###### МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

###### ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

###### НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

###### Факультет информационных технологий

**Кафедра параллельных вычислений**

ОТЧЕТ

О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

«ВВЕДЕНИЕ В АРХИТЕКТУРУ ARM»

студента 2 курса, 23208 группы

**Веретенникова Андрея Александровича**

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

Д.С. Иванишкин

Новосибирск 2024

**СОДЕРЖАНИЕ**

[ЦЕЛЬ 3](#_Toc18443921)

[ЗАДАНИЕ 3](#_Toc18443922)

[ОПИСАНИЕ РАБОТЫ 4](#_Toc18443923)

[ЗАКЛЮЧЕНИЕ 5](#_Toc18443924)

[Приложение 1 (2, …). *Наименование приложения* 6](#_Toc18443925)

# ЦЕЛЬ

*Познакомиться с программной архитектурой ARM и провести анализ ассемблерного листинга программы для архитектуры ARM.*

# ЗАДАНИЕ

*Вариант №1. Алгоритм вычисления числа Пи с помощью разложения в ряд (ряд Грегори-Лейбница) по формуле Лейбница N первых членов ряда.*

# ОПИСАНИЕ РАБОТЫ

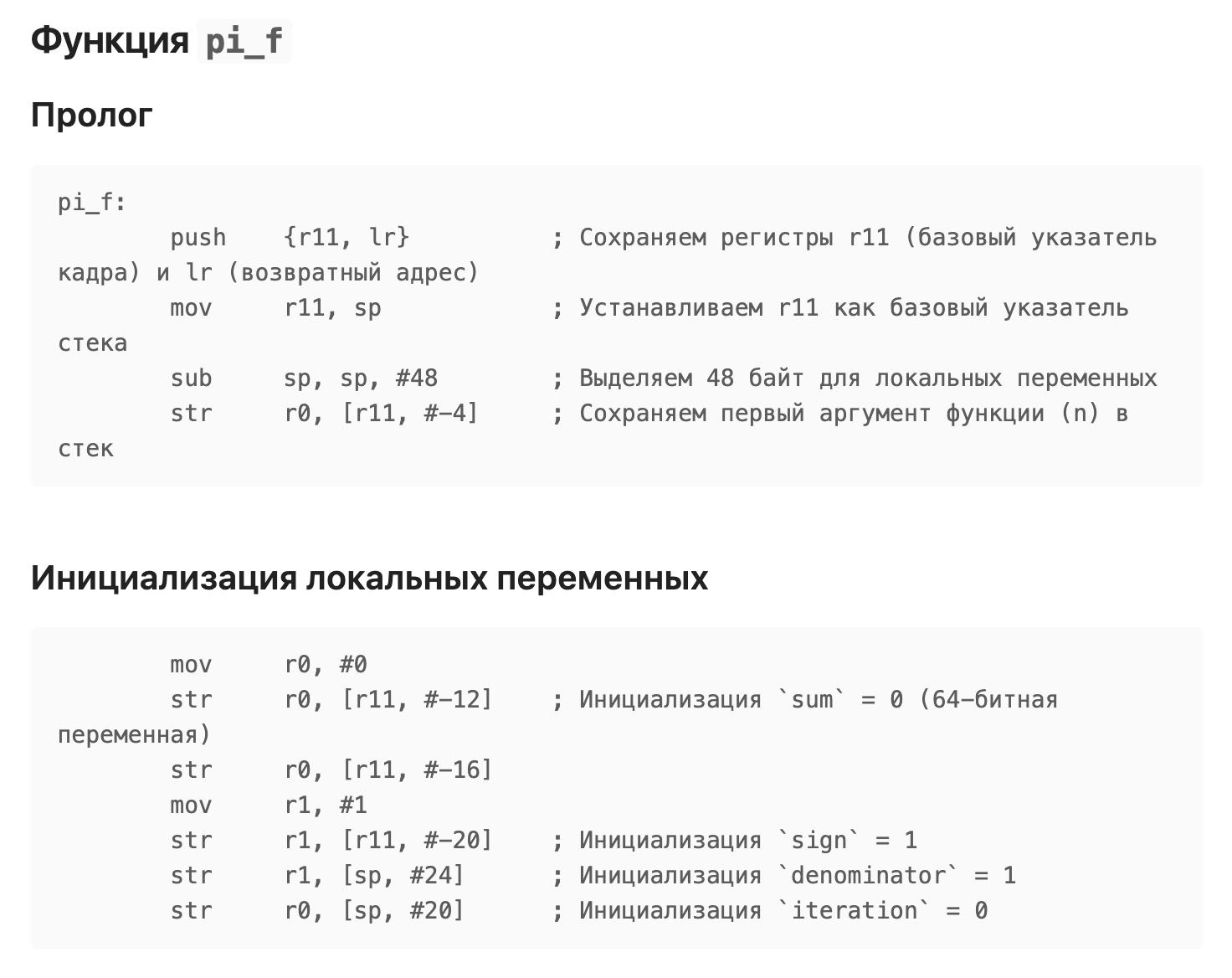
*Пошаговое описание выполненной работы:*

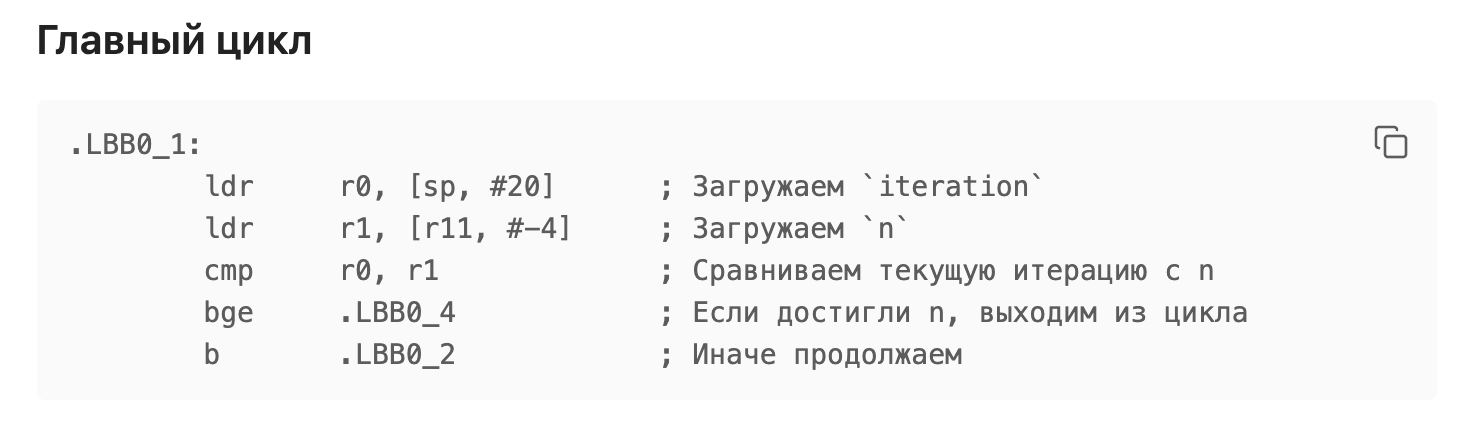
1. *Изучить основы программной архитектуры ARM.*
2. *Для программы из лабораторной работы №1 «ОПРЕДЕЛЕНИЕ ВРЕМЕНИ РАБОТЫ ПРИКЛАДНЫХ ПРОГРАММ» сгенерировать ассемблерные листинги, используя различные уровни комплексной оптимизации.*
3. *Проанализировать полученные листинги.*

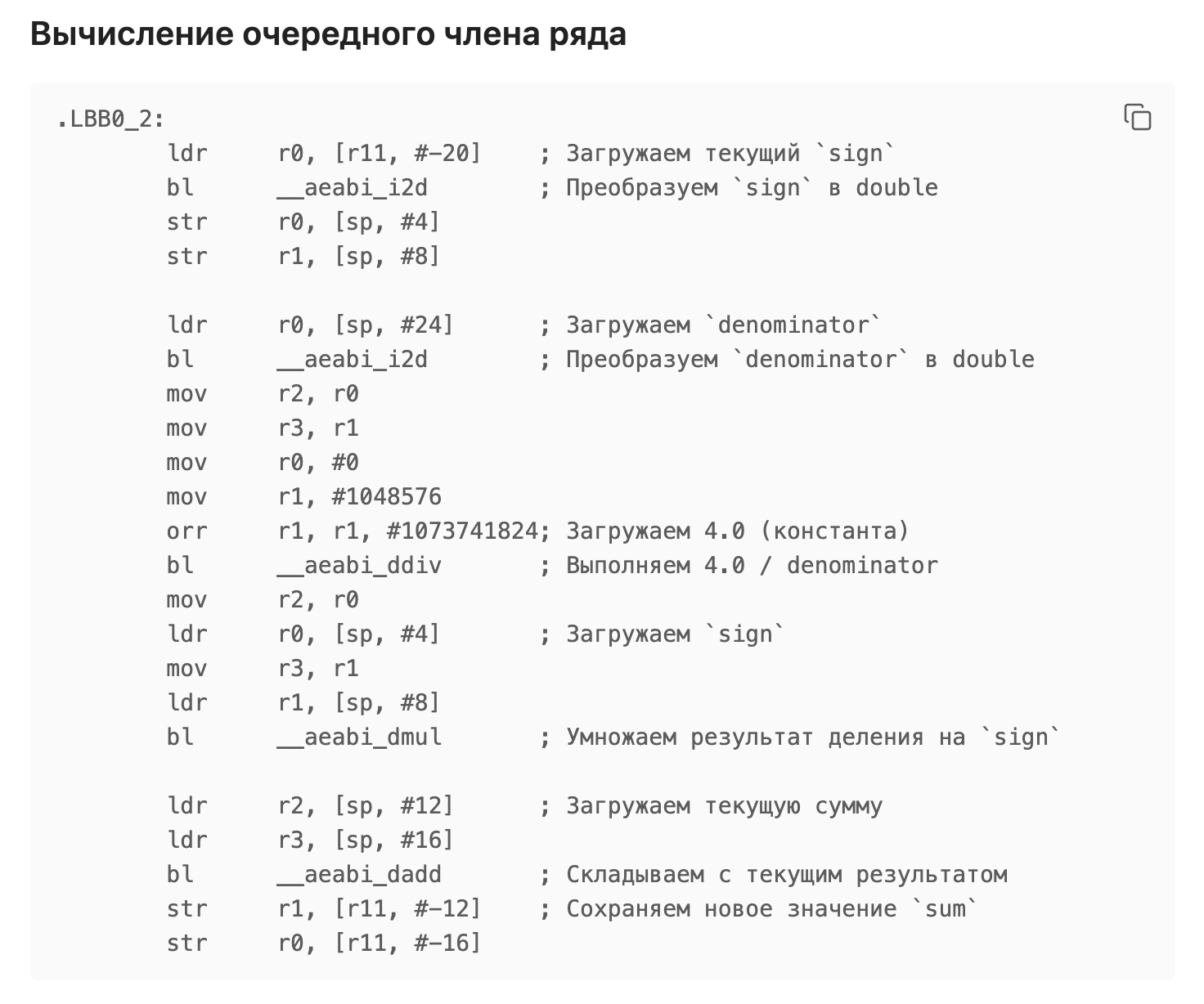
*Выбранная методика: сайт для генерации кода в ассемблерный листинг (Godbolt.org), используемый компилятор armv7-a clang 18.1.0.*

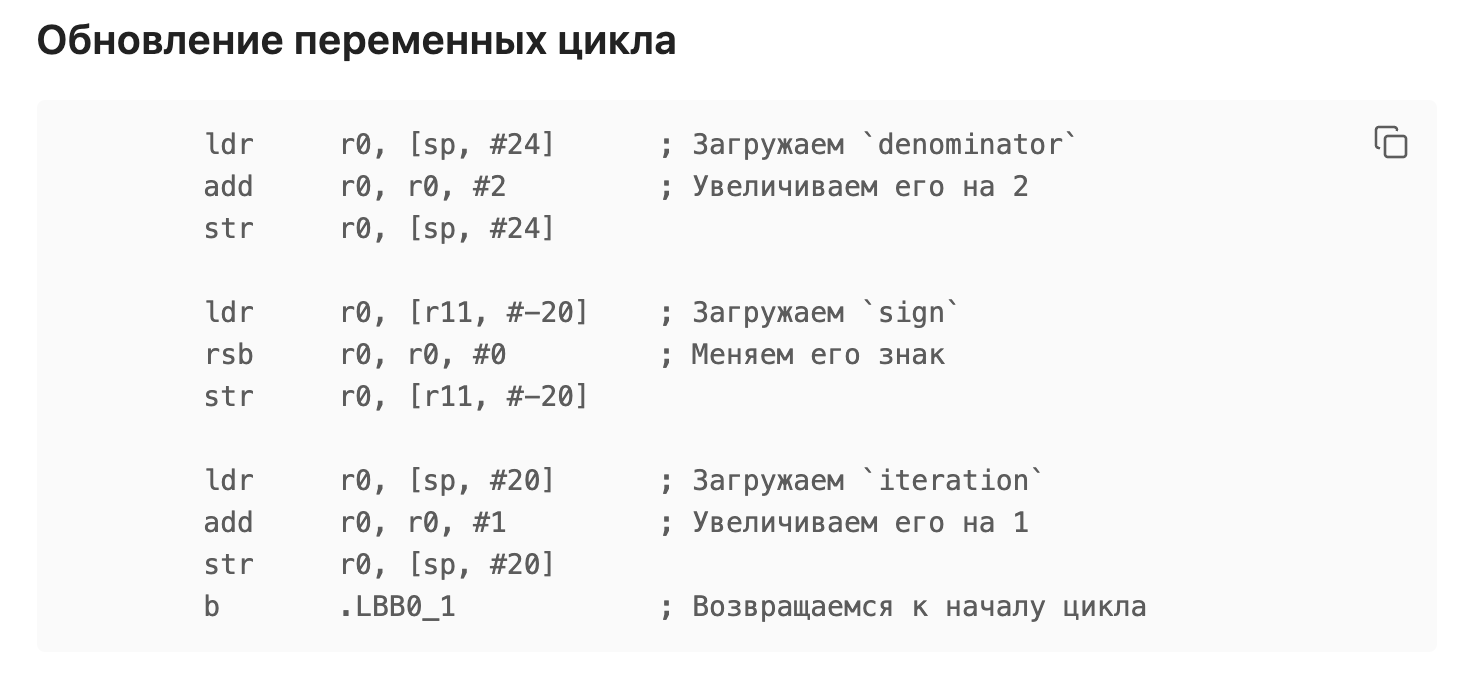
# ХОД РАБОТЫ

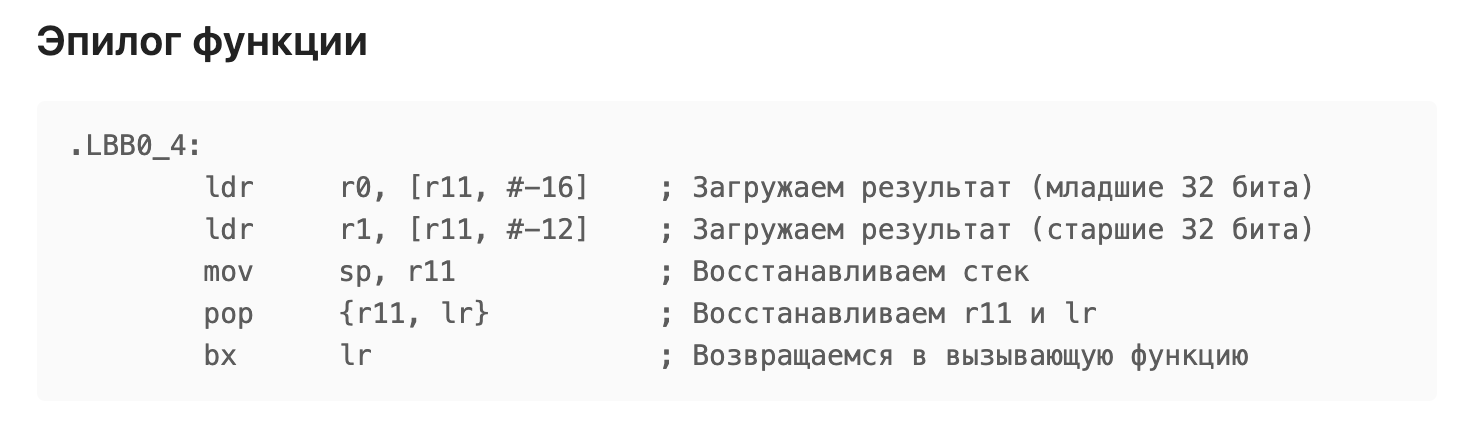
Разбор листинга:

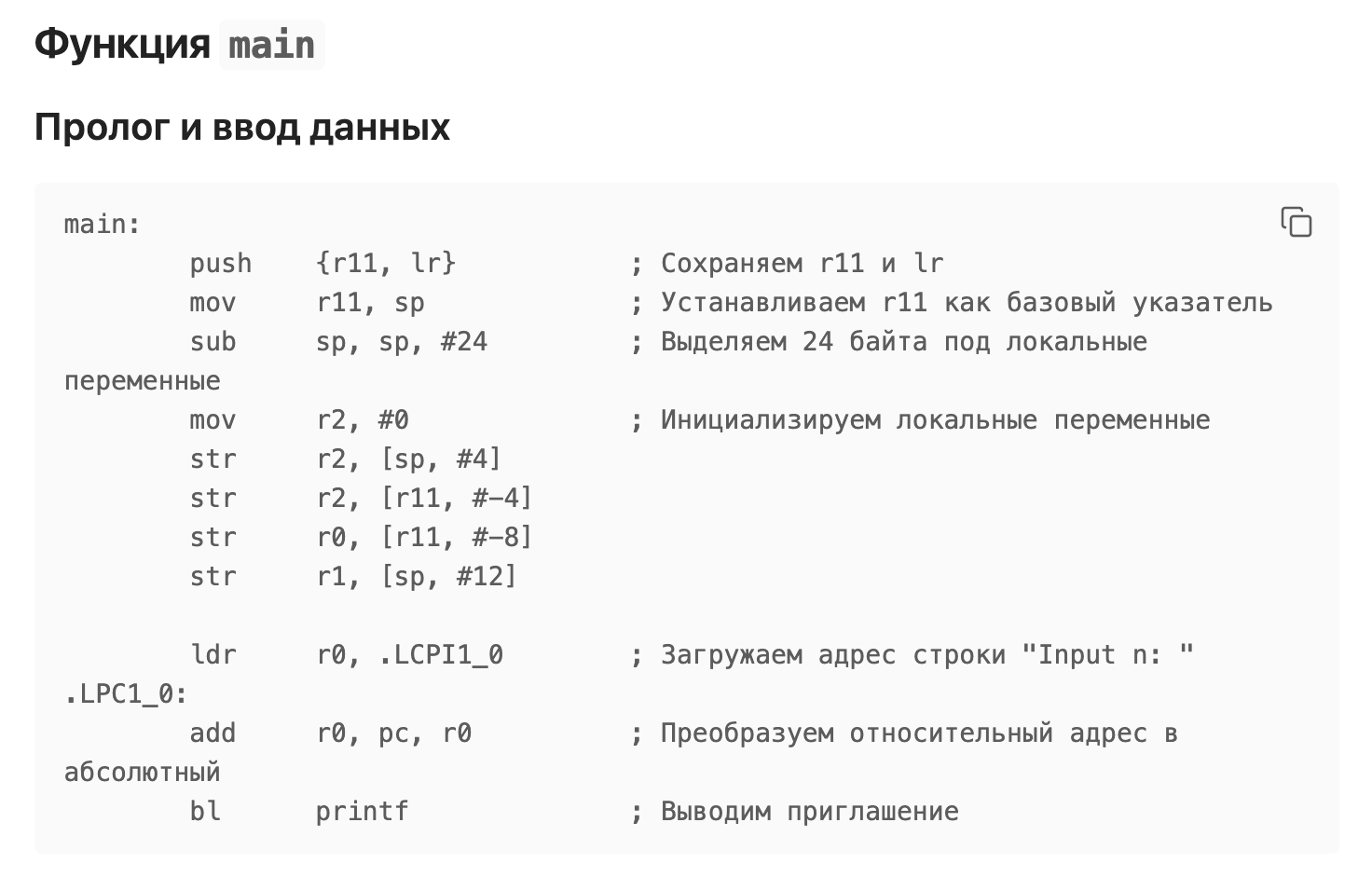


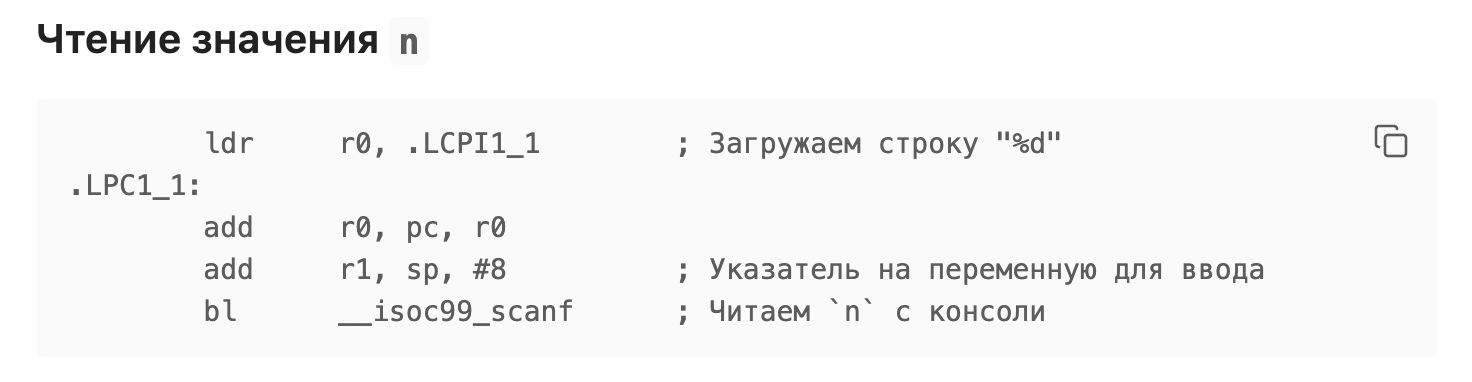


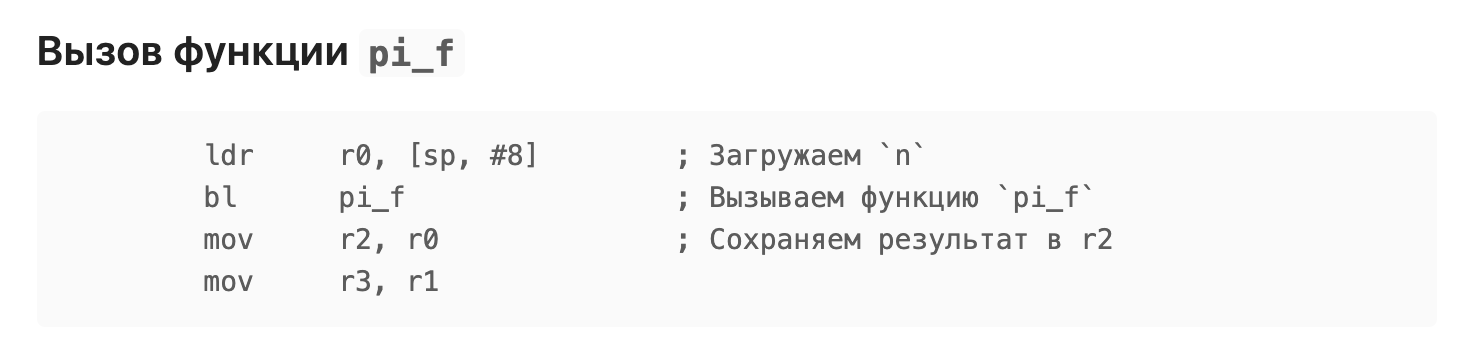


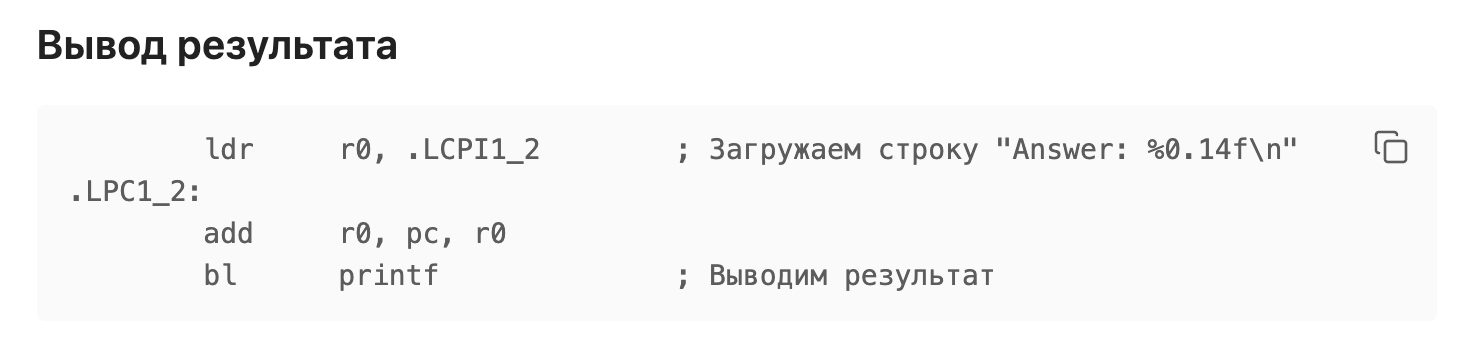


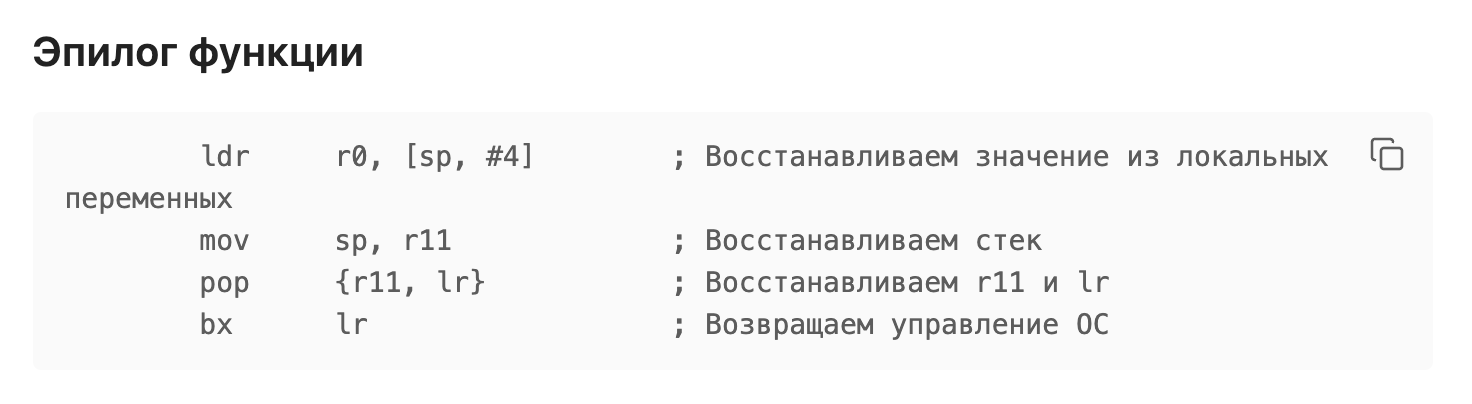


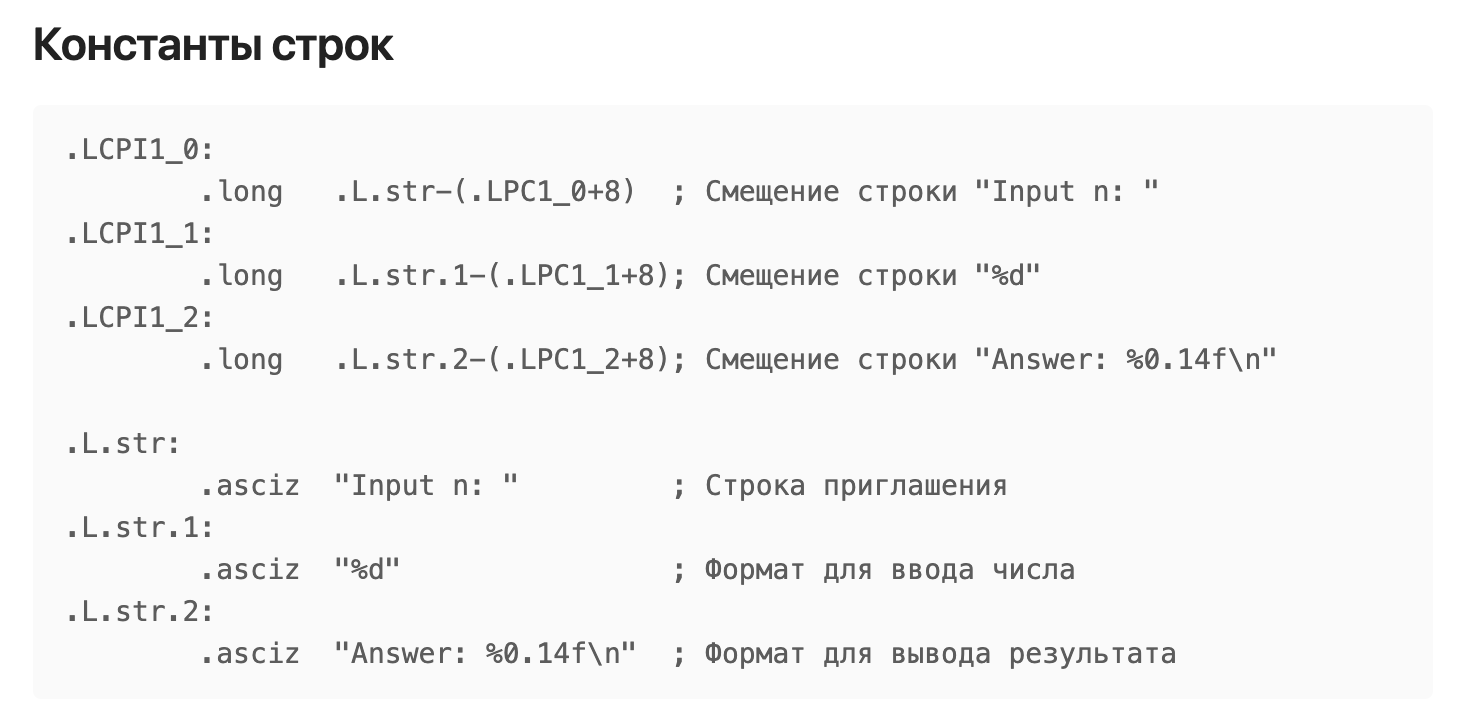












ЗАКЛЮЧЕНИЕ

*В ходе лабораторной работы я познакомился с программной архитектурой ARM и провел анализ ассемблерного листинга программы для этой архитектуры.*

# Приложение 1. *Листинг ARM*

pi\_f:

push {r11, lr}

mov r11, sp

sub sp, sp, #48

str r0, [r11, #-4]

mov r0, #0

str r0, [r11, #-12]

str r0, [r11, #-16]

mov r1, #1

str r1, [r11, #-20]

str r1, [sp, #24]

str r0, [sp, #20]

b .LBB0\_1

.LBB0\_1:

ldr r0, [sp, #20]

ldr r1, [r11, #-4]

cmp r0, r1

bge .LBB0\_4

b .LBB0\_2

.LBB0\_2:

ldr r0, [r11, #-20]

bl \_\_aeabi\_i2d

str r0, [sp, #4]

str r1, [sp, #8]

ldr r0, [sp, #24]

bl \_\_aeabi\_i2d

mov r2, r0

mov r3, r1

mov r0, #0

mov r1, #1048576

orr r1, r1, #1073741824

bl \_\_aeabi\_ddiv

mov r2, r0

ldr r0, [sp, #4]

mov r3, r1

ldr r1, [sp, #8]

ldr r12, [r11, #-16]

str r12, [sp, #12]

ldr r12, [r11, #-12]

str r12, [sp, #16]

bl \_\_aeabi\_dmul

ldr r2, [sp, #12]

ldr r3, [sp, #16]

bl \_\_aeabi\_dadd

str r1, [r11, #-12]

str r0, [r11, #-16]

ldr r0, [sp, #24]

add r0, r0, #2

str r0, [sp, #24]

ldr r0, [r11, #-20]

rsb r0, r0, #0

str r0, [r11, #-20]

b .LBB0\_3

.LBB0\_3:

ldr r0, [sp, #20]

add r0, r0, #1

str r0, [sp, #20]

b .LBB0\_1

.LBB0\_4:

ldr r0, [r11, #-16]

ldr r1, [r11, #-12]

mov sp, r11

pop {r11, lr}

bx lr

main:

push {r11, lr}

mov r11, sp

sub sp, sp, #24

mov r2, #0

str r2, [sp, #4]

str r2, [r11, #-4]

str r0, [r11, #-8]

str r1, [sp, #12]

ldr r0, .LCPI1\_0

.LPC1\_0:

add r0, pc, r0

bl printf

ldr r0, .LCPI1\_1

.LPC1\_1:

add r0, pc, r0

add r1, sp, #8

bl \_\_isoc99\_scanf

ldr r0, [sp, #8]

bl pi\_f

mov r2, r0

mov r3, r1

ldr r0, .LCPI1\_2

.LPC1\_2:

add r0, pc, r0

bl printf

ldr r0, [sp, #4]

mov sp, r11

pop {r11, lr}

bx lr

.LCPI1\_0:

.long .L.str-(.LPC1\_0+8)

.LCPI1\_1:

.long .L.str.1-(.LPC1\_1+8)

.LCPI1\_2:

.long .L.str.2-(.LPC1\_2+8)

.L.str:

.asciz "Input n: "

.L.str.1:

.asciz "%d"

.L.str.2:

.asciz "Answer: %0.14f\n"