Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Архитектура вычислительных систем

**ОТЧЁТ**

к лабораторной работе

на тему

Лабораторная работа №3: Ассемблер RISC-V

Выполнил: студент группы 053506

Осадчий Олег Эдуардович

Проверил: Ст. Преподаватель,

Шиманский Валерий Владимирович

Минск 2022

[**Цели лабораторной работы:**](#_fcodwx50q8wg) **1**

[**Упражнение 1: Подключение ваших файлов к Venus**](#_uo9x8z3vdc9j) **1**

[**Упражнение 2: Знакомство с Venus**](#_bcuthk34u2by) **2**

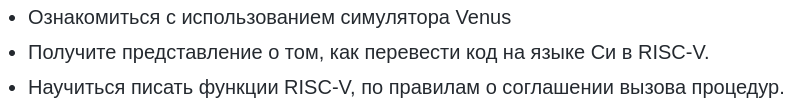
[**Упражнение 3: Перевод с языка C на RISC-V**](#_6oru7kdz7ot3) **4**

[**Упражнение 4: Задание по вариантам.**](#_qk7bpun3jh0r) **6**

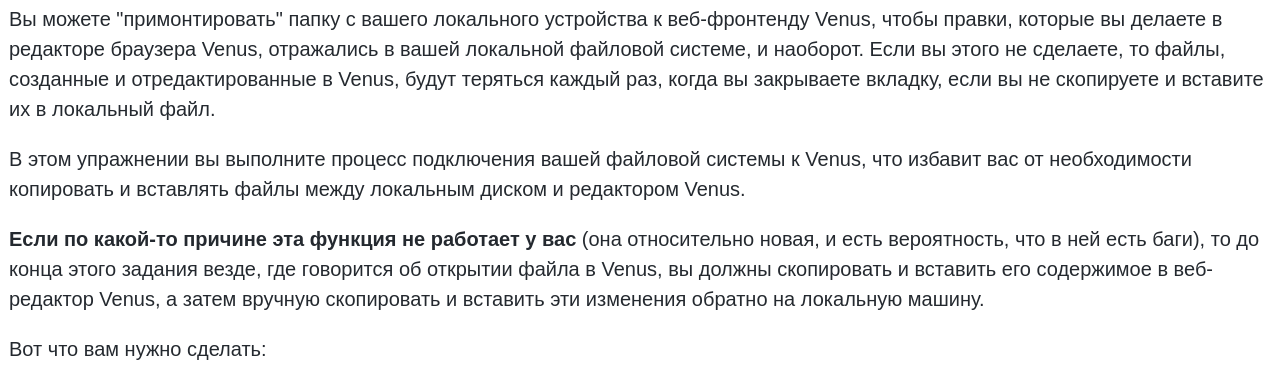
[**Упражнение 5: Практика работы с массивами**](#_bkukukc6jpe6) **8**

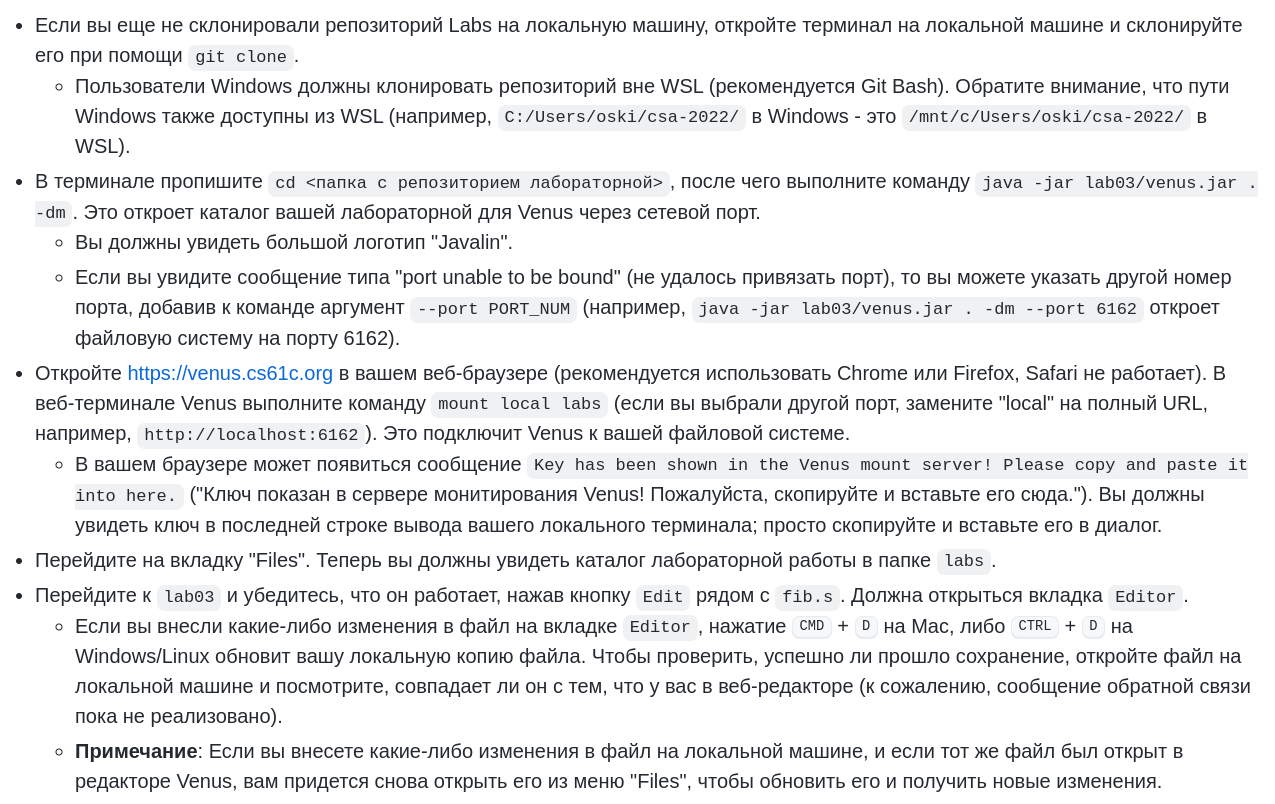
[**Выводы:**](#_hkb6l2bbk985) **9**

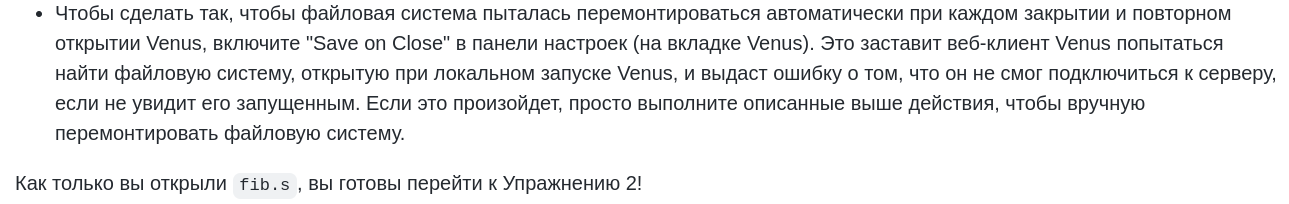
# **Цели лабораторной работы:**



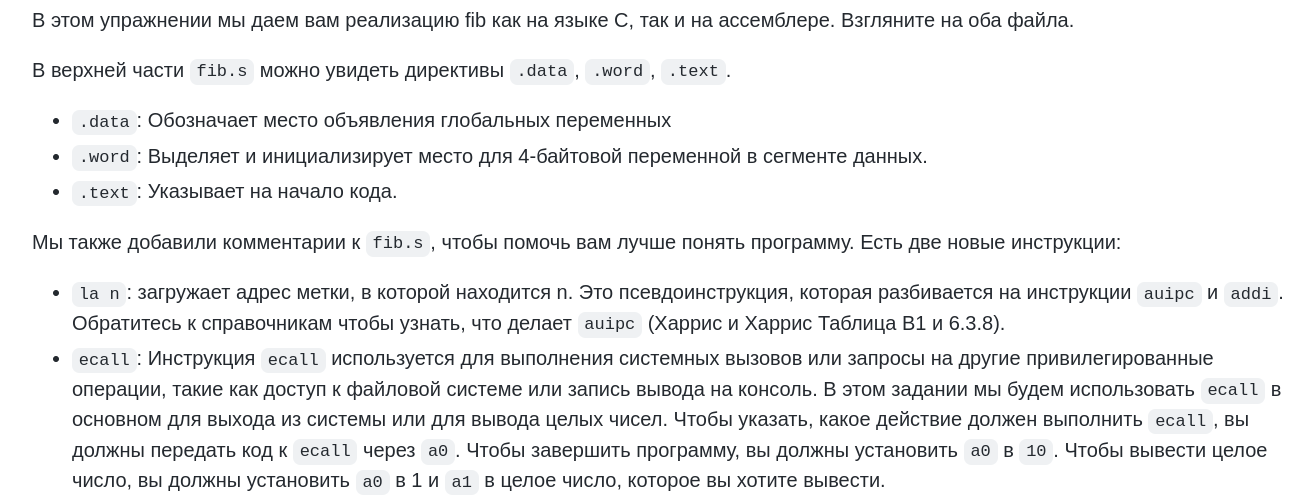
# **Упражнение 1: Подключение ваших файлов к Venus**

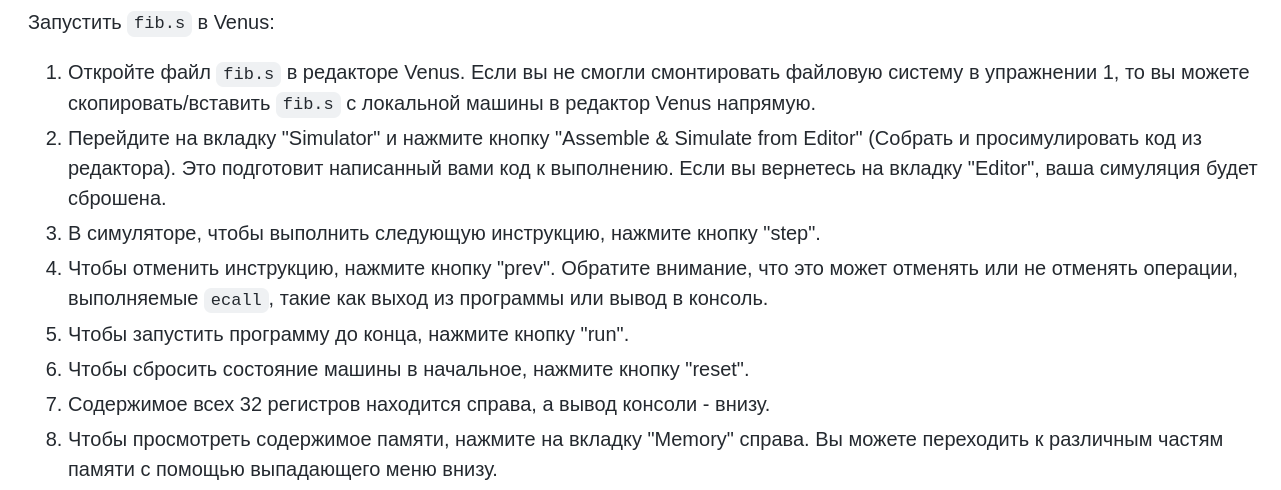


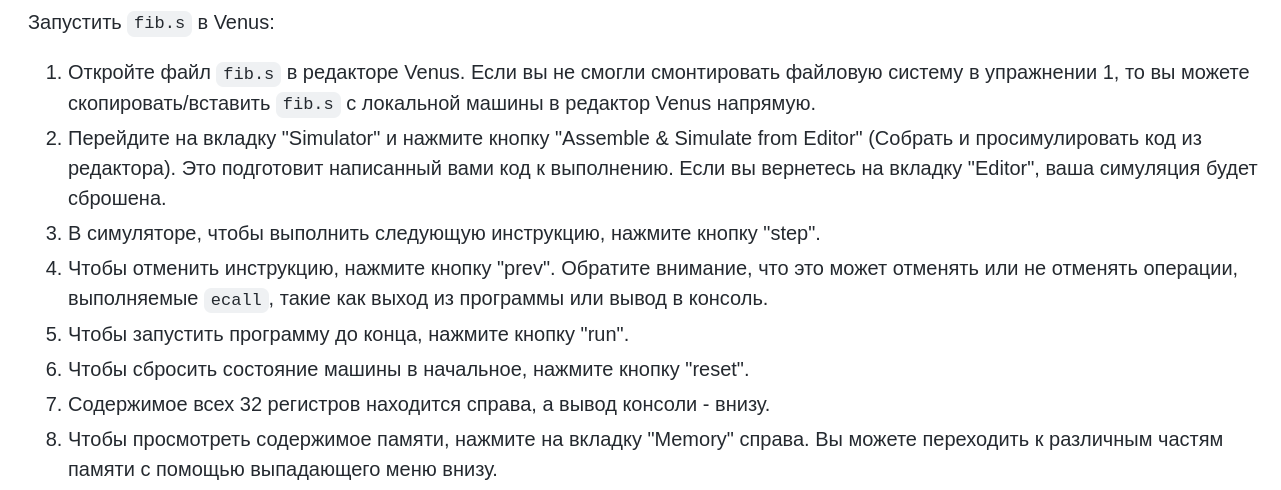


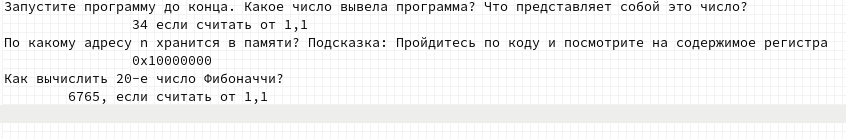


# **Упражнение 2: Знакомство с Venus**











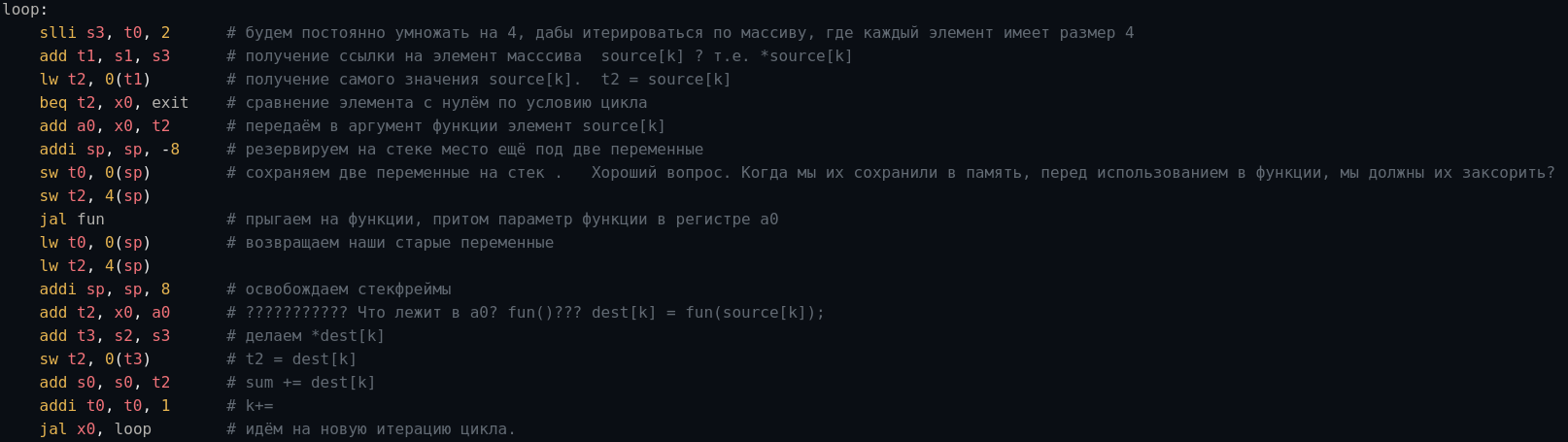
# **Упражнение 3: Перевод с языка C на RISC-V**

# 

1.переменную k представляет регистр t0(x5)

2.переменную sum представляет регистр s0

3.данными регистрами являются s1 , s2



loop:

slli s3, t0, 2 # будем постоянно умножать на 4, дабы итерироваться по массиву, где каждый элемент имеет размер 4

add t1, s1, s3 # получение ссылки на элемент масссива source[k] ? т.е. \*source[k]

lw t2, 0(t1) # получение самого значения source[k]. t2 = source[k]

beq t2, x0, exit # сравнение элемента с нулём по условию цикла

add a0, x0, t2 # передаём в аргумент функции элемент source[k]

addi sp, sp, -8 # резервируем на стеке место ещё под две переменные

sw t0, 0(sp) # сохраняем две переменные на стек . Хороший вопрос. Когда мы их сохранили в память, перед использованием в функции, мы должны их заксорить?

sw t2, 4(sp)

jal fun # прыгаем на функции, притом параметр функции в регистре a0

lw t0, 0(sp) # возвращаем наши старые переменные

lw t2, 4(sp)

addi sp, sp, 8 # освобождаем стекфреймы

add t2, x0, a0 # ??????????? Что лежит в a0? fun()??? dest[k] = fun(source[k]);

add t3, s2, s3 # делаем \*dest[k]

sw t2, 0(t3) # t2 = dest[k]

add s0, s0, t2 # sum += dest[k]

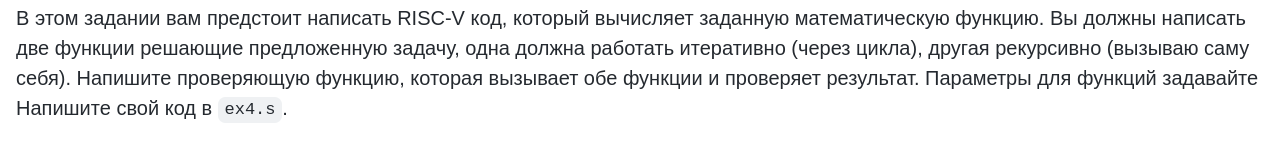
addi t0, t0, 1 # k+=

jal x0, loop # идём на новую итерацию цикла.

5. при помощи конструкций x(register), где x - смещение, а register - память, относительно которой смещаемся, туда мы можем передавать регистры, содержащие адреса памяти.

то есть, сначала мы делаем la s1, source, загружая адрес source в s1 , затем мы итерируемся по адресу, прибавляя нужное смещение add t1, s1, s3 , и только потом получаем само число lw t2, 0(t1)

# **Упражнение 4: Задание по вариантам.**

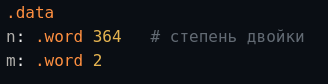


# 

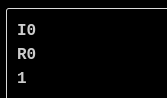
# 

В силу того, что листинг ассемблерного кода выглядит довольно плохо, свой код я прикреплю в документе вместе с отчетом. Здесь же будут находиться лишь выводы с проверкой.

N = 364



Вывод:

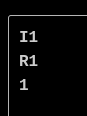


Где, I0 - означает, что итерационная функция вернула 0, то есть, число не является степенью двойки.

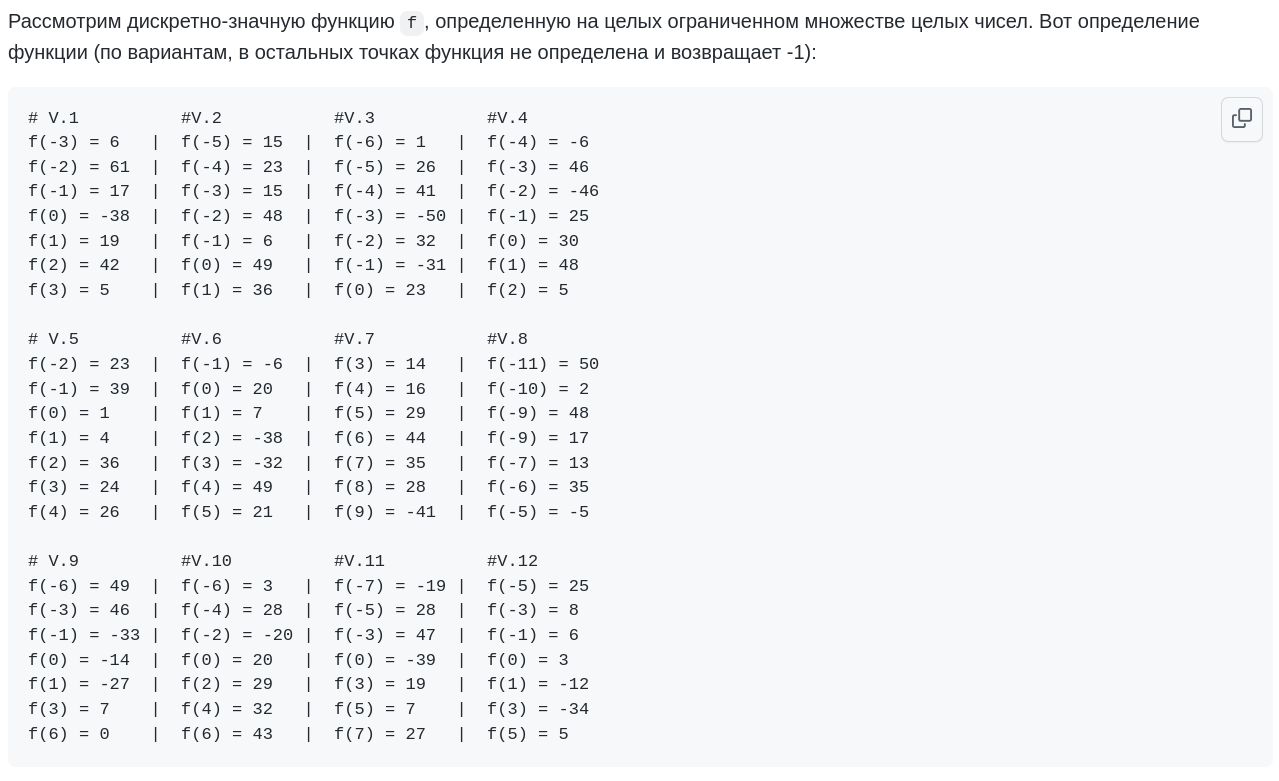
Аналогично и R0.

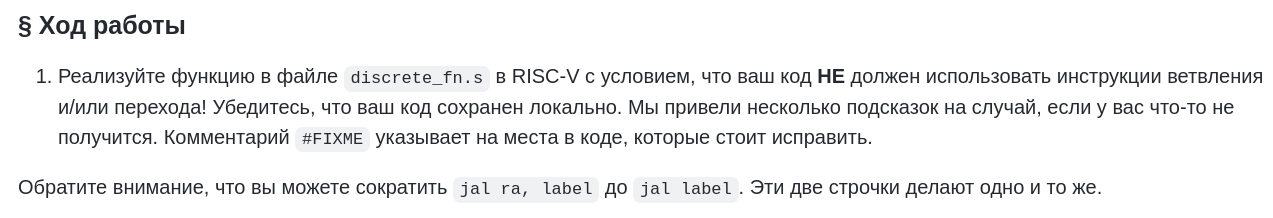
1 в конце означает, что обе функции дали одинаковый результат.

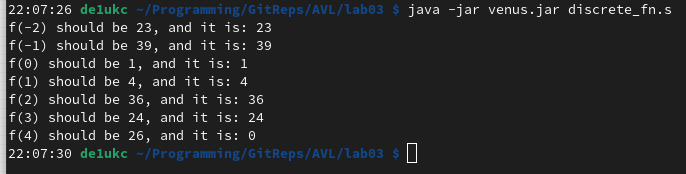
Теперь n = 1024



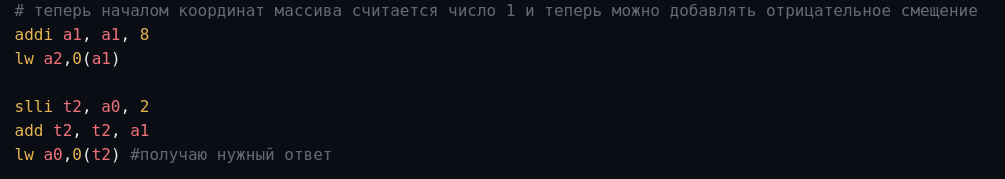
# **Упражнение 5: Практика работы с массивами**







Решение сводится к следующим строчкам:



# **Выводы:**

В данной лабораторной работе, я ознакомился с новым ассемблером, используемым в RISC-V. Я изучил простейшие команды, научился писать и вызывать функции. Так же я научился запускать свои готовые проекты через терминал. Использовать облачный сервер для работы с Venus. Выполнил простейшие задачи на новом для себя ассемблере.

# 