Министерство образования и науки РФ
Новосибирский государственный технический университет

Кафедра ТПИ

Лабораторная работа № 6
По Операционным системам и компьютерным сетям

Инструментальные средства разработки программ

Факультет: ПМИ
Группа: ПМ-24
Бригада: 12
Студенты: Герасименко Вадим
Параскун Иван

Преподаватель: Сивак М. А.

Новосибирск
2024

1. Цель работы

Целью работы является изучение основных этапов разработки и отладки приложений в ОС Linux, а также приобретение практических навыков по использованию инструментальных средств фонда свободного программного обеспечения при компиляции исходного кода, сборке, отладке и тестировании программ, написанных на языке С.

2. Ход работы

1. Выберите из Приложения 2 программу в соответствии с номером бригады и скопируйте ее в файл ~/workdir/mainprog.c

```
[pmi-b2412@pmi-srv-openscaler workdir]$ vim mainprog.c
[pmi-b2412@pmi-srv-openscaler workdir]$ cat mainprog.c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

#define MAX_LENGTH 256

int get_word(char*string, int num)
{
    int count = 0;
    int end = strlen(string);
    for (int i = 0; i < end - 1; i++)
    {
        if (count == num)
            return i;
        if (string[i] == ' ')
            count++;
    }
    return -1;
}
int count_words(char*string)
{
    int length = strlen(string);
    int first_word_start = get_word(string, 0);
    int second_word_start = get_word(string, 1);
    int count = 0;
    for (int i = 0; i < second_word_start; i++)
        if (strncmp(string + i, string + second_word_start, length - second_word_start) == 0)
            count++;
    return count;
}

int main(int argc, char* argv[])
{
    char string[MAX_LENGTH];
    while (fgets(string, MAX_LENGTH, stdin) != (char*)NULL)
    {
        printf("'%d'\n", count_words(string));
    }
    return 0;
}
```

2. С помощью редактора vi в рабочем каталоге сервера создайте make-файл, а в локальном репозитории создайте новую ветку lab_6. Основные сведения по редактору приведены в приложении 1.

Создание make-файла:

```
#  Standart compile mainprog.c
prog: mainprog.c
        gcc -o mainprog mainprog.c
# Debugging compile mainprog.c
testprog: mainprog.c
        gcc -o test_mainprog -g mainprog.c
# End Makefile

"makefile" 9L, 209B
```

Создание новой ветки:

```
[pmi-b2412@pmi-srv-openscaler workdir]$ git checkout -b lab_6
Переключено на новую ветку «lab_6»
```

3. С помощью make-файла выполните компиляцию файла mainprog.c, используя правило prog. Обнаруженные при компиляции ошибки исправьте с помощью редактора vi. После каждого исправления измененную версию программы сохраните в ветке lab_6 с поясняющим комментарием. Сведения об ошибках занесите в отчет (номер строки, значение строки до устранения и после устранения ошибки и пояснения).

Ошибок при компиляции не обнаружено:

```
[pmi-b2412@pmi-srv-openscaler workdir]$ make prog
gcc -o mainprog mainprog.c
[pmi-b2412@pmi-srv-openscaler workdir]$
```

4. Запустите исполняемый файл mainprog и проверьте программу на подготовленных наборах тестовых данных, результаты тестирования занесите в отчет.

```
[pmi-b2412@pmi-srv-openscaler workdir]$ ./mainprog
banana ban
'0'
testtesttest test
'0'
```

Ожидаемые результаты: 1 и 3 соответственно.

Вывод: программа работает неверно, выдает некорректный результат и не прекращает выполнение.

5. При обнаружении семантических ошибок перекомпилируйте программу с помощью правила testprog make-файла и запустите отладчик gdb, с помощью которого найдите причины появления ошибок. Проверьте программу на всех наборах тестовых данных. Сведения о найденных ошибках занесите в отчет, исправленную версию программы сохраните в репозитории с поясняющим комментарием.

Отладка:

```
(gdb) next
23          int first_word_start = get_word(string, 0);
(gdb) next
24          int second_word_start = get_word(string, 1);
(gdb) next
25          int count = 0;
(gdb) info locals
length = 14
first_word_start = 0
second_word_start = 9
count = 0
(gdb) next
26          for (int i = 0; i < second_word_start; i++)
(gdb) next
27              if (strncmp(string + i, string + second_word_start, length - second_word_start) == 0)
(gdb) info locals
i = 0
length = 14
first_word_start = 0
second_word_start = 9
count = 0
(gdb)
```

Ясно, что ошибка возникает в строке 27 из-за неправильного расчёта длины входящего слова.

Для исправления ошибки необходимо в последнем переданном аргументе отнять единицу.

Исправленная версия функции count_words:

```
int count_words(char*string)
{
    int length = strlen(string);
    int first_word_start = get_word(string, 0);
    int second_word_start = get_word(string, 1);
    int count = 0;
    for (int i = 0; i < second_word_start; i++)
        if (strncmp(string + i, string + second_word_start, length - second_word_start - 1) == 0)
            count++;
    return count;
}
```

Результаты:

```
[pmi-b2412@pmi-srv-openscaler workdir]$ vim mainprog.c
[pmi-b2412@pmi-srv-openscaler workdir]$ make prog
gcc -o mainprog mainprog.c
[pmi-b2412@pmi-srv-openscaler workdir]$ ./mainprog
testtesttest test
'3'
banana ban
'1'
```

Теперь программа работает корректно.

6. Выведите журнал список всех изменений файла mainprog.c, выполненных в ходе отладки программы, занесите список в отчет.

```
[pmi-b2412@pmi-srv-openscaler workdir]$ git log -- mainprog.c
commit 20569f9d907c08b68cac591b1a69063131bc2226 (HEAD -> lab_6)
Author: Gerasimenko <v.gerasimenko.2022@stud.nstu.ru>
Date:   Tue May 14 14:26:01 2024 +0700

    Исправлено количество сравниваемых символов в функции strncmp

commit 877750cf44cb6d681dd281a2bdb9ee03a99f3e37
Author: Gerasimenko <v.gerasimenko.2022@stud.nstu.ru>
Date:   Mon May 13 18:21:52 2024 +0700

    Начальная версия программы из методического пособия
[pmi-b2412@pmi-srv-openscaler workdir]$
```

7. Определите размер исполняемого модуля отлаженной программы. Удалите всю отладочную информацию и снова определите размер исполняемого модуля, сравните с предыдущим результатом, результат сравнения занесите в отчет и подтвердите скриншотом.

```
[pmi-b2412@pmi-srv-openscaler workdir]$ ls -lh mainprog
-rwxr-xr-x 1 pmi-b2412 pmi-b2412 17K мая 14 13:04 mainprog
[pmi-b2412@pmi-srv-openscaler workdir]$ strip -s mainprog
[pmi-b2412@pmi-srv-openscaler workdir]$ ls -lh mainprog
-rwxr-xr-x 1 pmi-b2412 pmi-b2412 15K мая 14 14:28 mainprog
```

17Кб и 15Кб.

8. Выполните разбиение программы mainprog на функции в соответствии с номером бригады из таблицы 3. Здесь prog1() и prog2() – условные имена функций, используемых в программе. Обратите внимание на тип функции (внутренняя или внешняя), тип файла (.c, .h или .o) и тип модуля (исходный или объектный). Занесите в отчет измененный текст программы.

Условие:

№ бригады	main()		prog1()		prog2()	
12	внутренний, исходный	mainprog.c	внешний, исходный	prog1.c	внешний, объектный	prog2.o

mainprog.c:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

#define MAX_LENGTH 256

extern int count_words(char*string);

int main(int argc, char* argv[])
{
    char string[MAX_LENGTH];
    while (fgets(string, MAX_LENGTH, stdin) != (char*)NULL)
    {
        printf("'%d'\n", count_words(string));
    }
    return 0;
}
```

prog1.c:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>


extern int get_word(char*string, int num);

int count_words(char *string)
{
   int length = strlen(string);
   int first_word_start = get_word(string, 0);
   int second_word_start = get_word(string, 1); //second -> first
   int count = 0;
   for (int i = 0; i < second_word_start; i++)
      if (strncmp(string + i, string + second_word_start, length - second_word_start - 1) == 0)
         count++;
   return count;
}
```

prog2.o:

```
[pmi-b2412@pmi-srv-openscaler workdir]$ gcc -c prog2.c
[pmi-b2412@pmi-srv-openscaler workdir]$ cat prog2.o
ELF>`@@
UH??H?? H?}??u??E?H?E?H????E??E??)?E?;E?u?E??,?E?Hc?H?E?H??< u?E??E??E???9E?/??????GCC: (GNU) 10.3.1 GNU??zRx
d                                                                                                    iA?C
??      iprog2.cget_wordstrlen???????? .symtab.strtab.shstrtab.rela.text.data.bss.comment.note.GNU-stack.note.gnu.property.rela.eh_frame @i?
&??10?:?J?b?]@?
(x
```

9. Выполните сборку программы в соответствии с вариантом задания (см. табл.3), используя неявный вызов компоновщика и задав имя исполняемого файла mainprog_1, проверьте корректность работы программы и занесите в отчет результаты ее тестирования.

   Программа работает корректно.

```
[pmi-b2412@pmi-srv-openscaler workdir]$ gcc mainprog.c prog1.c prog2.o -o mainprog_1
[pmi-b2412@pmi-srv-openscaler workdir]$ ./mainprog_1
banana ban
'1'
testtesttest test
'3'
```

10. Выполните сборку программы в соответствии с вариантом задания (см. табл.3), используя явный вызов компоновщика. Результатом сборки должны быть исполняемый файл mainprog_2 и карта памяти progmap; проверьте корректность работы программы и занесите в отчет результаты ее тестирования.

    Явный вызов компоновщика:

```
[pmi-b2412@pmi-srv-openscaler workdir]$ ld -dynamic-linker  /lib64/ld-linux-x86-64.so.2  -o mainprog_2  -
Map=mainprogmap mainprog.o prog1.o prog2.o -lc /usr/lib64/crt1.o  /usr/lib64/crti.o  /usr/lib/gcc/x86_64-
linux-gnu/10.3.1/crtbegin.o /usr/lib/gcc/x86_64-linux-gnu/10.3.1/crtend.o  /usr/lib64/crtn.o
[pmi-b2412@pmi-srv-openscaler workdir]$
```

Сборка прошла успешно:

```
[pmi-b2412@pmi-srv-openscaler workdir]$ ls
mainprog    mainprog_2  mainprogmap  makefile           PostfixExpression.cpp  prog1.o  prog2.o
mainprog_1  mainprog.c  mainprog.o   PostfixExpression  prog1.c                prog2.c  test_mainprog
[pmi-b2412@pmi-srv-openscaler workdir]$
```

Программа работает корректно:

```
[pmi-b2412@pmi-srv-openscaler workdir]$ ./mainprog_2
testtesttest test
'3'
banana ban
'1'
```

11. Сохраните в репозитории все файлы, сформированные при выполнении п.8-10.

```
[pmi-b2412@pmi-srv-openscaler workdir]$ git commit -m "Файлы, сформированные при выполнении пункто
в 8-10"[lab_6 4d110c0] Файлы, сформированные при выполнении пунктов 8-10
 8 files changed, 543 insertions(+), 40 deletions(-)
 rewrite mainprog.c (68%)
 create mode 100644 mainprog.o
 create mode 100755 mainprog_1
 create mode 100755 mainprog_2
 create mode 100644 mainprogmap
 create mode 100644 prog1.c
 create mode 100644 prog1.o
 create mode 100644 prog2.o
[pmi-b2412@pmi-srv-openscaler workdir]$ git log --oneline
4d110c0 (HEAD -> lab_6) Файлы, сформированные при выполнении пунктов 8-10
```

12. Из карты памяти progmap определите размеры машинного кода модулей mainprog.o, prog1.o и prog2.o, сравните их с размерами исходного и объектного кода этих модулей. Размеры файлов типа .c и .o определите с помощью команды ls. Результат представьте в виде таблицы 4, все данные должны быть подтверждены скриншотами.

Имя модуля (функции)	Исходный, байт	Объектный, байт	Машинный код, байт
prog1	480	1600	157
prog2	312	1500	105
mainprog	321	1700	97

Машинный код (hex):

```
.text          0x0000000000401070       0x61 mainprog.o
               0x0000000000401070                 main
.text          0x00000000004010d1       0x9d prog1.o
               0x00000000004010d1                 count_words
.text          0x000000000040116e       0x69 prog2.o
               0x000000000040116e                 get_word
```

Объектный:

```
[pmi-b2412@pmi-srv-openscaler workdir]$ ls -lh prog1.o prog2.o mainprog.o
-rw-r--r-- 1 pmi-b2412 pmi-b2412 1,7K мая 14 15:10 mainprog.o
-rw-r--r-- 1 pmi-b2412 pmi-b2412 1,6K мая 14 15:10 prog1.o
-rw-r--r-- 1 pmi-b2412 pmi-b2412 1,5K мая 14 14:55 prog2.o
```

Исходный:

```
[pmi-b2412@pmi-srv-openscaler workdir]$ ls -lh prog1.c prog2.c mainprog.c
-rw-r--r-- 1 pmi-b2412 pmi-b2412 321 мая 14 14:58 mainprog.c
-rw-r--r-- 1 pmi-b2412 pmi-b2412 480 мая 14 14:59 prog1.c
-rw-r--r-- 1 pmi-b2412 pmi-b2412 312 мая 14 14:46 prog2.c
```

13. Добавьте в make-файл, разработанный при выполнении п. 2, два новых правила, реализующие п. 9 и 10 задания. Проверьте корректность его работы и сохраните его в репозитории.

Проверка корректности implink:

```
[pmi-b2412@pmi-srv-openscaler workdir]$ make implink
gcc mainprog.c prog1.c prog2.o -o mainprog_1
[pmi-b2412@pmi-srv-openscaler workdir]$ ./mainprog_1
testtesttest test
'3'
```

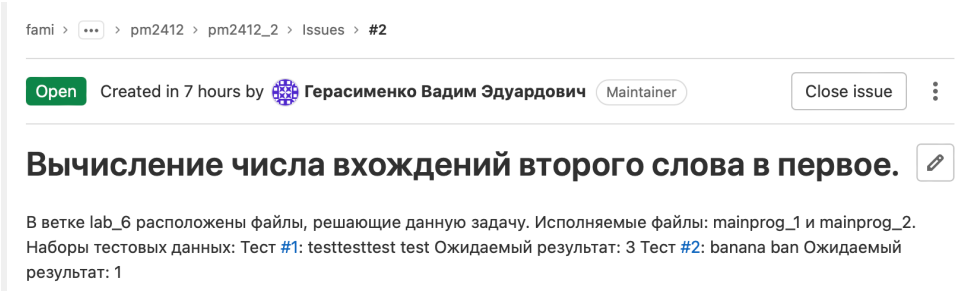
Проверка корректности explink:

```
[pmi-b2412@pmi-srv-openscaler workdir]$ make explink
ld -dynamic-linker  /lib64/ld-linux-x86-64.so.2  -o mainprog_2  -Map=mainprogmap mainprog.o prog1.o prog2.o -lc /usr/lib64/crt1.o
/usr/lib64/crti.o  /usr/lib/gcc/x86_64-linux-gnu/10.3.1/crtbegin.o /usr/lib/gcc/x86_64-linux-gnu/10.3.1/crtend.o  /usr/lib64/crtn.o

[pmi-b2412@pmi-srv-openscaler workdir]$ ./mainprog_2
testtesttest test
'3'
banana ban
'1'
```

14. Выполните экспорт всех данных из локального репозитория в облачный репозиторий НГТУ. В разделе Issues проекта GitLab добавить описание задания, например, «Сортировка слов в строке по возрастанию длины».

```
[pmi-b2412@pmi-srv-openscaler workdir]$ git push origin lab_6
Username for 'https://gitlab.cloud.nstu.ru': v.gerasimenko.2022@stud.nstu.ru
Password for 'https://v.gerasimenko.2022@stud.nstu.ru@gitlab.cloud.nstu.ru':
warning: переадресация на https://gitlab.cloud.nstu.ru/fami/pm-24/pm2412/pm2412_2.git/
Перечисление объектов: 22, готово.
Подсчет объектов: 100% (22/22), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (21/21), готово.
Запись объектов: 100% (21/21), 10.95 КиБ | 2.74 МиБ/с, готово.
Всего 21 (изменений 9), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote:
remote: To create a merge request for lab_6, visit:
remote:   http://gitlab.cloud.nstu.ru/fami/pm-24/pm2412/pm2412_2/-/merge_requests/new?merge_request%5Bsource_branch%5D=lab_6
remote:
To http://gitlab.cloud.nstu.ru/fami/pm-24/pm2412/pm2412_2.git
 * [new branch]      lab_6 -> lab_6
[pmi-b2412@pmi-srv-openscaler workdir]$
```



3. Вывод:

В ходе лабораторной работы были изучены основные этапы разработки и отладки приложений в ОС Linux, а также приобретены практические навыки по использованию инструментальных средств фонда свободного программного обеспечения при компиляции исходного кода, сборке, отладке и тестировании программ, написанных на языке С.

Приложение:

Программа 12. Вычисление числа вхождений второго слова в первое

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

#define MAX_LENGTH 256

int get_word(char*string, int num)
{
   int count = 0;
   int end = strlen(string);
   for (int i = 0; i < end - 1; i++)
   {
      if (count == num)
         return i;
      if (string[i] == ' ')
         count++;
   }
   return -1;
}
int count_words(char*string)
{
   int length = strlen(string);
   int first_word_start = get_word(string, 0);
   int second_word_start = get_word(string, 1);
   int count = 0;
   for (int i = 0; i < second_word_start; i++)
      if (strncmp(string + i, string + second_word_start, length - second_word_start) == 0)
         count++;
   return count;
}

int main(int argc, char* argv[])
{
   char string[MAX_LENGTH];
   while (fgets(string, MAX_LENGTH, stdin) != (char*)NULL)
   {
      printf("'%d'\n", count_words(string));
   }
   return 0;
}
```

Входные и выходные КОРРЕКТНЫЕ данные:

Входные данные: banana ban
Выходные данные: 1

Входные данные: testtesttest test
Выходные данные: 3