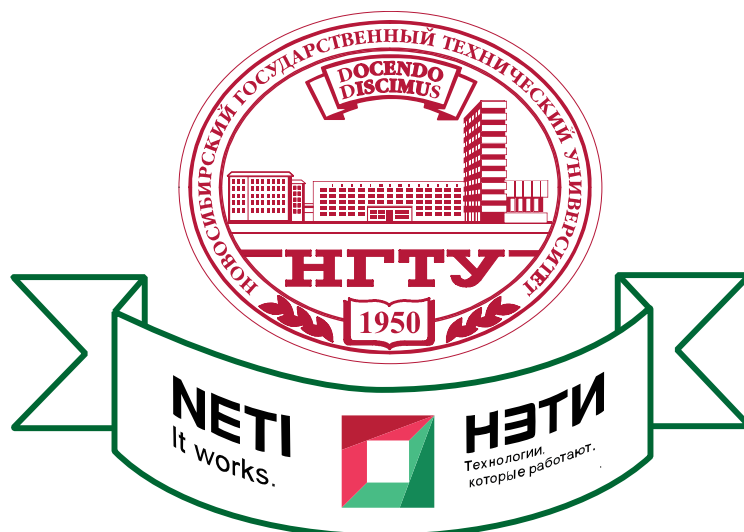


Министерство науки и высшего образования
Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

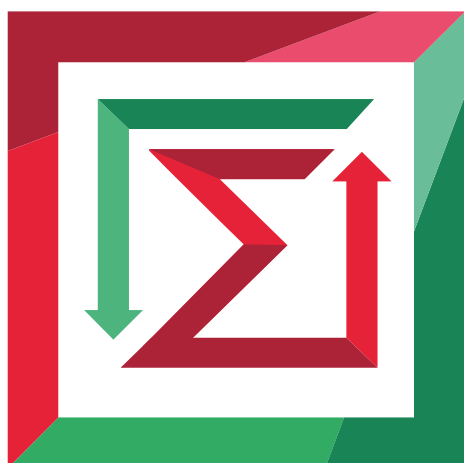


Теоретической и прикладной математики

Лабораторная работа № 6

по дисциплине «Операционные системы, среды и оболочки»

ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА РАЗРАБОТКИ ПРОГРАММ



Факультет:	ПМИ
Группа:	ПМИ-02
Вариант:	2
Студент:	Сидоров Даниил Игоревич
Преподаватель:	Кобылянский Валерий Григорьевич, Филиппова Елена Владимировна

Новосибирск

2026

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 Си.

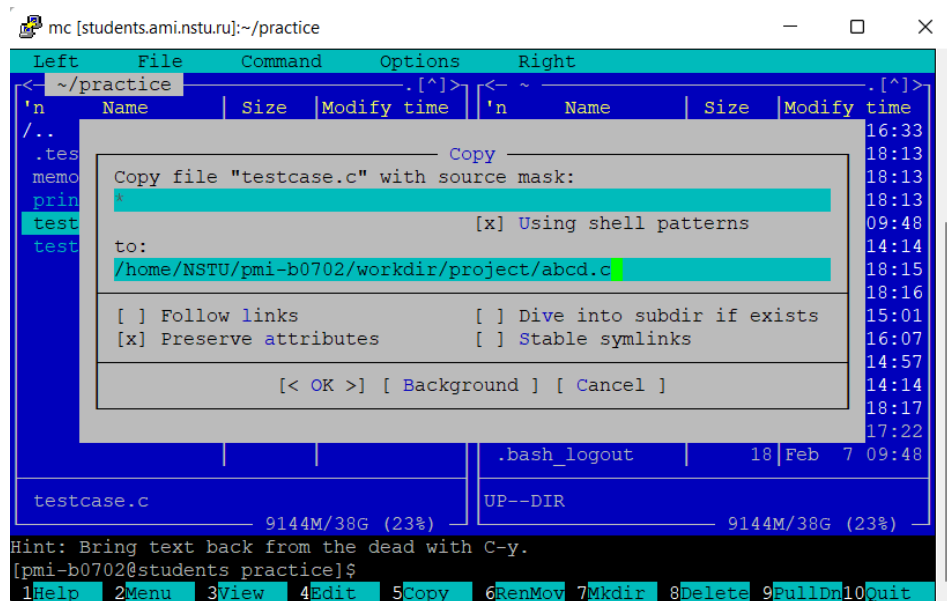
2. Ход работы

1. Запустили файловый менеджер **mc** и создали в домашнем каталоге подкаталоги **cvsroot**, **workdir** и **examples**, в каталоге **workdir** создали подкаталог **project**.

```
mc [pmi-b0702@students.ami.nstu.ru]:~  
Left File Command Options Right  
< ~ .[^]> < ~ .[^]>  
'n Name Size Modify time 'n Name Size Modify time  
/.. UP--DIR May 18 16:33 /.. UP--DIR May 18 16:33  
/.cache 4096 May 29 18:13 /.cache 4096 May 29 18:13  
/.config 4096 May 29 18:13 /.config 4096 May 29 18:13  
/.local 4096 May 29 18:13  
/.mozilla 4096 Feb 7 09:48  
/abc_02 4096 Feb 20 14:14  
/monthly 4096 Feb 20 15:01  
/practice 4096 Mar 6 16:07  
/reports 4096 Feb 20 14:57  
/trash_02 4096 Feb 20 14:14  
 5131 May 21 17:22  
 18 Feb 7 09:48  
 193 Feb 7 09:48  
 231 Feb 7 09:48  
 5810 May 21 17:01  
 18 Feb 7 09:48  
 193 Feb 7 09:48  
 231 Feb 7 09:48  
 5810 May 21 17:01  
UP--DIR 9144M/38G (23%) UP--DIR 9144M/38G (23%)  
Hint: M-! will allow you to execute programs and see the output in the viewer.  
[pmi-b0702@students ~]$  
1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn10Quit  
Create a new Directory  
Enter directory name:  
cvsroot  
[< OK >] [ Cancel ]  
'n Name Size Modify time  
/.. UP--DIR May 18 16:33  
/.cache 4096 May 29 18:13  
/.config 4096 May 29 18:13  
/.local 4096 May 29 18:13  
/.mozilla 4096 Feb 7 09:48  
/abc_02 4096 Feb 20 14:14  
/cvsroot 4096 May 29 18:15  
/examples 4096 May 29 18:16  
/monthly 4096 Feb 20 15:01  
/practice 4096 Mar 6 16:07  
/reports 4096 Feb 20 14:57  
/trash_02 4096 Feb 20 14:14  
/workdir 4096 May 29 18:17  
 5131 May 21 17:22  
 18 Feb 7 09:48  
 193 Feb 7 09:48  
 231 Feb 7 09:48  
 5810 May 21 17:01  
/workdir 9144M/38G (23%)
```


3. Скопировали файл **testcase.c** в каталог **project** под именем **abcd.c** и занесли в отчет исходный текст программы, предварительно сравнив его с текстом, приведенным в п.2.6.

Скопируем при помощи F5.



Исходный код:

```
#include <stdio.h>

#include <ctype.h>
#include <string.h>

#define WORD      0
#define IGNORE    1

/* Globals, used by both subroutines. */
char    *Words[BUFSIZ/2];          /* Worst case, single letters. */
int      WordCount;

/* Walk through the array of words, find those with the
 * matching character, printing them on stdout. Note that
 * the NULL character will match all words. */

void PrintWords(wc, match)

int wc;                          /* Number of words in Words[] */
char match;                      /* Attempt to match this character. */
{ register int    ix;            /* Index in Words[]. */
  register char   *cp;           /* Pointer for searching. */
  for (ix=0; ix < wc; ix++) {
    cp = Words[ix];
    /* Try to match the given character.
     * Scan the word, attempting to match,
     * or until the end of the word is found. */
    while ((*cp) && (*cp++ != match));
    if (*cp == match) /* Found a match? Write the word on stdout. */
      (void) printf("%s0", Words[ix]); } return; }
```

```

/* Find words in the gives buffer. The Words[] array is set
 * to point at words in the buffer, and the buffer modifeid
 * with NULL characters to delimit the words. */
int GetWords (buf)
char    buf[];                /* The output buffer. */
{
    register char    *cp;      /* Pointer for scanning. */
    int    end = strlen(buf); /* length of the buffer. */
    register int    wc = 0;    /* Number of words found. */
    int    state = IGNORE;    /* Current state. */
    /* For each character in the buffer. */
    for (cp = &buf[0]; cp < &buf[end]; cp++) {
        /* A simple state machine to process
         * the current character in the buffer. */
        switch(state) {
            case IGNORE:
                if (!isspace(*cp)) {
                    Words[wc++] = cp; /* Just started a word? Save it. */
state = WORD; /* set the state. */ } break;
            case WORD:
                if (isspace(*cp)) {
                    *cp = '\0'; /* Just completed aword? terminate it. */
                    state = IGNORE; /* set the state. */ } break; }}
        return wc; /* Return the word count. */ }

int main(argc, argv) int argc; char *argv[]; { char buf[BUFSIZ], match;
/* Check command line arguments. */
if (argc < 2) match = ' ';
/* No command line argument, match all words. */
else match = *++argv[1]; /* match the char after the first - */
/* Until no more output on stdin. */
while(gets(buf) != (char *)NULL) {
    WordCount = GetWords(buf); /* Paste the output buffer. */
    PrintWords(WordCount, match); /* Print the matching words. */ }
/* Manifests for state machine to parse output line. */

```

4. С помощью редактора **vi** создали в каталоге **project** make-файл согласно п. 2.3.

```

[pmi-b0702@students ~]$ vi ~/workdir/project/makefile

```

```

abcd: abcd.c
      gcc -o abcd abcd.c
testabcd: abcd.c
      gcc -o testabcd -g abcd.c

```

5. Выполнили компиляцию программы **abcd.c** с помощью make-файла, используя правило **abcd**. Исправили синтаксические ошибки с помощью редактора **vi**, информацию по ошибкам и их устранению занесли в отчет (номер строки, значение строки до устранения и после устранения ошибки, пояснения).

```
[pmi-b0702@students project]$ make abcd
gcc -o abcd abcd.c
abcd.c: In function 'PrintWords':
abcd.c:26:41: warning: missing terminating " character [enabled by default]
          (void) printf("%s0, Words[ix]); } return; }
                              ^
abcd.c:26:27: error: missing terminating " character
          (void) printf("%s0, Words[ix]); } return; }
                              ^
abcd.c:31:5: error: expected expression before 'int'
      int GetWords (buf)
      ^
abcd.c:61:19: error: expected declaration or statement at end of input
      PrintWords(WordCount, match); /* Print the matching words. */
      }
abcd.c:61:19: error: expected declaration or statement at end of input
make: *** [abcd] Error 1

[pmi-b0702@students project]$
```

```
abcd.c:58:14: warning: 'gets' is deprecated (declared at /usr/include/stdio.h:638) [-Wdeprecated-declarations]
      while(gets(buf) != (char *)NULL) {
      ^
```

Строка №26. Не закрыты кавычки. Заменяли “%s0 на “%s\n”

До исправления:

```
(void) printf("%s0, Words[ix]); } return; }
```

После исправления:

```
(void) printf("%s\n", Words[ix]); } return; }
```

Строка №61. Нет строки “return(0);”.

До исправления:

```
PrintWords(WordCount, match); /* Print the matching words. */
}
```

После исправления:

```
PrintWords(WordCount, match); /* Print the matching words. */
return(0); }
```

6. В каталоге **cvsroot** создали репозиторий CVS.

```
[pmi-b0702@students cvsroot]$ cvs -d ~/cvsroot init
```

< ~/cvsroot		.[^]>	
'n	Name	Size	Modify time
/..		UP--DIR	May 29 18:49
/CVSROOT		4096	May 29 18:54

7. Передали каталог **project** и файл **abcd.c** в репозиторий. При выполнении команды **commit** с помощью редактора **vi** ввели комментарий, например: "Пользователь pmi-b0702 передал файл abcd.c под управление CVS".

Свяжем рабочий каталог с хранилищем:

```
[pmi-b0702@students workdir]$ cvs -d ~/cvsroot checkout -l .  
cvs checkout: Updating .  
? project
```

Передадим проект в хранилище:

```
[pmi-b0702@students workdir]$ cvs -d ~/cvsroot add project  
Directory /home/NSTU/pmi-b0702/cvsroot/project added to the repository  
[pmi-b0702@students workdir]$
```

Передадим файл abcd.c в хранилище:

```
[pmi-b0702@students workdir]$ cvs -d ~/cvsroot add project/abcd.c  
cvs add: scheduling file `project/abcd.c' for addition  
cvs add: use 'cvs commit' to add this file permanently  
[pmi-b0702@students workdir]$
```

Внесём изменения в репозиторий:

```
Пользователь pmi-b0702 передал файл abcd.c под управление CVS.  
CVS: -----  
CVS: Enter Log. Lines beginning with `CVS:' are removed automatically  
CVS:  
CVS: Committing in project  
CVS:  
CVS: Added Files:  
CVS:   abcd.c  
CVS: -----
```

```
[pmi-b0702@students workdir]$ cvs -d ~/cvsroot commit  
cvs commit: Examining .  
cvs commit: Examining project  
RCS file: /home/NSTU/pmi-b0702/cvsroot/project/abcd.c,v  
done  
Checking in project/abcd.c;  
/home/NSTU/pmi-b0702/cvsroot/project/abcd.c,v <-- abcd.c  
initial revision: 1.1  
done  
[pmi-b0702@students workdir]$
```

8. Запустили исполняемый файл abcd и пояснили результат запуска.

```
[pmi-b0702@students project]$ ./abcd  
test test test  
set  
sgseges  
dfgdfgdfgdf  
dfgfdg dfgd  
^Z  
[6]+ Stopped ./abcd  
[pmi-b0702@students project]$
```

Запустили исполняемый файл `abcd`: ожидает ввода строки, но не работает после ввода символов. Программа работает неверно.

9. Перекомпилировали программу с помощью правила **testabcd** make-файла.

```
[pmi-b0702@students project]$ make testabcd
gcc -o testabcd -g abcd.c
[pmi-b0702@students project]$
```

10. С помощью отладчика **gdb** выполнили поиск и устранение семантических ошибок в программе **abcd**. Каждое исправление в программе должно сопровождаться записью в репозиторий новой версии с комментарием, поясняющим на русском языке суть исправлений (например, номер строки программы **abcd.c** и причина исправления). После устранения всех ошибок занесли в отчет результаты тестирования программы в двух вариантах запуска – без параметра и с параметром.

```
[pmi-b0702@students project]$ gdb testabcd
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-80.el7
Copyright (C) 2013 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".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /home/NSTU/pmi-b0702/workdir/project/testabcd...done.
(gdb)
```

```
(gdb) start
Temporary breakpoint 1 at 0x4007d6: file abcd.c, line 54.
Starting program: /home/NSTU/pmi-b0702/workdir/project/testabcd

Temporary breakpoint 1, main (argc=1, argv=0x7fffffffe2b8) at abcd.c:54
54      if (argc < 2) match = ' ';
Missing separate debuginfos, use: debuginfo-install glibc-2.17-106.el7_2.1.x86_64
(gdb)
```

```
description:
-----
revision 1.4
date: 2022/05/29 12:56:13; author: pmi-b0702; state: Exp; lines: +0 -1
Изменен цикл While ((*cp) && (*cp++!=match)); Заменено на ((*cp) &&(*cp!=match)) cp++;
-----
revision 1.3
date: 2022/05/29 12:54:26; author: pmi-b0702; state: Exp; lines: +1 -0
Изменена функция вывода: print(%s0,words[ix]); Заменена на print(%sn,words[ix]);
-----
revision 1.2
date: 2022/05/29 12:47:07; author: pmi-b0702; state: Exp; lines: +6 -5
Ошибка присваивания: match=' ' Заменено на match='\0'
-----
revision 1.1
date: 2022/05/29 12:05:25; author: pmi-b0702; state: Exp;
Пользователь pmi-b0702 передал файл abcd.c под управление CVS.
```



```
[pmi-b0702@students project]$ ./abcd
This is test abcd program
This
is
test
abcd
program
```

С параметром -t:

```
[pmi-b0702@students project]$ ./abcd -t
this is test abcd program
this
test
```

Программа работает верно.

11. После получения корректных результатов выполнения программы **abcd** с помощью редактора **vi** в начало отлаженной программы ввели комментарий: "Программа abcd отлажена с помощью отладчика gdb дд.мм.гг. бригадой группы ПМИ-XX в составе: ФИО1, ФИО2..." и сохранили в репозиторий финальную версию программы.

```
/*Программа abcd отлажена с помощью отладчика gdb 29.05.2022 ПМИ-02 бригада №2:Сидоров Д.И. */
#include <stdio.h>
#include <ctype.h>
#include <string.h>
```

```
[pmi-b0702@students project]$ cvs commit -m "финал"
cvs commit: Examining .
Checking in abcd.c;
/home/NSTU/pmi-b0702/cvsroot/project/abcd.c,v <-- abcd.c
new revision: 1.13; previous revision: 1.12
done
[pmi-b0702@students project]$
```

12. Вывели список изменений файла **abcd.c**, выполненных в ходе отладки программы, занесли список в отчет.

```
revision 1.4
date: 2022/05/29 12:56:13; author: pmi-b0702; state: Exp; lines: +0 -1
Изменен цикл While ((*cp) && (*cp++!=match)); Заменено на ((*cp) &&(*cp!=match)) cp++;
-----
revision 1.3
date: 2022/05/29 12:54:26; author: pmi-b0702; state: Exp; lines: +1 -0
Изменена функция вывода: print(%s0,words[ix]); Заменена на print(%sn,words[ix]);
-----
revision 1.2
date: 2022/05/29 12:47:07; author: pmi-b0702; state: Exp; lines: +6 -5
Ошибка присваивания: match=' ' Заменено на match='\0'
-----
revision 1.1
date: 2022/05/29 12:05:25; author: pmi-b0702; state: Exp;
Пользователь pmi-b0702 передал файл abcd.c под управление CVS.
=====
[pmi-b0702@students project]$
```

```
-----
revision 1.13
date: 2022/05/29 13:52:34; author: pmi-b0702; state: Exp; lines: +64 -62
финал
```

13.Определили размер исполняемого модуля отлаженной программы.

Удалили всю отладочную информацию и снова определили размер исполняемого модуля, сравнили с предыдущим результатом, результат сравнения занесли в отчет.

Размер отлаженной программы

```
[pmi-b0702@students project]$ ls -sh
total 36K
 12K abcd  4.0K abcd.c  4.0K CVS  4.0K makefile  12K testabcd
[pmi-b0702@students project]$ █
```

12Кб

Удаление отладочной информации

```
[pmi-b0702@students project]$ strip -s testabcd
[pmi-b0702@students project]$ █
```

Размер модуля после удаления отладочной информации

```
[pmi-b0702@students project]$ ls -sh
total 32K
 12K abcd  4.0K abcd.c  4.0K CVS  4.0K makefile  8.0K testabcd
[pmi-b0702@students project]$ █
```

8кб

```
[pmi-b0702@students project]$ ls -l
total 32
-rwxr-xr-x. 1 pmi-b0702 пользователи домена 8839 May 29 20:45 abcd
-rw-r--r--. 1 pmi-b0702 пользователи домена 3301 May 29 20:52 abcd.c
drwxr-xr-x. 2 pmi-b0702 пользователи домена 4096 May 29 20:52 CVS
-rw-r--r--. 1 pmi-b0702 пользователи домена  78 May 29 18:30 makefile
-rwxr-xr-x. 1 pmi-b0702 пользователи домена 6256 May 29 20:55 testabcd
[pmi-b0702@students project]$ █
```

Размер исполняемого модуля отлаженной программы 6256 байт

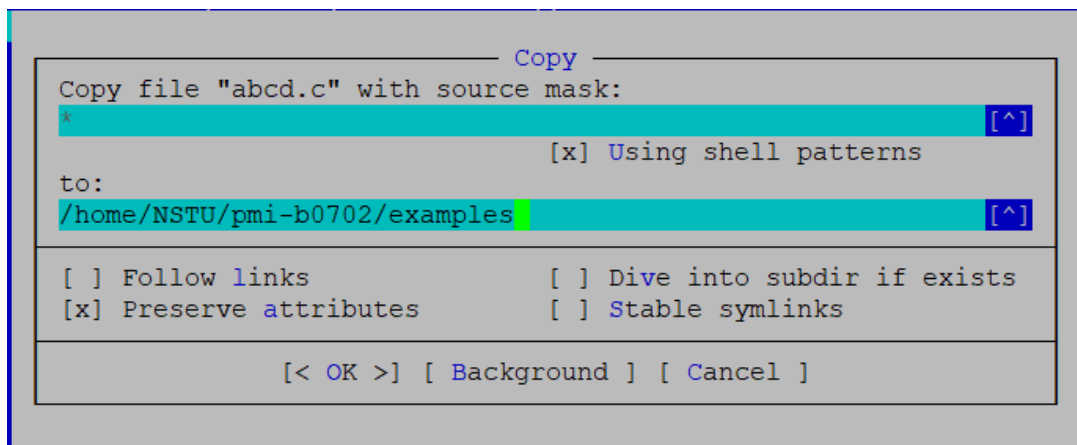
Размер Makefile 78 байт

14.Извлекли из репозитория полностью отлаженную программу **abcd** и скопировали её в каталог **example**, заменив в нем предыдущую версию программы. Все дальнейшие действия будут выполняться в этом каталоге.

Извлекли программу из репозитория

```
[pmi-b0702@students project]$ cvs checkout project
cvs checkout: Updating project
U project/abcd.c
[pmi-b0702@students project]$ █
```

Копируем при помощи F5 в каталог examples



15.Выполнили разбиение полностью отлаженной программы **abcd** на функции в соответствии с номером бригады из таблицы 3. Занесли в отчет измененный текст программы.

2	внутренний, исходный	abcd2.c	внешний, ис- ходный	printwords.h	внутренний, исходный	abcd2.c
---	-------------------------	---------	------------------------	--------------	-------------------------	---------

Abcd2.c:

/*Программа abcd отлажена с помощью отладчика gdb 29.05.2022 ПМИ-02 бригада №2:Сидоров Д.И. */

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#include "printwords.h"
#define WORD 0
#define IGNORE 1
/* Globals, used by both subroutines. */
char *Words[BUFSIZ/2]; /* Worst case, single letters. */
int WordCount;
int GetWords (buf)
char buf[]; /* The output buffer. */
{
    register char *cp; /* Pointer for scanning. */
    int end = strlen(buf); /* length of the buffer. */
    register int wc = 0; /* Number of words found. */
    int state = IGNORE; /* Current state. */
    /* For each character in the buffer. */
    for (cp = &buf[0]; cp < &buf[end]; cp++) {
        /* A simple state machine to process
         * the current character in the buffer. */
        switch(state) {
            case IGNORE:
                if (!isspace(*cp)) {
                    Words[wc++] = cp; /* Just started a word? Save it. */
                    state = WORD; /* set the state. */ } break;
            case WORD:
                if (isspace(*cp)) {
                    *cp = '\0'; /* Just completed a word? terminate it. */
```

```

        state = IGNORE;    /* set the state. */ } break; }}

    return wc; /* Return the word count. */ }

int main(argc, argv) int argc; char *argv[]; { char buf[BUFSIZ], match;
/* Check command line arguments. */
if (argc < 2) match = '\0';
/* No command line argument, match all words. */
else match = *++argv[1]; /* match the char after the first - */
/* Until no more output on stdin. */
while(fgets(buf,sizeof(buf),stdin) != (char *)NULL) {
    WordCount = GetWords(buf); /* Paste the output buffer. */
    PrintWords(WordCount, match); /* Print the matching words. */}

return(0);}

/* Manifests for state machine to parse output line. */

```

Printwords.h:

```

#include <stdio.h>

#include <ctype.h>
#include <string.h>
#define WORD    0
#define IGNORE  1
/* Globals, used by both subroutines. */
char    *Words[BUFSIZ/2];          /* Worst case, single letters. */
int      WordCount;

void PrintWords(wc, match)

int wc;                          /* Number of words in Words[] */
char match;                      /* Attempt to match this character. */
{ register int    ix;            /* Index in Words[]. */
  register char   *cp;           /* Pointer for searching. */
  for (ix=0; ix < wc; ix++) {
      cp = Words[ix];
      /* Try to match the given character.
       * Scan the word, attempting to match,
       * or until the end of the word is found. */
      while ((*cp) && (*cp != match)) cp++;
      if (*cp == match) /* Found a match? Write the word on stdout. */
          (void) printf("%s\n", Words[ix]); } return; }

```

- 16.Выполнили сборку программы в соответствии вариантом задания, используя неявный вызов компоновщика и задав имя исполняемого файла **abcd2_1**; проверили корректность работы программы и занесли в отчет результаты ее тестирования.

```

[pmi-b0702@students examples]$ gcc abcd2.c printwords.h -o abcd2_1
[pmi-b0702@students examples]$

```

```
[pmi-b0702@students examples]$ ./abcd2_1
this is test abcd2_1 program
this
is
test
abcd2_1
program
```

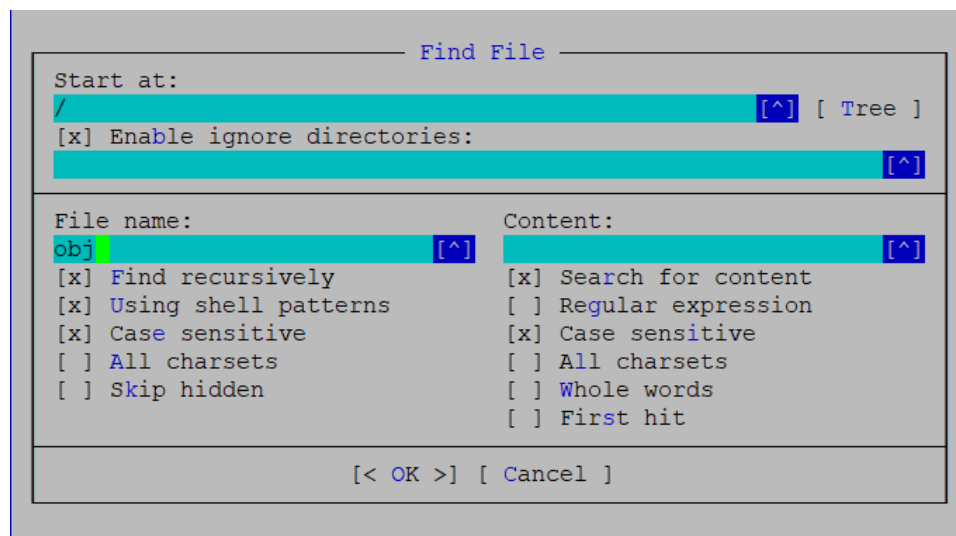
Тест с параметром -t:

```
[pmi-b0702@students examples]$ ./abcd2_1 -t
this is test abcd2_1 program
this
test
```

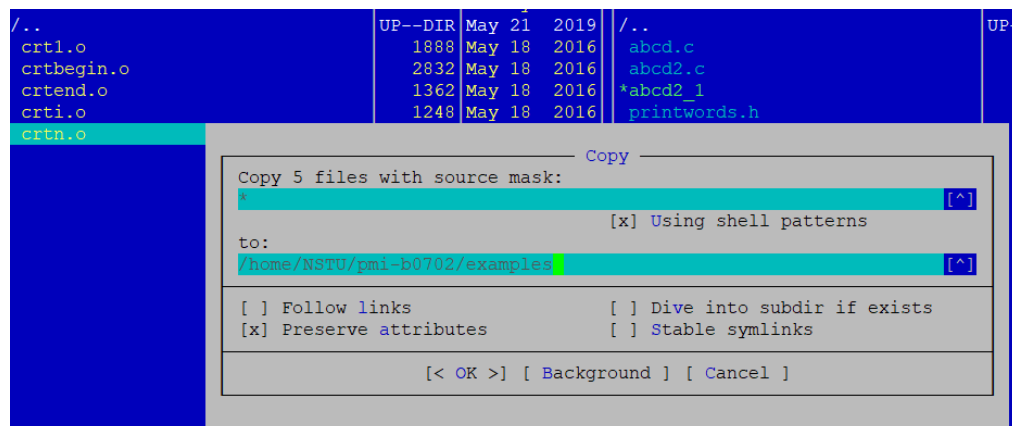
Программа работает верно

- 17.Выполнили поиск во внешней памяти сервера каталога **obj**, скопировали из него все файлы в каталог **example**, пояснили в отчете назначение скопированных файлов.

F9->Command->Find file



Выделяем все файлы и копируем при помощи F5.



~/examples		.[^]>	
'n	Name	Size	Modify time
	/..	UP--DIR	May 29 21:25
	abcd.c	3301	May 29 21:19
	abcd2.c	2230	May 29 21:25
	*abcd2_1	8840	May 29 21:25
	crt1.o	1888	May 18 2016
	crtbegin.o	2832	May 18 2016
	crtend.o	1362	May 18 2016
	crti.o	1248	May 18 2016
	crtn.o	961	May 18 2016
	printwords.h	1033	May 29 21:24

crt1.o – содержит код, инициализирующий среду исполнения языка C и вызывающий определенную пользователем функцию `main`;
crti.o – содержит пролог функции `_init`, помещаемый в начало секции `.init`;
crtn.o – содержит эпилог функции `_init`, помещаемый в конец секции `.init`;
crtbegin.o, crtend.o – обрабатывают глобальные конструкторы и деструкторы языка C++;

18.Выполнили сборку программы в соответствии вариантом задания, используя явный вызов компоновщика. Результатом сборки должны быть исполняемый файл **abcd2_2** и карта памяти **abcd2_map**; проверили корректность работы программы и занесли в отчет результаты ее тестирования.

```
[pmi-b0702@students examples]$ gcc -c abcd2.c printwords.h
[pmi-b0702@students examples]$
[pmi-b0702@students examples]$ ld -dynamic-linker /lib64/ld-linux-x86-64.so.2 -o abcd2_2 -Map=abcd2_map crt1.o crt1.o crtbegin.o abcd2.o -lc crtend.o crtn.o
[pmi-b0702@students examples]$
```

```
[pmi-b0702@students examples]$ ./abcd2_2
this is test abcd2_2 program
this
is
test
abcd2_2
program
```

Тест с параметром `-t`:

```
[pmi-b0702@students examples]$ ./abcd2_2 -t
this is test abcd2_2 program
this
test
```

Все работает верно.

19. Из карты памяти **abcd2_map** определили размеры машинного кода модулей **abcd2.o**, **printwords.o** и **getwords.o**, сравнили их с размерами исходного и объектного кода этих модулей (файлы типа **.c** и **.o**). Результат представили в виде таблицы, все данные должны быть подтверждены скриншотами.

Открыли файл **abcd2_map** с помощью F3

Имя модуля (функции)	Исходный, байт	Объектный, байт	Машинный код, байт
printwords	4.0K printwords.h	2.0M printwords.h.gch	0x000000000400630 PrintWords
getwords			0x0000000004006a2 GetWords
abcd2	4.0K abcd2.c	4.0K abcd2.o	0x000000000400620 0x1e1 abcd2.o 481 байт

20. Добавили в make-файл, разработанный при выполнении п.5, два новых правила, реализующие п. 16 и 18 задания. Проверили корректность его работы.

```
# Makefile for abcd.c
# Compile abcd.c normally
abcd: abcd.c
    gcc -o abcd abcd.c
# Compile abcd.c with debugging
testabcd: abcd.c
    gcc -o testabcd -g abcd.c
implicit: abcd2.c
    gcc printwords.h abcd2.c -o abcd2_1
explicit: abcd2.c
    gcc -c abcd2.c printwords.h
    ld -dynamic-linker /lib64/ld-linux-x84-64.so.2 -o abcd2_2 -Map=abcd2_map crt1.o crti.o crtbegin.o abcd2.o -lc crtend.o crtn.o
# End Makefile
```

Добавили правило **implicit** для сборки программы, используя неявный вызов компоновщика и правило **explicit** для сборки программы, используя явный вызов компоновщика.

```
[pmi-b0702@students examples]$ make implicit
gcc printwords.h abcd2.c -o abcd2_1
```

```
*abcd2_1 8840 May 29 22:07
[pmi-b0702@students examples]$ make explicit
gcc -c abcd2.c printwords.h
ld -dynamic-linker /lib64/ld-linux-x84-64.so.2 -o abcd2_2 -Map=abcd2_map crt1.o crti.o crtbegin.o abcd2.o -lc crtend.o crtn.o
[pmi-b0702@students examples]$
*abcd2_2 8632 May 29 22:11
```

Работает корректно.

3. Вывод

Контрольные вопросы проработаны.