

Санкт-Петербургский политехнический университет Петра Великого  
Высшая школа интеллектуальных систем и суперкомпьютерных технологий

**Отчёт по лабораторной работе № 3**

**Дисциплина:** Низкоуровневое программирование

**Тема:** Программирование RISC-V

Выполнила  
студент гр.3530901/10005

Вилисова Д. Д.

Преподаватель

Коренев Д. А.

“    ” \_\_\_\_\_

Санкт-Петербург

## Оглавление

1. Техническое задание
2. Метод решения
3. Руководство программисту
4. Реализация программы 1
5. Работа программы 1
6. Реализация программы 2 с подпрограммой
7. Работа программы 2

## **1 Техническое задание**

Разработать программу, реализующую сортировку выбором массива чисел in-place.

## **2 Метод решения**

За каждый проход по массиву выбирается минимальный элемент (для сортировки по возрастанию) и происходит обмен его местами с первым элементом в еще не отсортированной части массива, тем самым уменьшив длину этой части на один, и так до тех пор, пока не будут отсортированы все элементы.

Значение длины массива будет размещено в регистре a1, в регистре a2 хранится значение счетчика i, в регистре a5 хранится значение счетчика j, в регистре a3 хранится адрес текущего элемента массива, в a4 хранится адрес последнего элемента отсортированной части массива.

Пример: в результате работы программы (подпрограммы) массив [5, 3, 7, 8, 9, 1, 4, 2] преобразуется в массив [9, 8, 7, 5, 4, 3, 2, 1].

## **3. Руководство программисту**

Начальные данные к программе: адрес нулевого элемента массива (и соответственно сам массив) и его длина. В реализации без подпрограммы адрес хранится в a3 и a4, длина массива хранится в a1. В реализации через подпрограмму предполагается, что аргументами (регистры a2 и a3) передается адрес нулевого элемента массива и аргументом (регистр a0) – длина массива.

## **4. Реализация программы 1**

```

.text
__start:
.globl __start
lw a1, array_length # a1 = <длина массива>
li a2, 1 # i счетчик
la a3, array # <адрес 0 элемента массива>
la a4, array # <адрес 0 элемента массива>

main_loop: bgeu a2, a1, finish # if (i >= array.length) goto finish
           addi a6, a4, 4 # min элемент неотсортированной части
           addi a5, a2, 1 # j счетчик

           loop: bgeu a5, a1, continue_main # if (j >= array.length) goto continue_main
                lw t0, 0(a6) # t0 = array[i + 1], min for loop

                slli a7, a5, 2
                add a7, a3, a7
                lw t1, 0(a7) # t1 = array[j]

                addi a5, a5, 1 # j++
                bgeu t1, t0, loop # if (array[j] >= array[i+1]) goto loop
                sw t1, 0(a6) # array [i + 1] = t1

                sw t0, 0(a7) # array [j] = t0
                jal zero, loop

continue_main: lw t2, 0(a4) # t2 = array[i]
               lw t3, 0(a6) # t3 = array[j], min значение полученное при выполнении loop
               bgeu t3, t2, next # if (t3 >= t2) goto next
               sw t2, 0(a6) # array [j] = t2
               sw t3, 0(a4) # array[i] = t3
next: addi a2, a2, 1 # i++
      addi a4, a4, 4
      jal zero, main_loop

finish: li a0, 10
ecall

.rodata
array_length:
    .word 7

.data
array:
    .word 8, 3, 9, 7, 2, 1, 5

```

## 5. Работа программы 1

Входные данные:

```

.rodata
array_lenght:
    .word 8

.data
array:
    .word 5, 3, 7, 8, 9, 1, 4, 2

```

Элементы массива после выполнения программы:

0x000100a8	00	00	00	09
0x000100a4	00	00	00	08
0x000100a0	00	00	00	07
0x0001009c	00	00	00	05
0x00010098	00	00	00	04
0x00010094	00	00	00	03
0x00010090	00	00	00	02
0x0001008c	00	00	00	01
длина массива				
0x00010088	00	00	00	08

Массив отсортирован.

## 6. Реализация программы 2 с подпрограммой

```

.text
__start:
.globl __start
    addi sp, sp, -16 # выделение памяти в стеке
    sw ra, 12(sp) # сохранение ra

    la a0, array # <адрес 0 элемента массива>
    lw a1, array_length # длина массива

    call selection_sort

    lw ra, 12(sp) # восстановление ra
    addi sp, sp, 16 # освобождение памяти в стеке

    mv a1, a0
    li a0, 17
    ecall

```

```

.text
selection_sort:
.globl selection_sort
li a7, 1 # счетчик i
addi a3, a0, 0

main_loop: bgeu a7, a1, finish # if (i >= array.length) goto finish
            addi a4, a3, 4 # min элемент неотсортированной части
            addi a5, a7, 1 # j счетчик

loop: bgeu a5, a1, continue_main # if (j >= array.length) goto continue_main
      lw t0, 0(a4) # t0 = array[i + 1], min for loop

      slli a6, a5, 2
      add a6, a0, a6
      lw t1, 0(a6) # t1 = array[j]

      addi a5, a5, 1 # j++
      bgeu t1, t0, loop # if (array[j] >= array[i+1]) goto loop
      sw t1, 0(a4) # array [i + 1] = t1
      sw t0, 0(a6) # array [j] = t0
      jal zero, loop

```

```

continue_main: lw t2, 0(a3) # t2 = array[i]
               lw t3, 0(a4) # t3 = array[j], min значение полученное при выполнении loop
               bgeu t3, t2, next # if (t3 >= t2) goto next
               sw t2, 0(a4) # array [j] = t2
               sw t3, 0(a3) # array[i] = t3
next: addi a7, a7, 1 # i++
      addi a3, a3, 4
      jal zero, main_loop

finish:
li a0, 0
ret

```

## 7. Работа программы 2

Входные данные:

```

.rodata
array_lenght:
    .word 9

.data
array:
    .word 5, 3, 7, 8, 9, 1, 4, 2, 6

```

Элементы массива после выполнения программы:

0x000100c4	00	00	00	09
0x000100c0	00	00	00	08
0x000100bc	00	00	00	07
0x000100b8	00	00	00	06
0x000100b4	00	00	00	05
0x000100b0	00	00	00	04
0x000100ac	00	00	00	03
0x000100a8	00	00	00	02
0x000100a4	00	00	00	01
0x000100a0	00	00	00	00

Массив отсортирован.