entry=<optimized out>, auxv=<optimized out>)
    at rld.c:2534
#2  0x00007ffff7f02000 in _dl_sysdep_start (
    start_argptr=start_argptr_entry=0x7ffff7f02000,
    dl_main=dl_main_entry=0x7ffff7f02000 <dl_main>) at ../elf/dl-sysdep.c:256
#3  0x00007ffff7f02000 in _dl_start_final (arg=0x7ffff7f02000) at rld.c:587
#4  _dl_start (arg=0x7ffff7f02000) at rld.c:1596
#5  0x00007ffff7f02000 in _start () from /lib64/ld-linux-x86-64.so.2
#6  0x0000000000000001 in ?? ()
#7  0x00007ffff7f02000 in ?? ()
#8  0x0000000000000000 in ?? ()
```

Анализ с помощью утилиты splint

```
calculate.c:28:5: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:34:5: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:35:8: Dangerous equality comparison involving float types:
    SecondNumeral == 0
    Two real (float, double, or long double) values are compared directly using
    == or != primitive. This may produce unexpected results since floating point
    representations are inexact. Instead, compare the difference to FLT_EPSILON
    or DBL_EPSILON. (Use -realcompare to inhibit warning)
calculate.c:38:13: Return value type double does not match declared type float:
    (HUGE_VAL)
    To allow all numeric types to match, use +relaxtypes.
calculate.c:46:5: Return value (type int) ignored: scanf("%f", &Sec...
calculate.c:47:11: Return value type double does not match declared type float:
    (pow(Numeral, SecondNumeral))
calculate.c:50:11: Return value type double does not match declared type float:
    (sqrt(Numeral))
calculate.c:52:11: Return value type double does not match declared type float:
    (sin(Numeral))
calculate.c:54:11: Return value type double does not match declared type float:
    (cos(Numeral))
calculate.c:56:11: Return value type double does not match declared type float:
```

```
[mazimov@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
```

1. Чтобы получить информацию о возможностях программ `gcc`, `make`, `gdb` и др. нужно воспользоваться командой `man` или опцией `-help (-h)` для каждой команды.
2. 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 (возмож- но в нескольких вариантах); – анализ разработанного кода; – сборка, компиляция и разработка исполняемого модуля; – тестирование и отладка, сохранение произведённых изменений; • доку-

4. Основное назначение компилятора языка Си в UNIX заключается в компиляции всей программы и получении исполняемого файла/модуля.
5. Для сборки разрабатываемого приложения и собственно компиляции полезно воспользоваться утилитой `make`. Она позволяет автоматизировать процесс преобразования файлов программы из одной формы в другую, отслеживает взаимосвязи между файлами.
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

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.