# Министерство образования и науки Российской Федерации

## Федеральное государственное бюджетное образовательное учреждение высшего образования

## NSTU_Logo_blue«Новосибирский государственный технический университет»

## Кафедра прикладной математики

## Лабораторная работа № 7 по дисциплине «Операционные системы»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| сигма градиент синий1 | Факультет: | ПМИ | |  |  |
| Группа: | ПМИ-62 | |  |  |
| Студенты: | Мамонова Е.В., Ершов П.К. | |  |  |
| Преподаватель: | Кобылянский В.Г. | |  |  |
|  |  |

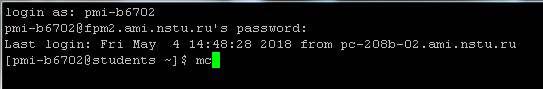
Новосибирск

2018

1. Цель работы.

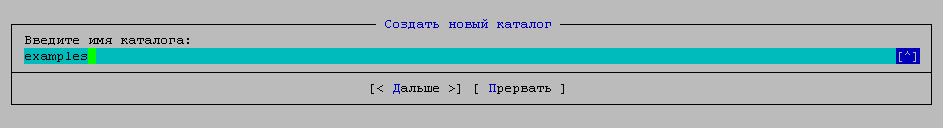
Целью работы является изучение основных этапов разработки и отладки приложений в ОС Linux, а также приобретение практических навыков по использованию инструментальных средств фонда свободного программного обеспечения при компиляции исходного кода, сборке, отладке и тестировании программ, написанных на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

1. Ход работы
2. Запустите файловый менеджер **mc** .



1. В домашнем каталоге создайте подкаталог **examples**.

F7

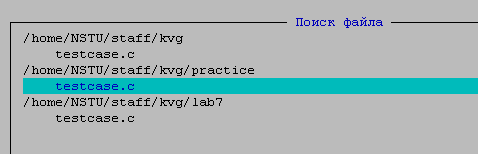




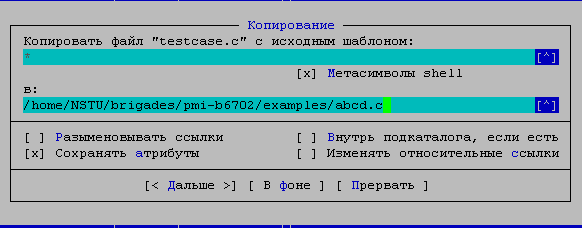
1. Выполните поиск во внешней памяти файла **testcase.c**

Команда-> Поиск файла

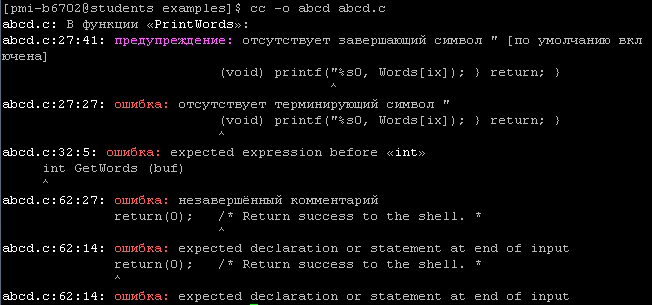




1. Скопируйте файл **testcase.c** в ваш подкаталог **examples** под именем **abcd.c**.



1. Выполните компиляцию программы **abcd.c.** Исправьте синтаксические ошибки с помощью редактора **vi,** всю информацию по ошибкам и их устранению занесите в отчет.



[pmi-b6702@students examples]$ vi abcd.c

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

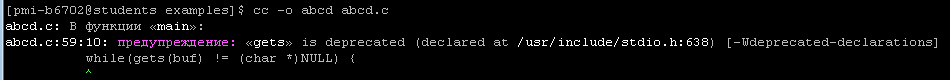
Исправлено на

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

/\* Return success to the shell. \*

Исправлено на

/\* Return success to the shell. \*/}



1. С помощью редактора **vi** создайте в поддиректории **examples** make-файл согласно п. 2.3. Далее различные варианты построения исполняемого модуля должны быть получены с помощью программы **make**.

[pmi-b6702@students examples]$ vi makefile

Вводим в новом файле следующий текст:

# Makefile for abcd.c

# Compile abcd.c normaly

abcd: abcd.c

gcc -o abcd abcd.c

# Compile abcd.c with debugging

testabcd: abcd.c

gcc -o testabcd -g abcd.c

# End Makefile

Выходим из редактора vi

: wq

[pmi-b6702@students examples]$ make abcd

gcc -o abcd abcd.c

abcd.c: In function ‘main’:

abcd.c:59:10: warning: ‘gets’ is deprecated (declared at /usr/include/stdio.h:638) [-Wdeprecated-declarations]

while(gets(buf) != (char \*)NULL) {

^

[pmi-b6702@students examples]$ make testabcd

gcc -o testabcd -g abcd.c

abcd.c: In function ‘main’:

abcd.c:59:10: warning: ‘gets’ is deprecated (declared at /usr/include/stdio.h:638) [-Wdeprecated-declarations]

while(gets(buf) != (char \*)NULL) {

^

7. Выполните отладку программы **abcd.c** с помощью отладчика **gdb**. Занесите в отчет результаты выполнения программы до и после устранения каждой ошибки, а также суть исправлений (например, номер строки программы **abcd.c** и причина исправления)..

[pmi-b6702@students examples]$ 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/brigades/pmi-b6702/examples/testabcd...done.

(gdb) break 55

Breakpoint 1 at 0x400796: file abcd.c, line 55.

(gdb) run

Starting program: /home/NSTU/brigades/pmi-b6702/examples/testabcd

Breakpoint 1, main (argc=1, argv=0x7fffffffe2b8) at abcd.c:55

55 if (argc < 2) match = ' ';

Missing separate debuginfos, use: debuginfo-install glibc-2.17-106.el7\_2.1.x86\_64

(gdb) step

59 while(gets(buf) != (char \*)NULL) {

(gdb) step

this is a test program

60 WordCount = GetWords(buf); /\* Paste the input buffer. \*/

(gdb) step

GetWords (buf=0x7fffffffc1c0 "this is a test program") at abcd.c:35

35 int end = strlen(buf); /\* length of the buffer. \*/

(gdb)

36 register int wc = 0; /\* Number of words found. \*/

(gdb) display end

1: end = 22

(gdb) step

37 int state = IGNORE; /\* Current state. \*/

1: end = 22

(gdb)

39 for (cp = &buf[0]; cp < &buf[end]; cp++) {

1: end = 22

(gdb)

42 switch(state) {

1: end = 22

(gdb) display state

2: state = 1

(gdb) step

44 if (!isspace(\*cp)) {

2: state = 1

1: end = 22

(gdb)

45 Words[wc++] = cp; /\* Just started a word? Save it. \* /

2: state = 1

1: end = 22

(gdb)

46 state = WORD; /\* Reset the state. \*/ } break;

2: state = 1

1: end = 22

(gdb)

39 for (cp = &buf[0]; cp < &buf[end]; cp++) {

2: state = 0

1: end = 22

(gdb)

42 switch(state) {

2: state = 0

1: end = 22

(gdb)

48 if (isspace(\*cp)) {

2: state = 0

1: end = 22

(gdb)

39 for (cp = &buf[0]; cp < &buf[end]; cp++) {

2: state = 0

1: end = 22

(gdb)

42 switch(state) {

2: state = 0

1: end = 22

(gdb) break 51

Breakpoint 2 at 0x400772: file abcd.c, line 51.

(gdb) continue

Continuing.

Breakpoint 2, GetWords (buf=0x7fffffffc1c0 "this") at abcd.c:51

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

2: state = 0

1: end = 22

(gdb) print wc

$1 = 5

(gdb) step

main (argc=1, argv=0x7fffffffe2b8) at abcd.c:61

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

(gdb)

PrintWords (wc=5, match=32 ' ') at abcd.c:20

20 for (ix=0; ix < wc; ix++) {

(gdb) print wc

$2 = 5

(gdb) print match

$3 = 32 ' '

(gdb) step

21 cp = Words[ix];

(gdb)

25 while ((\*cp) && (\*cp++ != match));

(gdb)

26 if (\*cp == match) /\* Found a match? Write the word on stdout. \*/

(gdb)

20 for (ix=0; ix < wc; ix++) {

(gdb)

21 cp = Words[ix];

(gdb)

25 while ((\*cp) && (\*cp++ != match));

(gdb)

26 if (\*cp == match) /\* Found a match? Write the word on stdout. \*/

(gdb)

20 for (ix=0; ix < wc; ix++) {

(gdb)

21 cp = Words[ix];

(gdb) print cp

$6 = 0x7fffffffc1c7 ""

(gdb) print \*cp

$7 = 0 '\000'

(gdb) step

25 while ((\*cp) && (\*cp++ != match));

(gdb) quit

A debugging session is active.

Inferior 1 [process 41812] will be killed.

Quit anyway? (y or n) y

**Исправлено:**

Строка 55

if (argc < 2) match = ' ';

на

if (argc < 2) match = '\0';

Присваиваем не пробел, а нулевой символ. Т.е символ конца строки.

Строка 27

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

на

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

Слова должны выводится в столбик

Строка 25

while ((\*cp) && (\*cp++ != match));

на

while ((\*cp) && (\*cp != match)){cp++;}

Указатель поиска должен находится за циклом

**Проверка 1:**

[pmi-b6702@students examples]$ 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/brigades/pmi-b6702/examples/testabcd...done.

(gdb) run

Starting program: /home/NSTU/brigades/pmi-b6702/examples/testabcd

this is a test the abcd program

this

is

a

test

the

abcd

program

^Z

Program received signal SIGTSTP, Stopped (user).

0x00007ffff7b019d0 in \_\_read\_nocancel () from /lib64/libc.so.6

Missing separate debuginfos, use: debuginfo-install glibc-2.17-106.el7\_2.1.x86\_64

(gdb) quit

A debugging session is active.

Inferior 1 [process 42744] will be killed.

Quit anyway? (y or n) y

**Проверка 2:**

[pmi-b6702@students examples]$ 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/brigades/pmi-b6702/examples/testabcd...done.

(gdb) run -t

Starting program: /home/NSTU/brigades/pmi-b6702/examples/testabcd -t

this is a test the abcd program

this

test

the

^Z

Program received signal SIGTSTP, Stopped (user).

0x00007ffff7b019d0 in \_\_read\_nocancel () from /lib64/libc.so.6

Missing separate debuginfos, use: debuginfo-install glibc-2.17-106.el7\_2.1.x86\_64

(gdb) quit

A debugging session is active.

Inferior 1 [process 42793] will be killed.

Quit anyway? (y or n) y

8. После получения корректных результатов выполнения программы abcd с помощью редактора vi в начало отлаженной программы введите комментарий: "Программа abcd отлажена с помощью отладчика gdb дд.мм.гг. бригадой группы ПМ-ХХ в составе: ФИО1, ФИО2..."

[pmi-b6702@students examples]$ vi abcd.c

Исправленная программа abcd

/\* Программа abcd отлажена с помощью отладчика gdb

11.05.2018.

Бригадой группы ПМИ-62 в составе:

Мамонова Елизавета Владимировна,

Ершов Пётр Константинович. \*/

#include <stdio.h>

#include <ctype.h>

#include <string.h>

/\* Manifests for state machine to parse input line. \*/

#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 works, find those with the

\* matching charakter, printing them on stdout. Note that

\* the NULL charakter will match all words. \*/

void PrintWords(wc, match)

int wc; /\* Number of words in Words[] \*/

char match; /\* Attempt to match this charakter. \*/

{ 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; }

/\* 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 input 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; /\* Reset the state. \*/ } break;

case WORD:

if (isspace(\*cp)) {

\*cp = '\0'; /\* Just completed aword? terminate it. \*/

state = IGNORE; /\* Reset 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 input on stdin. \*/

while(gets(buf) != (char \*)NULL) {

WordCount = GetWords(buf); /\* Paste the input buffer. \*/

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

return(0); /\* Return success to the shell. \*/}