Отчёт по лабораторной работе № 10 по курсу «Архитектура компьютера и информационных систем»

Студент группы М8О-112Б-22

Епифанов Евгений Валерьевич, № по списку: 9

# E-mail: [epi\_abitur](mailto:epi_abitur)@mail.ru

# Телефон: 8(916)783-95-69

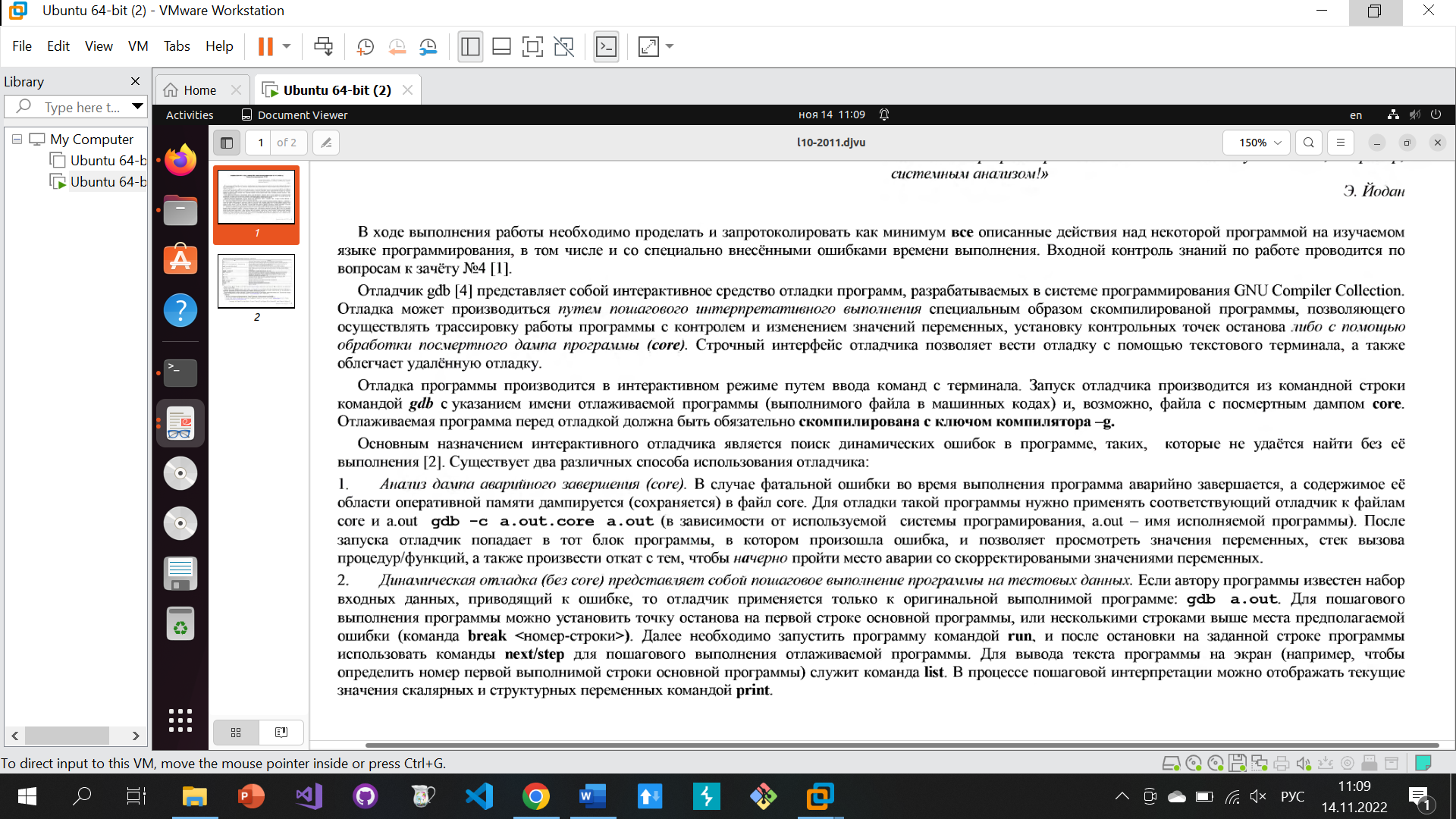
# Работа выполнена: 13 ноября 2022 г.

Преподаватель: Никулин Сергей Петрович, каф.806

**1. Тема:** Отладчик системы программирования OC Unix.

**2. Цель работы:** Научиться отлаживать простейшие программы, написанные на языке Си

**3. Задание:** составить и отладить собственную простейшую программу на Си.



**4. Оборудование (ПЭВМ студента):**

Ноутбук с процессором Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz, 1992 МГц, ядер: 4, логических процессоров: 8, ОП 8 ГБ. SSD 256 ГБ.

**5. Программное обеспечение ЭВМ студента:**

* Операционная система семейства UNIX, наименование: Linux Ubuntu (22.04) 5.15.0-48-generic #54-Ubuntu SMP Fri Aug 26 13:26:29 UTC 2022 x86\_64 x86\_64 x86\_64 GNU/Linux.
* Интерпретатор команд: GNU bash, version 5.1.16(1)-release (x86\_64-pc-linux-gnu).
* Компилятор gcc Linux.
* Редактор текстов: GNU Emacs 28.2
* Отладчик gdb
* Утилиты и прочие программы.

**6. Идея, метод, алгоритм решения задачи:**

1. Взаимодействие с программой с помощью gdb.
   1. help - Подсказка по разделу отладчика.
   2. list - Распечатка текста функции/процедуры/файла или всей программы, начиная с указанной строки.
   3. break - Задание точки остановки.
   4. run - Запуск программы на выполнение.
   5. set args - Предварительная установка параметров командной строки.
   6. print - Печать значения выражения.
   7. next - Выполнение очередной строки программы при пошаговой трассировке.
   8. step - Выполнение очередной строки программы.
   9. set var - Присваивание значения переменной.
   10. ptype - Выводит тип переменной.
   11. backtrace (bt) - Распечатка содержимого стека вызовов.
   12. continue - Продолжение выполнения программы после остановки.
   13. quit - Выход из отладчика.
   14. show args - Просмотр установленных параметров.
2. Программа:

#include <stdio.h>

#include <math.h>

int mod(int n, int m) // Функция mod

{

return(((n%m)+m)%m);

}

int main()

{

long long R = 10, a1 = -10, b1 = -10, a2 = -20, b2 = -20, i = 13, j = -9, l = -4, di, dj, dl;

int k;

for (k = 0; k <= 50; ++k)

{

di = i;

dj = j;

dl = l;

if ( ((((a1 - i)\*(a1 - i)) + ((b1 - j)\*(b1 - j))) < R\*R) && ((((a2 - i)\*(a2 - i)) + ((b2 - j)\*(b2 - j))) < R\*R))

{

printf("%s%s%lli%s%lli%s%lli%s%i\n", "Попадание!", " x=", i, " y=", j, " l=",l , " k=", k);

return 1;

}

i = (((mod((i + j), 30)) / (mod((fabs(l)), 5) + 1)) + ((mod((i + l), 30)) / (mod((fabs(j)), 5) + 1)));

j = ((mod((fmax(k\*i, (k + 1)\*j)), 25)) - ((fabs(j - l)) / 10));

l = (((fabs(j - l)) / 10) + (fmin((mod((i + l), 20)), (mod(j\*k, 20)))) - 10);

}

printf("%s%s%lli%s%lli%s%lli%s%i\n", "Промах!", " x=", di, " y=", dj, " l=", dl, " k=", k - 1);

return 0;

}

Данная программа проверяет, попадает ли точка с динамическими координатами в область пересечения 2 окружностей с заданными центрами и радиусами на одном из 50 шагов.

1. GDB:

Переносимый отладчик проекта GNU, который работает на многих UNIX-подобных системах и умеет производить отладку многих языков программирования, включая Си, C++, Free Pascal, FreeBASIC, Ada, Фортран и Rust. GDB - свободное программное обеспечение, распространяемое по лицензии GPL

**7. Сценарий выполнения работы:**

1. Придумать задачу для написания программы или использовать программу из другой лабораторной работы.
2. Написать программу в текстовом редакторе Emacs.
3. Отладить программу с помощью gdb.

**8. Распечатка протокола:**

evgeniy2@evgeniy2:~$ emacs laba9.c

Изображение выглядит как текст, снимок экрана, компьютер

Автоматически созданное описание

evgeniy2@evgeniy2:~$ cc -g laba9.c -lm

evgeniy2@evgeniy2:~$ gdb a.out

GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90

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-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 a.out...

(gdb) help

List of classes of commands:

aliases -- User-defined aliases of other commands.

breakpoints -- Making program stop at certain points.

data -- Examining data.

files -- Specifying and examining files.

internals -- Maintenance commands.

obscure -- Obscure features.

running -- Running the program.

stack -- Examining the stack.

status -- Status inquiries.

support -- Support facilities.

text-user-interface -- TUI is the GDB text based interface.

tracepoints -- Tracing of program execution without stopping the program.

user-defined -- User-defined commands.

Type "help" followed by a class name for a list of commands in that class.

Type "help all" for the list of all commands.

Type "help" followed by command name for full documentation.

Type "apropos word" to search for commands related to "word".

Type "apropos -v word" for full documentation of commands related to "word".

Command name abbreviations are allowed if unambiguous.

(gdb) help next

next, n

Step program, proceeding through subroutine calls.

Usage: next [N]

Unlike "step", if the current source line calls a subroutine,

this command does not enter the subroutine, but instead steps over

the call, in effect treating it as a single source line.

(gdb) list 1

1 #include <stdio.h>

2 #include <math.h>

3

4 int mod(int n, int m) // Функция mod

5 {

6 return(((n%m)+m)%m);

7 }

8

9 int main()

10 {

(gdb)

11 long long R = 10, a1 = -10, b1 = -10, a2 = -20, b2 = -20, i = 13, j = -9, l = -4, di, dj, dl;

12 int k;

13 for (k = 0; k <= 50; ++k)

14 {

15 di = i;

16 dj = j;

17 dl = l;

18 if ( ((((a1 - i)\*(a1 - i)) + ((b1 - j)\*(b1 - j))) < R\*R) && ((((a2 - i)\*(a2 - i)) + ((b2 - j)\*(b2 - j))) < R\*R))

19 {

20 printf("%s%s%lli%s%lli%s%lli%s%i\n", "Попадание!", " x=", i, " y=", j, " l=",l , " k=", k);

(gdb)

21 return 1;

22 }

23

24 i = (((mod((i + j), 30)) / (mod((fabs(l)), 5) + 1)) + ((mod((i + l), 30)) / (mod((fabs(j)), 5) + 1)));

25 j = ((mod((fmax(k\*i, (k + 1)\*j)), 25)) - ((fabs(j - l)) / 10));

26 l = (((fabs(j - l)) / 10) + (fmin((mod((i + l), 20)), (mod(j\*k, 20)))) - 10);

27 }

28 printf("%s%s%lli%s%lli%s%lli%s%i\n", "Промах!", " x=", di, " y=", dj, " l=", dl, " k=", k - 1);

29 return 0;

30 }

(gdb)

31

(gdb)

Line number 32 out of range; laba9.c has 31 lines.

(gdb) list 15

10 {

11 long long R = 10, a1 = -10, b1 = -10, a2 = -20, b2 = -20, i = 13, j = -9, l = -4, di, dj, dl;

12 int k;

13 for (k = 0; k <= 50; ++k)

14 {

15 di = i;

16 dj = j;

17 dl = l;

18 if ( ((((a1 - i)\*(a1 - i)) + ((b1 - j)\*(b1 - j))) < R\*R) && ((((a2 - i)\*(a2 - i)) + ((b2 - j)\*(b2 - j))) < R\*R))

19 {

(gdb) break 10

Breakpoint 1 at 0x11ba: file laba9.c, line 11.

(gdb) break 16

Breakpoint 2 at 0x120e: file laba9.c, line 16.

(gdb) set args 1 1

(gdb) show args

Argument list to give program being debugged when it is started is "1 1".

(gdb) run

Starting program: /home/evgeniy2/a.out 1 1

[Thread debugging using libthread\_db enabled]

Using host libthread\_db library "/lib/x86\_64-linux-gnu/libthread\_db.so.1".

Breakpoint 1, main () at laba9.c:11

11 long long R = 10, a1 = -10, b1 = -10, a2 = -20, b2 = -20, i = 13, j = -9, l = -4, di, dj, dl;

(gdb) print l

$1 = 140737354016860

(gdb) continue

Continuing.

Breakpoint 2, main () at laba9.c:16

16 dj = j;

(gdb) print di

$2 = 13

(gdb) print l

$3 = -4

(gdb) print l

$4 = -4

(gdb) print R

$5 = 10

(gdb) set var R = 100

(gdb) print R

$6 = 100

(gdb) ptype R

type = long long

(gdb) bt

#0 main () at laba9.c:16

(gdb) continue

Continuing.

Попадание! x=22 y=21 l=-6 k=2

[Inferior 1 (process 3153) exited with code 01]

(gdb) quit

evgeniy2@evgeniy2:~$ cc -g laba9.c -lm

evgeniy2@evgeniy2:~$ gdb a.out

GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90

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-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 a.out...

(gdb) break 16

Breakpoint 1 at 0x120e: file laba9.c, line 16.

(gdb) run

Starting program: /home/evgeniy2/a.out

[Thread debugging using libthread\_db enabled]

Using host libthread\_db library "/lib/x86\_64-linux-gnu/libthread\_db.so.1".

Breakpoint 1, main () at laba9.c:16

16 dj = j;

(gdb) generate-core-file

warning: Memory read failed for corefile section, 4096 bytes at 0xffffffffff600000.

Saved corefile core.4253

(gdb) kill

Kill the program being debugged? (y or n) y

[Inferior 1 (process 4253) killed]

(gdb) exec-file core.4253

"/home/evgeniy2/core.4253": not in executable format: file format not recognized

(gdb) exit

evgeniy2@evgeniy2:~$ gdb -c core.4253 a.out

GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90

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-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 a.out...

[New LWP 4253]

[Thread debugging using libthread\_db enabled]

Using host libthread\_db library "/lib/x86\_64-linux-gnu/libthread\_db.so.1".

Core was generated by `/home/evgeniy2/a.out'.

Program terminated with signal SIGTRAP, Trace/breakpoint trap.

#0 main () at laba9.c:16

16 dj = j;

(gdb)

evgeniy2@evgeniy2:~$

evgeniy2@evgeniy2:~$ gdb a.out

GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90

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-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 a.out...

(gdb) break 10

Breakpoint 1 at 0x11ba: file laba9.c, line 11.

(gdb) run

Starting program: /home/evgeniy2/a.out

[Thread debugging using libthread\_db enabled]

Using host libthread\_db library "/lib/x86\_64-linux-gnu/libthread\_db.so.1".

Breakpoint 1, main () at laba9.c:11

11 long long R = 10, a1 = -10, b1 = -10, a2 = -20, b2 = -20, i = 13, j = -9, l = -4, di, dj, dl;

(gdb) next

13 for (k = 0; k <= 50; ++k)

(gdb) next

15 di = i;

(gdb) next

16 dj = j;

(gdb) next

17 dl = l;

(gdb) next

18 if ( ((((a1 - i)\*(a1 - i)) + ((b1 - j)\*(b1 - j))) < R\*R) && ((((a2 - i)\*(a2 - i)) + ((b2 - j)\*(b2 - j))) < R\*R))

(gdb) next

24 i = (((mod((i + j), 30)) / (mod((fabs(l)), 5) + 1)) + ((mod((i + l), 30)) / (mod((fabs(j)), 5) + 1)));

(gdb) step 3

mod (n=4, m=5) at laba9.c:6

6 return(((n%m)+m)%m);

(gdb) step 4

mod (n=9, m=5) at laba9.c:6

6 return(((n%m)+m)%m);

9. Дневник отладки:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб. или дом. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  |  |  |  |  |  |  |

10. Замечания по существу работы: -

11. Выводы: Я научился отлаживать простейшие программы, написанные на языке Си.

Недочёты при выполнении задания могут быть устранены следующим образом: -