Задание №2

Нгуен Ань Зунг - ИУ7И-21Б

# Программа

#include <stdio.h>

#include <math.h>

#define EPS 1e-6

double calculate\_d(void)

{

double d = 0.0;

double x = 0.0;

int n = 1;

scanf("%lf", &x);

if (x < -EPS)

{

printf("Invalid input!\n");

return -1.0;

}

while (x >= -EPS) // Повторяем чтение и вычисление для всех неотрицательных значений x (или не меньше -EPS)

{

if (x < -EPS) // Если входное значение меньше -EPS (почти равное 0)

{

break;

}

d = d + sqrt(x / n); // Вычисляем квадратный корень из x / n и добавляем его к переменной d

n += 1; // Увеличиваем переменную n на единицу

if ((scanf("%lf", &x)) != 1)

{

printf("Invalid input!\n");

return -1.0;

}

}

return d;

}

int main(void)

{

double d = calculate\_d();

if (d < 0.0)

{

return 1;

}

double gx = sin(d);

printf("%lf", gx);

return 0;

}

# Этапы получения исполняемого файла

Всего есть 4 этапа получения исполняемого файла:

* **Обработка препроцессором**
  + *Удаление комментариев (замена их на пробельные символы)*
  + *Вставка файлов (директива include)*
  + *Текстовые замены (директива define)*
  + *Условная компиляция (директива if)*

Данный этап можно выполнить командой *cpp*:

**$ cpp main.c**

**$ cpp -o main.i main.c**

**$ cpp main.c > main.**

Результат работы препроцуссора :

# 5 "main.c"

double calculate\_d(void)

{

double d = 0.0;

double x = 0.0;

int n = 1;

scanf("%lf", &x);

if (x < -1e-6)

{

printf("Invalid input!\n");

return -1.0;

}

while (x >= -1e-6)

{

if (x < -1e-6)

{

break;

}

d = d + sqrt(x / n);

n += 1;

if ((scanf("%lf", &x)) != 1)

{

printf("Invalid input!\n");

return -1.0;

}

}

return d;

}

int main(void)

{

double d = calculate\_d();

if (d < 0.0)

{

return 1;

}

double gx = sin(d);

printf("%lf", gx);

return 0;

}

**Компиляция**  
На данном этапе код, что получился после препроцессирования, «переводится» с языка Си на язык Ассемблера. Это даёт несколько преимуществ:

* + *Упрощение реализации и отладки транслятора*
  + *Повышение переносимости с одной платформы на другую*

**$ c99 -S -fverbose-asm -masm=intel main.i**

Трансляции код на язык Ассемблера

Флагом «-S» мы запускаем только процесс компиляции, «-fverbose-asm» позволяет добавить некоторые «полезные» комментарии, а «-masm-intel» указывает на то, что идёт переход на язык ассемблера *типа интел*.

Результат работа компилятора:

.file "main.c"

.intel\_syntax noprefix

# GNU C99 (Ubuntu 11.3.0-1ubuntu1~22.04) version 11.3.0 (x86\_64-linux-gnu)

# compiled by GNU C version 11.3.0, GMP version 6.2.1, MPFR version 4.1.0, MPC version 1.2.1, isl version isl-0.24-GMP

# GGC heuristics: --param ggc-min-expand=100 --param ggc-min-heapsize=131072

# options passed: -masm=intel -mtune=generic -march=x86-64 -std=c99 -fasynchronous-unwind-tables -fstack-protector-strong -fstack-clash-protection -fcf-protection

.text

.section .rodata

.LC1:

.string "%lf"

.LC3:

.string "Invalid input!"

.text

.globl calculate\_d

.type calculate\_d, @function

calculate\_d:

.LFB0:

.cfi\_startproc

endbr64

push rbp #

.cfi\_def\_cfa\_offset 16

.cfi\_offset 6, -16

mov rbp, rsp #,

.cfi\_def\_cfa\_register 6

sub rsp, 32 #,

# main.c:6: {

mov rax, QWORD PTR fs:40 # tmp105, MEM[(<address-space-1> long unsigned int \*)40B]

mov QWORD PTR -8[rbp], rax # D.3217, tmp105

xor eax, eax # tmp105

# main.c:7: double d = 0.0;

pxor xmm0, xmm0 # tmp92

movsd QWORD PTR -16[rbp], xmm0 # d, tmp92

# main.c:8: double x = 0.0;

pxor xmm0, xmm0 # tmp93

movsd QWORD PTR -24[rbp], xmm0 # x, tmp93

# main.c:9: int n = 1;

mov DWORD PTR -28[rbp], 1 # n,

# main.c:10: scanf("%lf", &x);

lea rax, -24[rbp] # tmp94,

mov rsi, rax #, tmp94

lea rax, .LC1[rip] # tmp95,

mov rdi, rax #, tmp95

mov eax, 0 #,

call \_\_isoc99\_scanf@PLT #

# main.c:11: if (x < -EPS)

movsd xmm1, QWORD PTR -24[rbp] # x.0\_1, x

# main.c:11: if (x < -EPS)

movsd xmm0, QWORD PTR .LC2[rip] # tmp96,

comisd xmm0, xmm1 # tmp96, x.0\_1

jbe .L13 #,

# main.c:13: printf("Invalid input!\n");

lea rax, .LC3[rip] # tmp97,

mov rdi, rax #, tmp97

call puts@PLT #

# main.c:14: return -1.0;

movsd xmm0, QWORD PTR .LC4[rip] # \_11,

jmp .L10 #

.L13:

# main.c:16: while (x >= -EPS) // Повторяем чтение и вычисление для всех неотрицательных значений x (или не меньше -EPS)

jmp .L5 #

.L9:

# main.c:18: if (x < -EPS) // Если входное значение меньше -EPS (почти равное 0)

movsd xmm1, QWORD PTR -24[rbp] # x.1\_2, x

# main.c:18: if (x < -EPS) // Если входное значение меньше -EPS (почти равное 0)

movsd xmm0, QWORD PTR .LC2[rip] # tmp98,

comisd xmm0, xmm1 # tmp98, x.1\_2

ja .L14 #,

# main.c:22: d = d + sqrt(x / n); // Вычисляем квадратный корень из x / n и добавляем его к переменной d

movsd xmm0, QWORD PTR -24[rbp] # x.2\_3, x

pxor xmm1, xmm1 # \_4

cvtsi2sd xmm1, DWORD PTR -28[rbp] # \_4, n

divsd xmm0, xmm1 # x.2\_3, \_4

movq rax, xmm0 # \_5, x.2\_3

movq xmm0, rax #, \_5

call sqrt@PLT #

# main.c:22: d = d + sqrt(x / n); // Вычисляем квадратный корень из x / n и добавляем его к переменной d

movsd xmm1, QWORD PTR -16[rbp] # tmp100, d

addsd xmm0, xmm1 # tmp99, tmp100

movsd QWORD PTR -16[rbp], xmm0 # d, tmp99

# main.c:23: n += 1; // Увеличиваем переменную n на единицу

add DWORD PTR -28[rbp], 1 # n,

# main.c:24: if ((scanf("%lf", &x)) != 1)

lea rax, -24[rbp] # tmp101,

mov rsi, rax #, tmp101

lea rax, .LC1[rip] # tmp102,

mov rdi, rax #, tmp102

mov eax, 0 #,

call \_\_isoc99\_scanf@PLT #

# main.c:24: if ((scanf("%lf", &x)) != 1)

cmp eax, 1 # \_7,

je .L5 #,

# main.c:26: printf("Invalid input!\n");

lea rax, .LC3[rip] # tmp103,

mov rdi, rax #, tmp103

call puts@PLT #

# main.c:27: return -1.0;

movsd xmm0, QWORD PTR .LC4[rip] # \_11,

jmp .L10 #

.L5:

# main.c:16: while (x >= -EPS) // Повторяем чтение и вычисление для всех неотрицательных значений x (или не меньше -EPS)

movsd xmm0, QWORD PTR -24[rbp] # x.3\_8, x

comisd xmm0, QWORD PTR .LC2[rip] # x.3\_8,

jnb .L9 #,

jmp .L8 #

.L14:

# main.c:20: break;

nop

.L8:

# main.c:31: return d;

movsd xmm0, QWORD PTR -16[rbp] # \_11, d

.L10:

# main.c:32: }

movq rax, xmm0 # <retval>, \_11

mov rdx, QWORD PTR -8[rbp] # tmp106, D.3217

sub rdx, QWORD PTR fs:40 # tmp106, MEM[(<address-space-1> long unsigned int \*)40B]

je .L11 #,

# main.c:32: }

call \_\_stack\_chk\_fail@PLT #

.L11:

movq xmm0, rax #, <retval>

leave

.cfi\_def\_cfa 7, 8

ret

.cfi\_endproc

.LFE0:

.size calculate\_d, .-calculate\_d

.globl main

.type main, @function

main:

.LFB1:

.cfi\_startproc

endbr64

push rbp #

.cfi\_def\_cfa\_offset 16

.cfi\_offset 6, -16

mov rbp, rsp #,

.cfi\_def\_cfa\_register 6

sub rsp, 16 #,

# main.c:36: double d = calculate\_d();

call calculate\_d #

movq rax, xmm0 # tmp84,

mov QWORD PTR -16[rbp], rax # d, tmp84

# main.c:37: if (d < 0.0)

pxor xmm0, xmm0 # tmp85

comisd xmm0, QWORD PTR -16[rbp] # tmp85, d

jbe .L20 #,

# main.c:39: return 1;

mov eax, 1 # \_1,

jmp .L18 #

.L20:

# main.c:41: double gx = sin(d);

mov rax, QWORD PTR -16[rbp] # tmp86, d

movq xmm0, rax #, tmp86

call sin@PLT #

movq rax, xmm0 # tmp87,

mov QWORD PTR -8[rbp], rax # gx, tmp87

# main.c:42: printf("%lf", gx);

mov rax, QWORD PTR -8[rbp] # tmp88, gx

movq xmm0, rax #, tmp88

lea rax, .LC1[rip] # tmp89,

mov rdi, rax #, tmp89

mov eax, 1 #,

call printf@PLT #

# main.c:43: return 0;

mov eax, 0 # \_1,

.L18:

# main.c:44: }

leave

.cfi\_def\_cfa 7, 8

ret

.cfi\_endproc

.LFE1:

.size main, .-main

.section .rodata

.align 8

.LC2:

.long -1598689907

.long -1095710985

.align 8

.LC4:

.long 0

.long -1074790400

.ident "GCC: (Ubuntu 11.3.0-1ubuntu1~22.04) 11.3.0"

.section .note.GNU-stack,"",@progbits

.section .note.gnu.property,"a"

.align 8

.long 1f - 0f

.long 4f - 1f

.long 5

0:

.string "GNU"

1:

.align 8

.long 0xc0000002

.long 3f - 2f

2:

.long 0x3

3:

.align 8

4:

**Ассемблирование**  
На данном этапе код, что в текущий момент находится на языке ассемблера, переводится в машинный при помощи ретранслятора.

**$ as main.s -o main.o**

**$ hexdump main.o**

dunglasoi@DESKTOP-JG9N5AM:/mnt/d/LabC/LabC/lab\_01\_09\_01$ hexdump main.o

0000000 457f 464c 0102 0001 0000 0000 0000 0000

0000010 0001 003e 0001 0000 0000 0000 0000 0000

0000020 0000 0000 0000 0000 0698 0000 0000 0000

0000030 0000 0000 0040 0000 0000 0040 000e 000d

0000040 0ff3 fa1e 4855 e589 8348 20ec 4864 048b

0000050 2825 0000 4800 4589 31f8 66c0 ef0f f2c0

0000060 110f f045 0f66 c0ef 0ff2 4511 c7e8 e445

0000070 0001 0000 8d48 e845 8948 48c6 058d 0000

0000080 0000 8948 b8c7 0000 0000 00e8 0000 f200

0000090 100f e84d 0ff2 0510 0000 0000 0f66 c12f

00000a0 1c76 8d48 0005 0000 4800 c789 00e8 0000

00000b0 f200 100f 0005 0000 e900 00a3 0000 83e9

00000c0 0000 f200 100f e84d 0ff2 0510 0000 0000

00000d0 0f66 c12f 870f 0081 0000 0ff2 4510 66e8

00000e0 ef0f f2c9 2a0f e44d 0ff2 c15e 4866 7e0f

00000f0 66c0 0f48 c06e 00e8 0000 f200 100f f04d

0000100 0ff2 c158 0ff2 4511 83f0 e445 4801 458d

0000110 48e8 c689 8d48 0005 0000 4800 c789 00b8

0000120 0000 e800 0000 0000 f883 7401 4819 058d

0000130 0000 0000 8948 e8c7 0000 0000 0ff2 0510

0000140 0000 0000 1beb 0ff2 4510 66e8 2f0f 0005

0000150 0000 0f00 6a83 ffff ebff 9001 0ff2 4510

0000160 66f0 0f48 c07e 8b48 f855 4864 142b 2825

0000170 0000 7400 e805 0000 0000 4866 6e0f c9c0

0000180 f3c3 1e0f 55fa 8948 48e5 ec83 e810 0000

0000190 0000 4866 7e0f 48c0 4589 66f0 ef0f 66c0

00001a0 2f0f f045 0776 01b8 0000 eb00 4839 458b

00001b0 66f0 0f48 c06e 00e8 0000 6600 0f48 c07e

00001c0 8948 f845 8b48 f845 4866 6e0f 48c0 058d

00001d0 0000 0000 8948 b8c7 0001 0000 00e8 0000

00001e0 b800 0000 0000 c3c9 6c25 0066 6e49 6176

00001f0 696c 2064 6e69 7570 2174 0000 0000 0000

0000200 ed8d a0b5 c6f7 beb0 0000 0000 0000 bff0

0000210 4700 4343 203a 5528 7562 746e 2075 3131

0000220 332e 302e 312d 6275 6e75 7574 7e31 3232

0000230 302e 2934 3120 2e31 2e33 0030 0000 0000

0000240 0004 0000 0010 0000 0005 0000 4e47 0055

0000250 0002 c000 0004 0000 0003 0000 0000 0000

0000260 0014 0000 0000 0000 7a01 0052 7801 0110

0000270 0c1b 0807 0190 0000 001c 0000 001c 0000

0000280 0000 0000 0141 0000 4500 100e 0286 0d43

0000290 0306 0138 070c 0008 001c 0000 003c 0000

00002a0 0000 0000 0067 0000 4500 100e 0286 0d43

00002b0 0206 0c5e 0807 0000 0000 0000 0000 0000

00002c0 0000 0000 0000 0000 0000 0000 0000 0000

00002d0 0001 0000 0004 fff1 0000 0000 0000 0000

00002e0 0000 0000 0000 0000 0000 0000 0003 0001

00002f0 0000 0000 0000 0000 0000 0000 0000 0000

0000300 0000 0000 0003 0005 0000 0000 0000 0000

0000310 0000 0000 0000 0000 0008 0000 0012 0001

0000320 0000 0000 0000 0000 0141 0000 0000 0000

0000330 0014 0000 0010 0000 0000 0000 0000 0000

0000340 0000 0000 0000 0000 0023 0000 0010 0000

0000350 0000 0000 0000 0000 0000 0000 0000 0000

0000360 0028 0000 0010 0000 0000 0000 0000 0000

0000370 0000 0000 0000 0000 002d 0000 0010 0000

0000380 0000 0000 0000 0000 0000 0000 0000 0000

0000390 003e 0000 0012 0001 0141 0000 0000 0000

00003a0 0067 0000 0000 0000 0043 0000 0010 0000

00003b0 0000 0000 0000 0000 0000 0000 0000 0000

00003c0 0047 0000 0010 0000 0000 0000 0000 0000

00003d0 0000 0000 0000 0000 6d00 6961 2e6e 0063

00003e0 6163 636c 6c75 7461 5f65 0064 5f5f 7369

00003f0 636f 3939 735f 6163 666e 7000 7475 0073

0000400 7173 7472 5f00 735f 6174 6b63 635f 6b68

0000410 665f 6961 006c 616d 6e69 7300 6e69 7000

0000420 6972 746e 0066 0000 003e 0000 0000 0000

0000430 0002 0000 0003 0000 fffc ffff ffff ffff

0000440 004b 0000 0000 0000 0004 0000 0005 0000

0000450 fffc ffff ffff ffff 0058 0000 0000 0000

0000460 0002 0000 0003 0000 0014 0000 0000 0000

0000470 0065 0000 0000 0000 0002 0000 0003 0000

0000480 0000 0000 0000 0000 006d 0000 0000 0000

0000490 0004 0000 0006 0000 fffc ffff ffff ffff

00004a0 0075 0000 0000 0000 0002 0000 0003 0000

00004b0 001c 0000 0000 0000 008c 0000 0000 0000

00004c0 0002 0000 0003 0000 0014 0000 0000 0000

00004d0 00b7 0000 0000 0000 0004 0000 0007 0000

00004e0 fffc ffff ffff ffff 00d7 0000 0000 0000

00004f0 0002 0000 0003 0000 fffc ffff ffff ffff

0000500 00e4 0000 0000 0000 0004 0000 0005 0000

0000510 fffc ffff ffff ffff 00f0 0000 0000 0000

0000520 0002 0000 0003 0000 0000 0000 0000 0000

0000530 00f8 0000 0000 0000 0004 0000 0006 0000

0000540 fffc ffff ffff ffff 0100 0000 0000 0000

0000550 0002 0000 0003 0000 001c 0000 0000 0000

0000560 010f 0000 0000 0000 0002 0000 0003 0000

0000570 0014 0000 0000 0000 0136 0000 0000 0000

0000580 0004 0000 0008 0000 fffc ffff ffff ffff

0000590 014e 0000 0000 0000 0004 0000 0004 0000

00005a0 fffc ffff ffff ffff 0177 0000 0000 0000

00005b0 0004 0000 000a 0000 fffc ffff ffff ffff

00005c0 0190 0000 0000 0000 0002 0000 0003 0000

00005d0 fffc ffff ffff ffff 019d 0000 0000 0000

00005e0 0004 0000 000b 0000 fffc ffff ffff ffff

00005f0 0020 0000 0000 0000 0002 0000 0002 0000

0000600 0000 0000 0000 0000 0040 0000 0000 0000

0000610 0002 0000 0002 0000 0141 0000 0000 0000

0000620 2e00 7973 746d 6261 2e00 7473 7472 6261

0000630 2e00 6873 7473 7472 6261 2e00 6572 616c

0000640 742e 7865 0074 642e 7461 0061 622e 7373

0000650 2e00 6f72 6164 6174 2e00 6f63 6d6d 6e65

0000660 0074 6e2e 746f 2e65 4e47 2d55 7473 6361

0000670 006b 6e2e 746f 2e65 6e67 2e75 7270 706f

0000680 7265 7974 2e00 6572 616c 652e 5f68 7266

0000690 6d61 0065 0000 0000 0000 0000 0000 0000

00006a0 0000 0000 0000 0000 0000 0000 0000 0000

\*

00006d0 0000 0000 0000 0000 0020 0000 0001 0000

00006e0 0006 0000 0000 0000 0000 0000 0000 0000

00006f0 0040 0000 0000 0000 01a8 0000 0000 0000

0000700 0000 0000 0000 0000 0001 0000 0000 0000

0000710 0000 0000 0000 0000 001b 0000 0004 0000

0000720 0040 0000 0000 0000 0000 0000 0000 0000

0000730 0428 0000 0000 0000 01c8 0000 0000 0000

0000740 000b 0000 0001 0000 0008 0000 0000 0000

0000750 0018 0000 0000 0000 0026 0000 0001 0000

0000760 0003 0000 0000 0000 0000 0000 0000 0000

0000770 01e8 0000 0000 0000 0000 0000 0000 0000

0000780 0000 0000 0000 0000 0001 0000 0000 0000

0000790 0000 0000 0000 0000 002c 0000 0008 0000

00007a0 0003 0000 0000 0000 0000 0000 0000 0000

00007b0 01e8 0000 0000 0000 0000 0000 0000 0000

00007c0 0000 0000 0000 0000 0001 0000 0000 0000

00007d0 0000 0000 0000 0000 0031 0000 0001 0000

00007e0 0002 0000 0000 0000 0000 0000 0000 0000

00007f0 01e8 0000 0000 0000 0028 0000 0000 0000

0000800 0000 0000 0000 0000 0008 0000 0000 0000

0000810 0000 0000 0000 0000 0039 0000 0001 0000

0000820 0030 0000 0000 0000 0000 0000 0000 0000

0000830 0210 0000 0000 0000 002c 0000 0000 0000

0000840 0000 0000 0000 0000 0001 0000 0000 0000

0000850 0001 0000 0000 0000 0042 0000 0001 0000

0000860 0000 0000 0000 0000 0000 0000 0000 0000

0000870 023c 0000 0000 0000 0000 0000 0000 0000

0000880 0000 0000 0000 0000 0001 0000 0000 0000

0000890 0000 0000 0000 0000 0052 0000 0007 0000

00008a0 0002 0000 0000 0000 0000 0000 0000 0000

00008b0 0240 0000 0000 0000 0020 0000 0000 0000

00008c0 0000 0000 0000 0000 0008 0000 0000 0000

00008d0 0000 0000 0000 0000 006a 0000 0001 0000

00008e0 0002 0000 0000 0000 0000 0000 0000 0000

00008f0 0260 0000 0000 0000 0058 0000 0000 0000

0000900 0000 0000 0000 0000 0008 0000 0000 0000

0000910 0000 0000 0000 0000 0065 0000 0004 0000

0000920 0040 0000 0000 0000 0000 0000 0000 0000

0000930 05f0 0000 0000 0000 0030 0000 0000 0000

0000940 000b 0000 0009 0000 0008 0000 0000 0000

0000950 0018 0000 0000 0000 0001 0000 0002 0000

0000960 0000 0000 0000 0000 0000 0000 0000 0000

0000970 02b8 0000 0000 0000 0120 0000 0000 0000

0000980 000c 0000 0004 0000 0008 0000 0000 0000

0000990 0018 0000 0000 0000 0009 0000 0003 0000

00009a0 0000 0000 0000 0000 0000 0000 0000 0000

00009b0 03d8 0000 0000 0000 004e 0000 0000 0000

00009c0 0000 0000 0000 0000 0001 0000 0000 0000

00009d0 0000 0000 0000 0000 0011 0000 0003 0000

00009e0 0000 0000 0000 0000 0000 0000 0000 0000

00009f0 0620 0000 0000 0000 0074 0000 0000 0000

0000a00 0000 0000 0000 0000 0001 0000 0000 0000

0000a10 0000 0000 0000 0000

0000a18

Причём, следует отметить, что на выходе получается не текстовый файл, а двоичный, что ещё называется *объектным*.

* **Компоновка**
  + Объединение нескольких объектных файлов
  + Связывание переменных и функций, которые требуются очередному объектному файлу, но находятся где-то в другом месте
  + Добавление специального кода, который подготавливает окружение для вызова функции main, а после её завершения выполняет обратные действия

Данный этап выполняется командой *ld*:

**$ ld -dynamic-linker /lib64/ld-linux-x86-64.so.2 \**

**/usr/lib/x86\_64-linux-gnu/crt1.o \**

**/usr/lib/x86\_64-linux-gnu/crti.o -lc \**

**/usr/lib/x86\_64-linux-gnu/crtn.o main.o -o main.exe -lm**

# Почему **gcc** и **clang** называются «программами-драйверами»

GCC и Clang называются "программами-драйверами" потому, что они являются основными компиляторами для языка программирования C/C++. Они управляют процессом компиляции и создания исполняемых файлов для программ, написанных на C/C++.  
  
Программы-драйверы, в частности, предоставляют интерфейс для передачи параметров компиляции, таких как опции оптимизации, флаги безопасности и другие параметры. Они также обрабатывают зависимости между файлами и проверяют ошибки компиляции.  
  
Помимо этого, GCC и Clang являются крайне гибкими и могут использоваться в различных средах и на разных платформах. Они могут компилировать программы для множества архитектур процессоров, включая x86, ARM, PowerPC и другие, а также для многих операционных систем, таких как Linux, macOS, Windows и других. Это позволяет разработчикам писать программы на языке C/C++ и выполнять их на различных платформах, что является важным преимуществом для создания кроссплатформенных приложений.

1. Top of Form

# **4**. Этапы и их выполнение для получения исполняемого файла компилятором **gcc**

* **Компиляция**На данном этапе **gcc** делает препроцессирование и компиляцию в ассемблерный код на одном этапе с помощью следующей команды:

**$ gcc -E main.c -o main.i**

**$ gcc -v main.c -o main.s**

* **Ассемблирование**На данном этапе **gcc** делает перевод ассемблерный код в машинный с помощью следующей команды:

**$ as -v –64 -o main.o main.s**

* **Компоновка**На данном этапе **gcc** связывает объектный файл, созданный этапом выше, с другими объектными файлами, создавая исполняемый:

**$ gcc -v -save-temps -o app.exe main.c -lm**

-v: Отображает шаги и команды, выполненные во время компиляции, а также другую информацию, такую ​​как входные файлы, выходные файлы, пути к каталогам, используемые файлы библиотек и прочее.

-save-temps: Сохранения временных файлов, созданных в процессе компиляции.

(a) Разница между этими шагами и моими зависит от того, какие опции и флаги использовались при компиляции моей программы.

(b) Временные файлы, созданные с помощью ключа -save-temps, содержат промежуточные результаты компиляции..

(c) Эти файлы отличаются от файлов, полученных во втором пункте, тем, что они являются временными и не предназначены для окончательной сборки в исполняемый файл. Файлы, созданные во втором пункте, являются конечным результатом компиляции и могут быть использованы для создания исполняемого файла.

(d) Когда программа компонуется, объектные файлы создаются во время компиляции, а статические или динамические библиотеки объединяются. Компоновщик связывает эти файлы в один исполняемый файл..

(e) Объектные файлы используются для хранения промежуточного кода, созданного во время компиляции исходных файлов. Эти файлы можно использовать для создания исполняемого файла.

# Этапы и их выполнение для получения исполняемого файла компилятором **clang**

Компилятор clang выполняет следующие этапы для получения исполняемого файла:

+ Препроцессинг: Это первый шаг в процессе компиляции, когда компилятор читает и обрабатывает специальные директивы, такие как #include, #define, #ifdef,... На этом этапе компилятор также определяет директивы, библиотеки и файлы препроцессора, которые будут использоваться.

Команда для выполнения этапа препроцессинга в clang:

**$ clang -E input\_file.c -o output\_file.i**

+ Компиляция: В этом этапе компилятор преобразует промежуточный файл с расширением .i в объектный файл с расширением .o, содержащий машинный код.

Команда для выполнения этапа компиляции в clang:

**$ clang -c input\_file.c -o output\_file.o**

+ Компоновка: в этом этапе компоновщик (линкер) объединяет объектные файлы и статические библиотеки, создавая исполняемый файл.

Команда для выполнения этапа компоновки в clang:

**$ clang object\_file1.o object\_file2.o -o output\_file**

Здесь object\_file1.o и object\_file2.o - это объектные файлы, которые необходимо скомпоновать, а output\_file - это имя исполняемого файла, который будет создан.

# Ассемблерный листинг в **gcc**

С помощью команды *help* мы можем узнать, какие ключи нужны для передачи параметров компилятору с языка ассемблера:

* *-Xassembler*

**$ gcc -Xassembler -a=main\_asm.s main.c -lm**

* *-Wa*

**$ gcc -Wa,-a=main\_asm.s main.c -lm**

Ассемблерный листинг при помощи первой команды:

GAS LISTING /tmp/ccZcwpRP.s page 1

1 .file "main.c"

2 .text

3 .section .rodata

4 .LC1:

5 0000 256C6600 .string "%lf"

6 .LC3:

7 0004 496E7661 .string "Invalid input!"

7 6C696420

7 696E7075

7 742100

8 .text

9 .globl calculate\_d

10 .type calculate\_d, @function

11 calculate\_d:

12 .LFB0:

13 .cfi\_startproc

14 0000 F30F1EFA endbr64

15 0004 55 pushq %rbp

16 .cfi\_def\_cfa\_offset 16

17 .cfi\_offset 6, -16

18 0005 4889E5 movq %rsp, %rbp

19 .cfi\_def\_cfa\_register 6

20 0008 4883EC20 subq $32, %rsp

21 000c 64488B04 movq %fs:40, %rax

21 25280000

21 00

22 0015 488945F8 movq %rax, -8(%rbp)

23 0019 31C0 xorl %eax, %eax

24 001b 660FEFC0 pxor %xmm0, %xmm0

25 001f F20F1145 movsd %xmm0, -16(%rbp)

25 F0

26 0024 660FEFC0 pxor %xmm0, %xmm0

27 0028 F20F1145 movsd %xmm0, -24(%rbp)

27 E8

28 002d C745E401 movl $1, -28(%rbp)

28 000000

29 0034 488D45E8 leaq -24(%rbp), %rax

30 0038 4889C6 movq %rax, %rsi

31 003b 488D0500 leaq .LC1(%rip), %rax

31 000000

32 0042 4889C7 movq %rax, %rdi

33 0045 B8000000 movl $0, %eax

33 00

34 004a E8000000 call \_\_isoc99\_scanf@PLT

34 00

35 004f F20F104D movsd -24(%rbp), %xmm1

35 E8

36 0054 F20F1005 movsd .LC2(%rip), %xmm0

36 00000000

37 005c 660F2FC1 comisd %xmm1, %xmm0

38 0060 761C jbe .L13

39 0062 488D0500 leaq .LC3(%rip), %rax

39 000000

40 0069 4889C7 movq %rax, %rdi

41 006c E8000000 call puts@PLT

41 00

42 0071 F20F1005 movsd .LC4(%rip), %xmm0

GAS LISTING /tmp/ccZcwpRP.s page 2

42 00000000

43 0079 E9A30000 jmp .L10

43 00

44 .L13:

45 007e E9830000 jmp .L5

45 00

46 .L9:

47 0083 F20F104D movsd -24(%rbp), %xmm1

47 E8

48 0088 F20F1005 movsd .LC2(%rip), %xmm0

48 00000000

49 0090 660F2FC1 comisd %xmm1, %xmm0

50 0094 0F878100 ja .L14

50 0000

51 009a F20F1045 movsd -24(%rbp), %xmm0

51 E8

52 009f 660FEFC9 pxor %xmm1, %xmm1

53 00a3 F20F2A4D cvtsi2sdl -28(%rbp), %xmm1

53 E4

54 00a8 F20F5EC1 divsd %xmm1, %xmm0

55 00ac 66480F7E movq %xmm0, %rax

55 C0

56 00b1 66480F6E movq %rax, %xmm0

56 C0

57 00b6 E8000000 call sqrt@PLT

57 00

58 00bb F20F104D movsd -16(%rbp), %xmm1

58 F0

59 00c0 F20F58C1 addsd %xmm1, %xmm0

60 00c4 F20F1145 movsd %xmm0, -16(%rbp)

60 F0

61 00c9 8345E401 addl $1, -28(%rbp)

62 00cd 488D45E8 leaq -24(%rbp), %rax

63 00d1 4889C6 movq %rax, %rsi

64 00d4 488D0500 leaq .LC1(%rip), %rax

64 000000

65 00db 4889C7 movq %rax, %rdi

66 00de B8000000 movl $0, %eax

66 00

67 00e3 E8000000 call \_\_isoc99\_scanf@PLT

67 00

68 00e8 83F801 cmpl $1, %eax

69 00eb 7419 je .L5

70 00ed 488D0500 leaq .LC3(%rip), %rax

70 000000

71 00f4 4889C7 movq %rax, %rdi

72 00f7 E8000000 call puts@PLT

72 00

73 00fc F20F1005 movsd .LC4(%rip), %xmm0

73 00000000

74 0104 EB1B jmp .L10

75 .L5:

76 0106 F20F1045 movsd -24(%rbp), %xmm0

76 E8

77 010b 660F2F05 comisd .LC2(%rip), %xmm0

77 00000000

78 0113 0F836AFF jnb .L9

GAS LISTING /tmp/ccZcwpRP.s page 3

78 FFFF

79 0119 EB01 jmp .L8

80 .L14:

81 011b 90 nop

82 .L8:

83 011c F20F1045 movsd -16(%rbp), %xmm0

83 F0

84 .L10:

85 0121 66480F7E movq %xmm0, %rax

85 C0

86 0126 488B55F8 movq -8(%rbp), %rdx

87 012a 64482B14 subq %fs:40, %rdx

87 25280000

87 00

88 0133 7405 je .L11

89 0135 E8000000 call \_\_stack\_chk\_fail@PLT

89 00

90 .L11:

91 013a 66480F6E movq %rax, %xmm0

91 C0

92 013f C9 leave

93 .cfi\_def\_cfa 7, 8

94 0140 C3 ret

95 .cfi\_endproc

96 .LFE0:

97 .size calculate\_d, .-calculate\_d

98 .globl main

99 .type main, @function

100 main:

101 .LFB1:

102 .cfi\_startproc

103 0141 F30F1EFA endbr64

104 0145 55 pushq %rbp

105 .cfi\_def\_cfa\_offset 16

106 .cfi\_offset 6, -16

107 0146 4889E5 movq %rsp, %rbp

108 .cfi\_def\_cfa\_register 6

109 0149 4883EC10 subq $16, %rsp

110 014d E8000000 call calculate\_d

110 00

111 0152 66480F7E movq %xmm0, %rax

111 C0

112 0157 488945F0 movq %rax, -16(%rbp)

113 015b 660FEFC0 pxor %xmm0, %xmm0

114 015f 660F2F45 comisd -16(%rbp), %xmm0

114 F0

115 0164 7607 jbe .L20

116 0166 B8010000 movl $1, %eax

116 00

117 016b EB39 jmp .L18

118 .L20:

119 016d 488B45F0 movq -16(%rbp), %rax

120 0171 66480F6E movq %rax, %xmm0

120 C0

121 0176 E8000000 call sin@PLT

121 00

122 017b 66480F7E movq %xmm0, %rax

GAS LISTING /tmp/ccZcwpRP.s page 4

122 C0

123 0180 488945F8 movq %rax, -8(%rbp)

124 0184 488B45F8 movq -8(%rbp), %rax

125 0188 66480F6E movq %rax, %xmm0

125 C0

126 018d 488D0500 leaq .LC1(%rip), %rax

126 000000

127 0194 4889C7 movq %rax, %rdi

128 0197 B8010000 movl $1, %eax

128 00

129 019c E8000000 call printf@PLT

129 00

130 01a1 B8000000 movl $0, %eax

130 00

131 .L18:

132 01a6 C9 leave

133 .cfi\_def\_cfa 7, 8

134 01a7 C3 ret

135 .cfi\_endproc

136 .LFE1:

137 .size main, .-main

138 .section .rodata

139 0013 00000000 .align 8

139 00

140 .LC2:

141 0018 8DEDB5A0 .long -1598689907

142 001c F7C6B0BE .long -1095710985

143 .align 8

144 .LC4:

145 0020 00000000 .long 0

146 0024 0000F0BF .long -1074790400

147 .ident "GCC: (Ubuntu 11.3.0-1ubuntu1~22.04) 11.3.0"

148 .section .note.GNU-stack,"",@progbits

149 .section .note.gnu.property,"a"

150 .align 8

151 0000 04000000 .long 1f - 0f

152 0004 10000000 .long 4f - 1f

153 0008 05000000 .long 5

154 0:

155 000c 474E5500 .string "GNU"

156 1:

157 .align 8

158 0010 020000C0 .long 0xc0000002

159 0014 04000000 .long 3f - 2f

160 2:

161 0018 03000000 .long 0x3

162 3:

163 001c 00000000 .align 8

164 4:

GAS LISTING /tmp/ccZcwpRP.s page 5

DEFINED SYMBOLS

ABS:0000000000000000 main.c

/tmp/ccZcwpRP.s:11 .text:0000000000000000 calculate\_d

/tmp/ccZcwpRP.s:100 .text:0000000000000141 main

UNDEFINED SYMBOLS

\_\_isoc99\_scanf

puts

sqrt

\_\_stack\_chk\_fail

sin

printf

# Создание map-файла в **gcc**

С помощью команды *help* мы можем узнать, какие ключи нужны для передачи параметров компоновщику:

* *-Xlinker*

**$ gcc -Xlinker -Map=main.map main.c -lm**

* -*Wl*

**$ gcc -Wl,-Map=main.map main.c -lm**

Создание map-файла при помощи первой команды:

Для удовлетворения ссылок на файл (символ) включён член архива

Merging program properties

As-needed library included to satisfy reference by file (symbol)

libm.so.6 /tmp/ccTlOlX6.o (sqrt@@GLIBC\_2.2.5)

libc.so.6 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o (\_\_libc\_start\_main@@GLIBC\_2.34)

Discarded input sections

.note.GNU-stack

0x0000000000000000 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.note.gnu.property

0x0000000000000000 0x20 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crti.o

.note.GNU-stack

0x0000000000000000 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crti.o

.note.GNU-stack

0x0000000000000000 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/crtbeginS.o

.note.gnu.property

0x0000000000000000 0x20 /usr/lib/gcc/x86\_64-linux-gnu/11/crtbeginS.o

.note.GNU-stack

0x0000000000000000 0x0 /tmp/ccTlOlX6.o

.note.gnu.property

0x0000000000000000 0x20 /tmp/ccTlOlX6.o

.note.GNU-stack

0x0000000000000000 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/crtendS.o

.note.gnu.property

0x0000000000000000 0x20 /usr/lib/gcc/x86\_64-linux-gnu/11/crtendS.o

.note.gnu.property

0x0000000000000000 0x20 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crtn.o

.note.GNU-stack

0x0000000000000000 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crtn.o

Memory Configuration

Name Origin Length Attributes

default 0x0000000000000000 0xffffffffffffffff

Linker script and memory map

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crti.o

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/crtbeginS.o

LOAD /tmp/ccTlOlX6.o

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/libm.so

START GROUP

LOAD /lib/x86\_64-linux-gnu/libm.so.6

LOAD /lib/x86\_64-linux-gnu/libmvec.so.1

END GROUP

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/libgcc.a

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/libgcc\_s.so

START GROUP

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/libgcc\_s.so.1

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/libgcc.a

END GROUP

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/libc.so

START GROUP

LOAD /lib/x86\_64-linux-gnu/libc.so.6

LOAD /usr/lib/x86\_64-linux-gnu/libc\_nonshared.a

LOAD /lib64/ld-linux-x86-64.so.2

END GROUP

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/libgcc.a

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/libgcc\_s.so

START GROUP

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/libgcc\_s.so.1

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/libgcc.a

END GROUP

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/crtendS.o

LOAD /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crtn.o

[!provide] PROVIDE (\_\_executable\_start = SEGMENT\_START ("text-segment", 0x0))

0x0000000000000318 . = (SEGMENT\_START ("text-segment", 0x0) + SIZEOF\_HEADERS)

.interp 0x0000000000000318 0x1c

\*(.interp)

.interp 0x0000000000000318 0x1c /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.note.gnu.property

0x0000000000000338 0x30

.note.gnu.property

0x0000000000000338 0x30 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.note.gnu.build-id

0x0000000000000368 0x24

\*(.note.gnu.build-id)

.note.gnu.build-id

0x0000000000000368 0x24 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.note.ABI-tag 0x000000000000038c 0x20

.note.ABI-tag 0x000000000000038c 0x20 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.hash

\*(.hash)

.gnu.hash 0x00000000000003b0 0x24

\*(.gnu.hash)

.gnu.hash 0x00000000000003b0 0x24 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.dynsym 0x00000000000003d8 0x120

\*(.dynsym)

.dynsym 0x00000000000003d8 0x120 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.dynstr 0x00000000000004f8 0xdb

\*(.dynstr)

.dynstr 0x00000000000004f8 0xdb /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.gnu.version 0x00000000000005d4 0x18

\*(.gnu.version)

.gnu.version 0x00000000000005d4 0x18 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.gnu.version\_d 0x00000000000005f0 0x0

\*(.gnu.version\_d)

.gnu.version\_d

0x00000000000005f0 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.gnu.version\_r 0x00000000000005f0 0x70

\*(.gnu.version\_r)

.gnu.version\_r

0x00000000000005f0 0x70 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.rela.dyn 0x0000000000000660 0xc0

\*(.rela.init)

(.rela.text .rela.text. .rela.gnu.linkonce.t.\*)

\*(.rela.fini)

(.rela.rodata .rela.rodata. .rela.gnu.linkonce.r.\*)

(.rela.data .rela.data. .rela.gnu.linkonce.d.\*)

.rela.data.rel.ro

0x0000000000000660 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.rela.data.rel.local

0x0000000000000660 0x18 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

(.rela.tdata .rela.tdata. .rela.gnu.linkonce.td.\*)

(.rela.tbss .rela.tbss. .rela.gnu.linkonce.tb.\*)

\*(.rela.ctors)

\*(.rela.dtors)

\*(.rela.got)

.rela.got 0x0000000000000678 0x78 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

(.rela.bss .rela.bss. .rela.gnu.linkonce.b.\*)

.rela.bss 0x00000000000006f0 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

(.rela.ldata .rela.ldata. .rela.gnu.linkonce.l.\*)

(.rela.lbss .rela.lbss. .rela.gnu.linkonce.lb.\*)

(.rela.lrodata .rela.lrodata. .rela.gnu.linkonce.lr.\*)

\*(.rela.ifunc)

.rela.ifunc 0x00000000000006f0 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.rela.fini\_array

0x00000000000006f0 0x18 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.rela.init\_array

0x0000000000000708 0x18 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.rela.plt 0x0000000000000720 0x90

\*(.rela.plt)

.rela.plt 0x0000000000000720 0x90 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

\*(.rela.iplt)

.relr.dyn

\*(.relr.dyn)

0x0000000000001000 . = ALIGN (CONSTANT (MAXPAGESIZE))

.init 0x0000000000001000 0x1b

\*(SORT\_NONE(.init))

.init 0x0000000000001000 0x16 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crti.o

0x0000000000001000 \_init

.init 0x0000000000001016 0x5 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crtn.o

.plt 0x0000000000001020 0x70

\*(.plt)

.plt 0x0000000000001020 0x70 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

\*(.iplt)

.plt.got 0x0000000000001090 0x10

\*(.plt.got)

.plt.got 0x0000000000001090 0x10 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x0000000000001090 \_\_cxa\_finalize@@GLIBC\_2.2.5

.plt.sec 0x00000000000010a0 0x60

\*(.plt.sec)

.plt.sec 0x00000000000010a0 0x60 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x00000000000010a0 puts@@GLIBC\_2.2.5

0x00000000000010b0 \_\_stack\_chk\_fail@@GLIBC\_2.4

0x00000000000010c0 printf@@GLIBC\_2.2.5

0x00000000000010d0 sin@@GLIBC\_2.2.5

0x00000000000010e0 \_\_isoc99\_scanf@@GLIBC\_2.7

0x00000000000010f0 sqrt@@GLIBC\_2.2.5

.text 0x0000000000001100 0x291

(.text.unlikely .text.\_unlikely .text.unlikely.\*)

(.text.exit .text.exit.)

(.text.startup .text.startup.)

(.text.hot .text.hot.)

(SORT\_BY\_NAME(.text.sorted.))

(.text .stub .text. .gnu.linkonce.t.\*)

.text 0x0000000000001100 0x26 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x0000000000001100 \_start

.text 0x0000000000001126 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crti.o

fill 0x0000000000001126 0xa

.text 0x0000000000001130 0xb9 /usr/lib/gcc/x86\_64-linux-gnu/11/crtbeginS.o

.text 0x00000000000011e9 0x1a8 /tmp/ccTlOlX6.o

0x00000000000011e9 calculate\_d

0x000000000000132a main

.text 0x0000000000001391 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/crtendS.o

.text 0x0000000000001391 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crtn.o

\*(.gnu.warning)

.fini 0x0000000000001394 0xd

\*(SORT\_NONE(.fini))

.fini 0x0000000000001394 0x8 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crti.o

0x0000000000001394 \_fini

.fini 0x000000000000139c 0x5 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crtn.o

[!provide] PROVIDE (\_\_etext = .)

[!provide] PROVIDE (\_etext = .)

[!provide] PROVIDE (etext = .)

0x0000000000002000 . = ALIGN (CONSTANT (MAXPAGESIZE))

0x0000000000002000 . = SEGMENT\_START ("rodata-segment", (ALIGN (CONSTANT (MAXPAGESIZE)) + (. & (CONSTANT (MAXPAGESIZE) - 0x1))))

.rodata 0x0000000000002000 0x30

(.rodata .rodata. .gnu.linkonce.r.\*)

.rodata.cst4 0x0000000000002000 0x4 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x0000000000002000 \_IO\_stdin\_used

fill 0x0000000000002004 0x4

.rodata 0x0000000000002008 0x28 /tmp/ccTlOlX6.o

.rodata1

\*(.rodata1)

.eh\_frame\_hdr 0x0000000000002030 0x3c

\*(.eh\_frame\_hdr)

.eh\_frame\_hdr 0x0000000000002030 0x3c /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x0000000000002030 \_\_GNU\_EH\_FRAME\_HDR

(.eh\_frame\_entry .eh\_frame\_entry.)

.eh\_frame 0x0000000000002070 0xcc

\*(.eh\_frame)

.eh\_frame 0x0000000000002070 0x30 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x2c (size before relaxing)

fill 0x00000000000020a0 0x0

.eh\_frame 0x00000000000020a0 0x28 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x40 (size before relaxing)

.eh\_frame 0x00000000000020c8 0x18 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x30 (size before relaxing)

.eh\_frame 0x00000000000020e0 0x18 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x30 (size before relaxing)

.eh\_frame 0x00000000000020f8 0x40 /tmp/ccTlOlX6.o

0x58 (size before relaxing)

.eh\_frame 0x0000000000002138 0x4 /usr/lib/gcc/x86\_64-linux-gnu/11/crtendS.o

(.eh\_frame.)

.gcc\_except\_table

(.gcc\_except\_table .gcc\_except\_table.)

.gnu\_extab

(.gnu\_extab)

.exception\_ranges

(.exception\_ranges)

0x0000000000003d80 . = DATA\_SEGMENT\_ALIGN (CONSTANT (MAXPAGESIZE), CONSTANT (COMMONPAGESIZE))

.eh\_frame

\*(.eh\_frame)

(.eh\_frame.)

.gnu\_extab

\*(.gnu\_extab)

.gcc\_except\_table

(.gcc\_except\_table .gcc\_except\_table.)

.exception\_ranges

(.exception\_ranges)

.tdata 0x0000000000003d80 0x0

[!provide] PROVIDE (\_\_tdata\_start = .)

(.tdata .tdata. .gnu.linkonce.td.\*)

.tbss

(.tbss .tbss. .gnu.linkonce.tb.\*)

\*(.tcommon)

.preinit\_array 0x0000000000003d80 0x0

[!provide] PROVIDE (\_\_preinit\_array\_start = .)

\*(.preinit\_array)

[!provide] PROVIDE (\_\_preinit\_array\_end = .)

.init\_array 0x0000000000003d80 0x8

[!provide] PROVIDE (\_\_init\_array\_start = .)

(SORT\_BY\_INIT\_PRIORITY(.init\_array.) SORT\_BY\_INIT\_PRIORITY(.ctors.\*))

\*(.init\_array EXCLUDE\_FILE(\*crtend?.o \*crtend.o \*crtbegin?.o \*crtbegin.o) .ctors)

.init\_array 0x0000000000003d80 0x8 /usr/lib/gcc/x86\_64-linux-gnu/11/crtbeginS.o

[!provide] PROVIDE (\_\_init\_array\_end = .)

.fini\_array 0x0000000000003d88 0x8

[!provide] PROVIDE (\_\_fini\_array\_start = .)

(SORT\_BY\_INIT\_PRIORITY(.fini\_array.) SORT\_BY\_INIT\_PRIORITY(.dtors.\*))

\*(.fini\_array EXCLUDE\_FILE(\*crtend?.o \*crtend.o \*crtbegin?.o \*crtbegin.o) .dtors)

.fini\_array 0x0000000000003d88 0x8 /usr/lib/gcc/x86\_64-linux-gnu/11/crtbeginS.o

[!provide] PROVIDE (\_\_fini\_array\_end = .)

.ctors

\*crtbegin.o(.ctors)

\*crtbegin?.o(.ctors)

\*(EXCLUDE\_FILE(\*crtend?.o \*crtend.o) .ctors)

(SORT\_BY\_NAME(.ctors.))

\*(.ctors)

.dtors

\*crtbegin.o(.dtors)

\*crtbegin?.o(.dtors)

\*(EXCLUDE\_FILE(\*crtend?.o \*crtend.o) .dtors)

(SORT\_BY\_NAME(.dtors.))

\*(.dtors)

.jcr

\*(.jcr)

.data.rel.ro 0x0000000000003d90 0x0

(.data.rel.ro.local .gnu.linkonce.d.rel.ro.local.\*)

(.data.rel.ro .data.rel.ro. .gnu.linkonce.d.rel.ro.\*)

.data.rel.ro 0x0000000000003d90 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.dynamic 0x0000000000003d90 0x200

\*(.dynamic)

.dynamic 0x0000000000003d90 0x200 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x0000000000003d90 \_DYNAMIC

.got 0x0000000000003f90 0x70

\*(.got.plt)

.got.plt 0x0000000000003f90 0x48 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x0000000000003f90 GLOBAL\_OFFSET\_TABLE

\*(.igot.plt)

\*(.got)

.got 0x0000000000003fd8 0x28 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

\*(.igot)

0x0000000000004000 . = DATA\_SEGMENT\_RELRO\_END (., 0x0)

.data 0x0000000000004000 0x10

(.data .data. .gnu.linkonce.d.\*)

.data 0x0000000000004000 0x4 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

0x0000000000004000 data\_start

0x0000000000004000 \_\_data\_start

.data 0x0000000000004004 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crti.o

.data 0x0000000000004004 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/crtbeginS.o

fill 0x0000000000004004 0x4

.data.rel.local

0x0000000000004008 0x8 /usr/lib/gcc/x86\_64-linux-gnu/11/crtbeginS.o

0x0000000000004008 \_\_dso\_handle

.data 0x0000000000004010 0x0 /tmp/ccTlOlX6.o

.data 0x0000000000004010 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/crtendS.o

.data 0x0000000000004010 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crtn.o

.tm\_clone\_table

0x0000000000004010 0x0

.tm\_clone\_table

0x0000000000004010 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/crtbeginS.o

.tm\_clone\_table

0x0000000000004010 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/crtendS.o

.data1

\*(.data1)

0x0000000000004010 \_edata = .

[!provide] PROVIDE (edata = .)

0x0000000000004010 . = .

0x0000000000004010 \_\_bss\_start = .

.bss 0x0000000000004010 0x8

\*(.dynbss)

.dynbss 0x0000000000004010 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

(.bss .bss. .gnu.linkonce.b.\*)

.bss 0x0000000000004010 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/Scrt1.o

.bss 0x0000000000004010 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crti.o

.bss 0x0000000000004010 0x1 /usr/lib/gcc/x86\_64-linux-gnu/11/crtbeginS.o

.bss 0x0000000000004011 0x0 /tmp/ccTlOlX6.o

.bss 0x0000000000004011 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/crtendS.o

.bss 0x0000000000004011 0x0 /usr/lib/gcc/x86\_64-linux-gnu/11/../../../x86\_64-linux-gnu/crtn.o

\*(COMMON)

0x0000000000004018 . = ALIGN ((. != 0x0)?0x8:0x1)

fill 0x0000000000004011 0x7

.lbss

\*(.dynlbss)

(.lbss .lbss. .gnu.linkonce.lb.\*)

\*(LARGE\_COMMON)

0x0000000000004018 . = ALIGN (0x8)

0x0000000000004018 . = SEGMENT\_START ("ldata-segment", .)

.lrodata

(.lrodata .lrodata. .gnu.linkonce.lr.\*)

.ldata 0x0000000000006018 0x0

(.ldata .ldata. .gnu.linkonce.l.\*)

0x0000000000006018 . = ALIGN ((. != 0x0)?0x8:0x1)

0x0000000000006018 . = ALIGN (0x8)

0x0000000000004018 \_end = .

[!provide] PROVIDE (end = .)

0x0000000000006018 . = DATA\_SEGMENT\_END (.)

.stab

\*(.stab)

.stabstr

\*(.stabstr)

.stab.excl

\*(.stab.excl)

.stab.exclstr

\*(.stab.exclstr)

.stab.index

\*(.stab.index)

.stab.indexstr

\*(.stab.indexstr)

.comment 0x0000000000000000 0x2b

\*(.comment)

.comment 0x0000000000000000 0x2b /usr/lib/gcc/x86\_64-linux-gnu/11/crtbeginS.o

0x2c (size before relaxing)

.comment 0x000000000000002b 0x2c /tmp/ccTlOlX6.o

.comment 0x000000000000002b 0x2c /usr/lib/gcc/x86\_64-linux-gnu/11/crtendS.o

.gnu.build.attributes

(.gnu.build.attributes .gnu.build.attributes.)

.debug

\*(.debug)

.line

\*(.line)

.debug\_srcinfo

\*(.debug\_srcinfo)

.debug\_sfnames

\*(.debug\_sfnames)

.debug\_aranges

\*(.debug\_aranges)

.debug\_pubnames

\*(.debug\_pubnames)

.debug\_info

(.debug\_info .gnu.linkonce.wi.)

.debug\_abbrev

\*(.debug\_abbrev)

.debug\_line

(.debug\_line .debug\_line. .debug\_line\_end)

.debug\_frame

\*(.debug\_frame)

.debug\_str

\*(.debug\_str)

.debug\_loc

\*(.debug\_loc)

.debug\_macinfo

\*(.debug\_macinfo)

.debug\_weaknames

\*(.debug\_weaknames)

.debug\_funcnames

\*(.debug\_funcnames)

.debug\_typenames

\*(.debug\_typenames)

.debug\_varnames

\*(.debug\_varnames)

.debug\_pubtypes

\*(.debug\_pubtypes)

.debug\_ranges

\*(.debug\_ranges)

.debug\_addr

\*(.debug\_addr)

.debug\_line\_str

\*(.debug\_line\_str)

.debug\_loclists

\*(.debug\_loclists)

.debug\_macro

\*(.debug\_macro)

.debug\_names

\*(.debug\_names)

.debug\_rnglists

\*(.debug\_rnglists)

.debug\_str\_offsets

\*(.debug\_str\_offsets)

.debug\_sup

\*(.debug\_sup)

.gnu.attributes

\*(.gnu.attributes)

/DISCARD/

\*(.note.GNU-stack)

\*(.gnu\_debuglink)

(.gnu.lto\_)

OUTPUT(a.out elf64-x86-64)

Данный файл содержит информацию о дополнительных библиотеках, карту памяти и сценарий компоновщика.

# Дизассемблирование полученного объектного файла

Дизассемблировать объектный файл можно командой *objdump -d* следующим образом:

**$ objdump -d main.o > main\_disasm.s**

Дизассемблированный объектный файл в main\_disasm.s:

|  |
| --- |
| main.o: file format elf64-x86-64  Disassembly of section .text:  0000000000000000 <calculate\_d>:  0: 55 push %rbp  1: 48 89 e5 mov %rsp,%rbp  4: 48 83 ec 30 sub $0x30,%rsp  8: 0f 57 c0 xorps %xmm0,%xmm0  b: f2 0f 11 45 f0 movsd %xmm0,-0x10(%rbp)  10: 0f 57 c0 xorps %xmm0,%xmm0  13: f2 0f 11 45 e8 movsd %xmm0,-0x18(%rbp)  18: c7 45 e4 01 00 00 00 movl $0x1,-0x1c(%rbp)  1f: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 26 <calculate\_d+0x26>  26: 48 8d 75 e8 lea -0x18(%rbp),%rsi  2a: b0 00 mov $0x0,%al  2c: e8 00 00 00 00 call 31 <calculate\_d+0x31>  31: f2 0f 10 05 00 00 00 movsd 0x0(%rip),%xmm0 # 39 <calculate\_d+0x39>  38: 00  39: 66 0f 2e 45 e8 ucomisd -0x18(%rbp),%xmm0  3e: 0f 86 20 00 00 00 jbe 64 <calculate\_d+0x64>  44: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 4b <calculate\_d+0x4b>  4b: b0 00 mov $0x0,%al  4d: e8 00 00 00 00 call 52 <calculate\_d+0x52>  52: f2 0f 10 05 00 00 00 movsd 0x0(%rip),%xmm0 # 5a <calculate\_d+0x5a>  59: 00  5a: f2 0f 11 45 f8 movsd %xmm0,-0x8(%rbp)  5f: e9 b5 00 00 00 jmp 119 <calculate\_d+0x119>  64: e9 00 00 00 00 jmp 69 <calculate\_d+0x69>  69: f2 0f 10 45 e8 movsd -0x18(%rbp),%xmm0  6e: f2 0f 10 0d 00 00 00 movsd 0x0(%rip),%xmm1 # 76 <calculate\_d+0x76>  75: 00  76: 66 0f 2e c1 ucomisd %xmm1,%xmm0  7a: 0f 82 8f 00 00 00 jb 10f <calculate\_d+0x10f>  80: f2 0f 10 05 00 00 00 movsd 0x0(%rip),%xmm0 # 88 <calculate\_d+0x88>  87: 00  88: 66 0f 2e 45 e8 ucomisd -0x18(%rbp),%xmm0  8d: 0f 86 05 00 00 00 jbe 98 <calculate\_d+0x98>  93: e9 77 00 00 00 jmp 10f <calculate\_d+0x10f>  98: f2 0f 10 45 f0 movsd -0x10(%rbp),%xmm0  9d: f2 0f 11 45 d8 movsd %xmm0,-0x28(%rbp)  a2: f2 0f 10 45 e8 movsd -0x18(%rbp),%xmm0  a7: f2 0f 2a 4d e4 cvtsi2sdl -0x1c(%rbp),%xmm1  ac: f2 0f 5e c1 divsd %xmm1,%xmm0  b0: e8 00 00 00 00 call b5 <calculate\_d+0xb5>  b5: 0f 28 c8 movaps %xmm0,%xmm1  b8: f2 0f 10 45 d8 movsd -0x28(%rbp),%xmm0  bd: f2 0f 58 c1 addsd %xmm1,%xmm0  c1: f2 0f 11 45 f0 movsd %xmm0,-0x10(%rbp)  c6: 8b 45 e4 mov -0x1c(%rbp),%eax  c9: 83 c0 01 add $0x1,%eax  cc: 89 45 e4 mov %eax,-0x1c(%rbp)  cf: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # d6 <calculate\_d+0xd6>  d6: 48 8d 75 e8 lea -0x18(%rbp),%rsi  da: b0 00 mov $0x0,%al  dc: e8 00 00 00 00 call e1 <calculate\_d+0xe1>  e1: 83 f8 01 cmp $0x1,%eax  e4: 0f 84 20 00 00 00 je 10a <calculate\_d+0x10a>  ea: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # f1 <calculate\_d+0xf1>  f1: b0 00 mov $0x0,%al  f3: e8 00 00 00 00 call f8 <calculate\_d+0xf8>  f8: f2 0f 10 05 00 00 00 movsd 0x0(%rip),%xmm0 # 100 <calculate\_d+0x100>  ff: 00  100: f2 0f 11 45 f8 movsd %xmm0,-0x8(%rbp)  105: e9 0f 00 00 00 jmp 119 <calculate\_d+0x119>  10a: e9 5a ff ff ff jmp 69 <calculate\_d+0x69>  10f: f2 0f 10 45 f0 movsd -0x10(%rbp),%xmm0  114: f2 0f 11 45 f8 movsd %xmm0,-0x8(%rbp)  119: f2 0f 10 45 f8 movsd -0x8(%rbp),%xmm0  11e: 48 83 c4 30 add $0x30,%rsp  122: 5d pop %rbp  123: c3 ret  124: 66 2e 0f 1f 84 00 00 cs nopw 0x0(%rax,%rax,1)  12b: 00 00 00  12e: 66 90 xchg %ax,%ax  0000000000000130 <main>:  130: 55 push %rbp  131: 48 89 e5 mov %rsp,%rbp  134: 48 83 ec 20 sub $0x20,%rsp  138: c7 45 fc 00 00 00 00 movl $0x0,-0x4(%rbp)  13f: e8 00 00 00 00 call 144 <main+0x14>  144: f2 0f 11 45 f0 movsd %xmm0,-0x10(%rbp)  149: 0f 57 c0 xorps %xmm0,%xmm0  14c: 66 0f 2e 45 f0 ucomisd -0x10(%rbp),%xmm0  151: 0f 86 0c 00 00 00 jbe 163 <main+0x33>  157: c7 45 fc 01 00 00 00 movl $0x1,-0x4(%rbp)  15e: e9 29 00 00 00 jmp 18c <main+0x5c>  163: f2 0f 10 45 f0 movsd -0x10(%rbp),%xmm0  168: e8 00 00 00 00 call 16d <main+0x3d>  16d: f2 0f 11 45 e8 movsd %xmm0,-0x18(%rbp)  172: f2 0f 10 45 e8 movsd -0x18(%rbp),%xmm0  177: 48 8d 3d 00 00 00 00 lea 0x0(%rip),%rdi # 17e <main+0x4e>  17e: b0 01 mov $0x1,%al  180: e8 00 00 00 00 call 185 <main+0x55>  185: c7 45 fc 00 00 00 00 movl $0x0,-0x4(%rbp)  18c: 8b 45 fc mov -0x4(%rbp),%eax  18f: 48 83 c4 20 add $0x20,%rsp  193: 5d pop %rbp  194: c3 ret |

Отличие дизассемблированного файла от ассемблерного заключается в следующем:

* Рядом с командами находится их машинная «версия»
* Числа заменены на их 16сс аналог
* Метки заменены цифрами

# Глобальные и локальные переменные в исходной программе

Исходный код после добавления глобальной проинициализированной переменной и глобальной неинициализированной переменной:

#include <stdio.h>

#include <math.h>

#define EPS 1e-6

double calculate\_d(void)

{

double d = 0.0;

double x = 0.0;

int n = 1;

scanf("%lf", &x);

if (x < -EPS)

{

printf("Invalid input!\n");

return -1.0;

}

while (x >= -EPS) // Повторяем чтение и вычисление для всех неотрицательных значений x (или не меньше -EPS)

{

if (x < -EPS) // Если входное значение меньше -EPS (почти равное 0)

{

break;

}

d = d + sqrt(x / n); // Вычисляем квадратный корень из x / n и добавляем его к переменной d

n += 1; // Увеличиваем переменную n на единицу

if ((scanf("%lf", &x)) != 1)

{

printf("Invalid input!\n");

return -1.0;

}

}

return d;

}

int main(void)

{

double d = calculate\_d();

if (d < 0.0)

{

return 1;

}

double gx = sin(d);

printf("%lf", gx);

return 0;

}

Узнать таблицу символов можно командой *nm*:

**$ nm main.o**

|  |
| --- |
| dunglasoi@DESKTOP-JG9N5AM:/mnt/d/LabC/LabC/lab\_01\_09\_01$ nm main.o  0000000000000000 r .L.str  0000000000000004 r .L.str.1  0000000000000000 r .LCPI0\_0  0000000000000008 r .LCPI0\_1  U \_\_isoc99\_scanf  0000000000000000 T calculate\_d  0000000000000130 T main  U printf  U sin  U sqrt |

Разделы файла же можно узнать при помощи ключа «-s», о котором мы узнали благодаря ключу «-help», командой *objdump*:

**$ objdump -s main.o**

|  |
| --- |
| dunglasoi@DESKTOP-JG9N5AM:/mnt/d/LabC/LabC/lab\_01\_09\_01$ objdump -s main.o  main.o: file format elf64-x86-64  Contents of section .text:  0000 554889e5 4883ec30 0f57c0f2 0f1145f0 UH..H..0.W....E.  0010 0f57c0f2 0f1145e8 c745e401 00000048 .W....E..E.....H  0020 8d3d0000 0000488d 75e8b000 e8000000 .=....H.u.......  0030 00f20f10 05000000 00660f2e 45e80f86 .........f..E...  0040 20000000 488d3d00 000000b0 00e80000 ...H.=.........  0050 0000f20f 10050000 0000f20f 1145f8e9 .............E..  0060 b5000000 e9000000 00f20f10 45e8f20f ............E...  0070 100d0000 0000660f 2ec10f82 8f000000 ......f.........  0080 f20f1005 00000000 660f2e45 e80f8605 ........f..E....  0090 000000e9 77000000 f20f1045 f0f20f11 ....w......E....  00a0 45d8f20f 1045e8f2 0f2a4de4 f20f5ec1 E....E...\*M...^.  00b0 e8000000 000f28c8 f20f1045 d8f20f58 ......(....E...X  00c0 c1f20f11 45f08b45 e483c001 8945e448 ....E..E.....E.H  00d0 8d3d0000 0000488d 75e8b000 e8000000 .=....H.u.......  00e0 0083f801 0f842000 0000488d 3d000000 ...... ...H.=...  00f0 00b000e8 00000000 f20f1005 00000000 ................  0100 f20f1145 f8e90f00 0000e95a fffffff2 ...E.......Z....  0110 0f1045f0 f20f1145 f8f20f10 45f84883 ..E....E....E.H.  0120 c4305dc3 662e0f1f 84000000 00006690 .0].f.........f.  0130 554889e5 4883ec20 c745fc00 000000e8 UH..H.. .E......  0140 00000000 f20f1145 f00f57c0 660f2e45 .......E..W.f..E  0150 f00f860c 000000c7 45fc0100 0000e929 ........E......)  0160 000000f2 0f1045f0 e8000000 00f20f11 ......E.........  0170 45e8f20f 1045e848 8d3d0000 0000b001 E....E.H.=......  0180 e8000000 00c745fc 00000000 8b45fc48 ......E......E.H  0190 83c4205d c3 .. ].  Contents of section .rodata.cst8:  0000 8dedb5a0 f7c6b0be 00000000 0000f0bf ................  Contents of section .rodata.str1.1:  0000 256c6600 496e7661 6c696420 696e7075 %lf.Invalid inpu  0010 74210a00 t!..  Contents of section .comment:  0000 00556275 6e747520 636c616e 67207665 .Ubuntu clang ve  0010 7273696f 6e203134 2e302e30 2d317562 rsion 14.0.0-1ub  0020 756e7475 3100 untu1.  Contents of section .eh\_frame:  0000 14000000 00000000 017a5200 01781001 .........zR..x..  0010 1b0c0708 90010000 1c000000 1c000000 ................  0020 00000000 24010000 00410e10 8602430d ....$....A....C.  0030 06031f01 0c070800 1c000000 3c000000 ............<...  0040 00000000 65000000 00410e10 8602430d ....e....A....C.  0050 0602600c 07080000 ..`.....  Contents of section .llvm\_addrsig:  0000 0708090a 0c |

Секции переменных и функций можно узнать при помощи ключа «-t», о котором мы вновь узнали благодаря ключу «-help», командой *objdump*:

|  |
| --- |
| $ objdump -t main.o  dunglasoi@DESKTOP-JG9N5AM:/mnt/d/LabC/LabC/lab\_01\_09\_01$ objdump -t main.o  main.o: file format elf64-x86-64  SYMBOL TABLE:  0000000000000000 l df ABS 0000000000000000 main.c  0000000000000000 l d .text 0000000000000000 .text  0000000000000000 l .rodata.cst8 0000000000000000 .LCPI0\_0  0000000000000008 l .rodata.cst8 0000000000000000 .LCPI0\_1  0000000000000000 l O .rodata.str1.1 0000000000000004 .L.str  0000000000000004 l O .rodata.str1.1 0000000000000010 .L.str.1  0000000000000000 g F .text 0000000000000124 calculate\_d  0000000000000000 UND 0000000000000000 \_\_isoc99\_scanf  0000000000000000 UND 0000000000000000 printf  0000000000000000 UND 0000000000000000 sqrt  0000000000000130 g F .text 0000000000000065 main  0000000000000000 UND 0000000000000000 sin |

Смотря на это, мы можем выделить, что:

* Проинициализированные переменные находятся в секции .data
* Неинициализированные переменные находятся в секции .bss

# Отладочная информация

Отладочную информацию можно добавить в файл при помощи ключа «-g» в команде *gcc*. Причём, количество информации – её подробность – можно указать цифрами от 1 до 3. Для наглядности используем ключ «-g3»:

**$ gcc -std=c99 -c main.c -g3 -lm -o main\_info.o**

Таблица символов:

**$ nm main\_info.o**

|  |
| --- |
| dunglasoi@DESKTOP-JG9N5AM:/mnt/d/LabC/LabC/lab\_01\_09\_01$ nm main\_info.o  U \_\_isoc99\_scanf  U \_\_stack\_chk\_fail  0000000000000000 T calculate\_d  0000000000000141 T main  U printf  U puts  U sin  U sqrt  0000000000000000 n wm4.0.b4d2b1ad100d530dc0d7c6ab2cf83b6f  0000000000000000 n wm4.cdefs.h.20.e2d4c614ade3e111562824957311b695  0000000000000000 n wm4.cdefs.h.616.8d7ca1b9d01e52f5b2c040c19a111f7b  0000000000000000 n wm4.features.h.19.be13bb4b33b2be4d5fdeac670166e1a8  0000000000000000 n wm4.features.h.428.88005812d8659b51326095520b75a16a  0000000000000000 n wm4.floatn.h.20.a55feb25f1f7464b830caad4873a8713  0000000000000000 n wm4.floatncommon.h.34.7e1840d7dfb19e9bdb51aeb077d76637  0000000000000000 n wm4.fplogb.h.23.f264b61801f4cf347bed2d0fad7232d9  0000000000000000 n wm4.libcheaderstart.h.31.045646cfd09d1c615866e08d91c4f364  0000000000000000 n wm4.libcheaderstart.h.37.e7d4b6f4649b40d3e0dce357ae78234f  0000000000000000 n wm4.libmsimddeclstubs.h.34.70d39999a9be1e0e0e3916021c6182d5  0000000000000000 n wm4.math.h.207.154fc91142a9a5fbf7efb31ec64eb6b5  0000000000000000 n wm4.math.h.24.808d6e83a8b3b11b5fa9117392e0d6ca  0000000000000000 n wm4.math.h.276.0ce1639e4fd0f75af1ac8728ad903714  0000000000000000 n wm4.math.h.314.55d9bc10b03e05989ad6400842f5a189  0000000000000000 n wm4.math.h.331.a75427efad95ca361cbcd39e72516aa4  0000000000000000 n wm4.math.h.401.6bdb6458b6d78f00f8450a35891b9a64  0000000000000000 n wm4.math.h.48.5dafc6157c80230c0414aa9de5e109ba  0000000000000000 n wm4.math.h.486.cf1fe4d0a3a4d4a82dd672c7833613b8  0000000000000000 n wm4.stdarg.h.34.3a23a216c0c293b3d2ea2e89281481e6  0000000000000000 n wm4.stdcpredef.h.19.88fdbfd5cf6f83ed579effc3e425f09b  0000000000000000 n wm4.stddef.h.181.c3f52a3b0f4c288bddb5dda1562858e2  0000000000000000 n wm4.stdio.h.147.dbd603e98db8f3e1583090fab2abd54e  0000000000000000 n wm4.stdio.h.24.5c1b97eef3c86b7a2549420f69f4f128  0000000000000000 n wm4.stdio.h.31.e39a94e203ad4e1d978c0fc68ce016ee  0000000000000000 n wm4.stdio.h.93.0122ffbd02ddfe34dfaf44a2e3561c5b  0000000000000000 n wm4.stdio\_lim.h.19.de8a40c165be4f8437982ec2cd6fd8b4  0000000000000000 n wm4.struct\_FILE.h.19.0888ac70396abe1031c03d393554032f  0000000000000000 n wm4.stubs64.h.10.7865f4f7062bab1c535c1f73f43aa9b9  0000000000000000 n wm4.time64.h.24.a8166ae916ec910dab0d8987098d42ee  0000000000000000 n wm4.types.h.109.56eb9ae966b255288cc544f18746a7ff  0000000000000000 n wm4.typesizes.h.24.ccf5919b8e01b553263cf8f4ab1d5fde  0000000000000000 n wm4.wordsize.h.4.baf119258a1e53d8dba67ceac44ab6bc |

Секции переменных и функций:

**$ objdump -t main\_info.o**

|  |
| --- |
| dunglasoi@DESKTOP-JG9N5AM:/mnt/d/LabC/LabC/lab\_01\_09\_01$ objdump -t main\_info.o  main\_info.o: file format elf64-x86-64  SYMBOL TABLE:  0000000000000000 l df ABS 0000000000000000 main.c  0000000000000000 l d .text 0000000000000000 .text  0000000000000000 l d .rodata 0000000000000000 .rodata  0000000000000000 l d .debug\_info 0000000000000000 .debug\_info  0000000000000000 l d .debug\_abbrev 0000000000000000 .debug\_abbrev  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_line 0000000000000000 .debug\_line  0000000000000000 l d .debug\_str 0000000000000000 .debug\_str  0000000000000000 l d .debug\_line\_str 0000000000000000 .debug\_line\_str  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l d .debug\_macro 0000000000000000 .debug\_macro  0000000000000000 l .group 0000000000000000 wm4.0.b4d2b1ad100d530dc0d7c6ab2cf83b6f  0000000000000000 l .group 0000000000000000 wm4.stdcpredef.h.19.88fdbfd5cf6f83ed579effc3e425f09b  0000000000000000 l .group 0000000000000000 wm4.stdio.h.24.5c1b97eef3c86b7a2549420f69f4f128  0000000000000000 l .group 0000000000000000 wm4.features.h.19.be13bb4b33b2be4d5fdeac670166e1a8  0000000000000000 l .group 0000000000000000 wm4.wordsize.h.4.baf119258a1e53d8dba67ceac44ab6bc  0000000000000000 l .group 0000000000000000 wm4.features.h.428.88005812d8659b51326095520b75a16a  0000000000000000 l .group 0000000000000000 wm4.cdefs.h.20.e2d4c614ade3e111562824957311b695  0000000000000000 l .group 0000000000000000 wm4.cdefs.h.616.8d7ca1b9d01e52f5b2c040c19a111f7b  0000000000000000 l .group 0000000000000000 wm4.stubs64.h.10.7865f4f7062bab1c535c1f73f43aa9b9  0000000000000000 l .group 0000000000000000 wm4.libcheaderstart.h.37.e7d4b6f4649b40d3e0dce357ae78234f  0000000000000000 l .group 0000000000000000 wm4.stdio.h.31.e39a94e203ad4e1d978c0fc68ce016ee  0000000000000000 l .group 0000000000000000 wm4.stddef.h.181.c3f52a3b0f4c288bddb5dda1562858e2  0000000000000000 l .group 0000000000000000 wm4.stdarg.h.34.3a23a216c0c293b3d2ea2e89281481e6  0000000000000000 l .group 0000000000000000 wm4.types.h.109.56eb9ae966b255288cc544f18746a7ff  0000000000000000 l .group 0000000000000000 wm4.typesizes.h.24.ccf5919b8e01b553263cf8f4ab1d5fde  0000000000000000 l .group 0000000000000000 wm4.time64.h.24.a8166ae916ec910dab0d8987098d42ee  0000000000000000 l .group 0000000000000000 wm4.struct\_FILE.h.19.0888ac70396abe1031c03d393554032f  0000000000000000 l .group 0000000000000000 wm4.stdio.h.93.0122ffbd02ddfe34dfaf44a2e3561c5b  0000000000000000 l .group 0000000000000000 wm4.stdio\_lim.h.19.de8a40c165be4f8437982ec2cd6fd8b4  0000000000000000 l .group 0000000000000000 wm4.stdio.h.147.dbd603e98db8f3e1583090fab2abd54e  0000000000000000 l .group 0000000000000000 wm4.floatn.h.20.a55feb25f1f7464b830caad4873a8713  0000000000000000 l .group 0000000000000000 wm4.floatncommon.h.34.7e1840d7dfb19e9bdb51aeb077d76637  0000000000000000 l .group 0000000000000000 wm4.math.h.24.808d6e83a8b3b11b5fa9117392e0d6ca  0000000000000000 l .group 0000000000000000 wm4.libcheaderstart.h.31.045646cfd09d1c615866e08d91c4f364  0000000000000000 l .group 0000000000000000 wm4.libmsimddeclstubs.h.34.70d39999a9be1e0e0e3916021c6182d5  0000000000000000 l .group 0000000000000000 wm4.math.h.48.5dafc6157c80230c0414aa9de5e109ba  0000000000000000 l .group 0000000000000000 wm4.fplogb.h.23.f264b61801f4cf347bed2d0fad7232d9  0000000000000000 l .group 0000000000000000 wm4.math.h.207.154fc91142a9a5fbf7efb31ec64eb6b5  0000000000000000 l .group 0000000000000000 wm4.math.h.276.0ce1639e4fd0f75af1ac8728ad903714  0000000000000000 l .group 0000000000000000 wm4.math.h.314.55d9bc10b03e05989ad6400842f5a189  0000000000000000 l .group 0000000000000000 wm4.math.h.331.a75427efad95ca361cbcd39e72516aa4  0000000000000000 l .group 0000000000000000 wm4.math.h.401.6bdb6458b6d78f00f8450a35891b9a64  0000000000000000 l .group 0000000000000000 wm4.math.h.486.cf1fe4d0a3a4d4a82dd672c7833613b8  0000000000000000 g F .text 0000000000000141 calculate\_d  0000000000000000 UND 0000000000000000 \_\_isoc99\_scanf  0000000000000000 UND 0000000000000000 puts  0000000000000000 UND 0000000000000000 sqrt  0000000000000000 UND 0000000000000000 \_\_stack\_chk\_fail  0000000000000141 g F .text 0000000000000067 main  0000000000000000 UND 0000000000000000 sin  0000000000000000 UND 0000000000000000 printf |

В сравнении с предыдущим выводом секций, в новом добавились отладочные символы и отладочные секции.

# Получение исполняемого файла

**$ gcc -o main.exe main.c -lm**

# Контрольные вопросы

(a) Объектные файлы с отладочной информацией обычно имеют больший размер, чем объектные файлы без отладочной информации, так как они содержат дополнительную информацию для отладки. Однако, размер исполняемых файлов с отладочной информацией и без зависит от того, была ли включена отладочная информация при создании исполняемого файла.

(b) Объектный файл обычно имеет больше секций и отладочной информации, чем исполняемый файл, так как он содержит всю необходимую информацию для линковки и создания исполняемого файла, а также для отладки. Исполняемый файл же предназначен только для запуска программы и, как правило, содержит только минимально необходимую информацию.

(c) Обычно расположение функций, глобальных и локальных переменных не меняется при компиляции объектного файла в исполняемый файл. Однако, существуют оптимизации компилятора, которые могут изменять порядок инструкций и расположение переменных в памяти, но это не является стандартной практикой.

# Используемые динамические библиотеки

Узнать используемые динамические библиотеки, что использует исполняемый файл, можно с помощью команды *ldd*:

**$ ldd main.exe**

|  |
| --- |
| dunglasoi@DESKTOP-JG9N5AM:/mnt/d/LabC/LabC/lab\_01\_09\_01$ ldd main.exe  linux-vdso.so.1 (0x00007fffc9a3c000)  libm.so.6 => /lib/x86\_64-linux-gnu/libm.so.6 (0x00007f4c01350000)  libc.so.6 => /lib/x86\_64-linux-gnu/libc.so.6 (0x00007f4c01120000)  /lib64/ld-linux-x86-64.so.2 (0x00007f4c01452000 |

Таким образом можно узнать, что исполняемый файл использует следующие динамические библиотеки:

* linux-vdso.so.1
* libm.so.6
* libc.so.6
* ld-linux-x86-64.so.2