Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

**Отчёт по лабораторной работе № 2**

Дисциплина: Низкоуровневое программирование

Тема: программирование для EDSAC

Вариант: 2

Выполнил студент гр. 3530901/10003 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.С. Гаранин

(подпись)

Принял преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Коренев Д.А.

(подпись)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г.

Санкт-Петербург

2022

**Оглавление**

1. ТЗ………………………………………………………………………стр. 3
2. Метод решения……………………………………………………….стр. 3
3. Руководство программиста………………………………………….стр. 3
4. Работа программы Orders1…………………………………………стр. 4

**1. ТЗ**

Написать программу для RISC-V, реализующую сортировку вставкой массива чисел in-place. Написать две версии программы, обычную и с подпрограммой.

**2. Метод решения**

Для реализации алгоритма необходимо проходить по массиву, попарно сравнивая текущее число и предыдущее. В случае, когда предыдущее число меньше текущего, эта часть массива уже отсортирована, можно двигаться дальше. Иначе, необходимо двигаться в обратную сторону по обработанной части массива, перебирая предыдущие числа и сравнивая каждое с текущим до тех пор, пока не найдется число меньше текущего. Далее сдвигаем массив после найденного числа и вставляем после него текущее.

Для реализации сдвига массива, я применил следующий способ: в случае, если в результате первой мы получили число больше текущего, мы копируем его в ячейку с индексом + 1. Таким образом, числа будут копироваться в следующие ячейки до тех пор, пока не найдется число меньше текущего. Затем вставляем это число перед ним, массив оказывается сдвинут на 1 индекс вперед, и обработанная часть массива оказывается отсортированной.

**3.Руководство программиста**

Начальные данные к программе: адрес нулевого элемента массива, массив и его длина. В случае с обычной программой длина массива хранится в регистре а3, адрес первого элемента массива – в а4. С подпрограммой: адрес массива в а0, длина в а1.

**4.Реализация программы**

Обычная программа:

.text

start:

.globl start

la a3, array\_length

lw a3, 0(a3) #записали длину массива в a3

la a4, array #адрес массива для основного цикла

li a6, 1 #i = 1

loop1:

bgeu a6, a3, loop\_main\_exit #if i >= array\_length goto loop\_main\_exit

slli s2, a6, 2 #сдвиг

add s2, a4, s2 #a[i+1]

addi s3, s2, -4 #a[i]

li a7, 0 #j = 0

addi a7, a6, 0 #j = i

lw t2, 0(s3) #t2 = a[i]

lw t3, 0(s2) #t3 = a[i+1]

bltu t2, t3, loop1\_exit #if a[i] < a[i+1] goto loop1\_exit

loop2:

bgeu zero, a7, zero\_insert #if 0 >= j goto zero\_insert

slli s4, a7, 2 #сдвиг

add s4, a4, s4 #a[j+1]

#addi s5, s4, -4 #a[j]

lw t4, -4(s4) #a[j]

lw t5, 0(s4) #a[j+1]

sw t4, 0(s4) # a[j+1] = a[j]

bltu t3, t4, loop2\_exit #if savedNum < a[j] goto loop2\_exit

insert:

sw t3, 0(s4) #a[j] = savedNum

bgeu t3, t4 loop2\_exit #if savedNum >= a[j] goto loop2\_exit

zero\_insert:

sw t3, 0(a4) #a[0] = savedNum

loop2\_exit:

bgeu zero, a7, loop1\_exit #if 0 >= j goto loop1\_exit

addi a7, a7, -1 #j--

bltu t3, t4, loop2 #if savedNum < a[j] goto loop2

loop1\_exit:

addi a6, a6, 1 #i++

bltu a6, a3, loop1 # if i < array\_length goto loop1

loop\_main\_exit:

finish:

li a0, 10 #x10 = 10

ecall #stop

.rodata

array\_length:

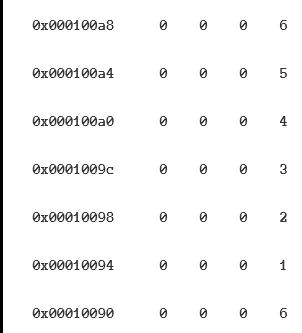
.word 6

.data

array:

.word 5,2,4,1,6,3

Вывод:



Тестовая программа:

# main.s

.text

start:

.globl start

call main

li a0, 10

ecall

main:

.globl main

la a0, array

lw a1, array\_length

addi sp, sp, -16 #выделение памяти в стеке

sw ra, 12(sp) #сохранение ra

call sort

lw ra, 12(sp) #воссстановление ra

addi sp, sp, 16 #освобождение памяти в стеке

ret

.rodata

array\_length:

.word 6

.data

array:

.word 2,5,4,1,6,3

Подпрограмма:

# sort.s

.text

sort:

.globl sort

#a0 = array

#a1 = array\_length

li a2, 1 # i = 1

loop1:

bgeu a2, a1, loop\_main\_exit #if i >= array\_length goto loop\_main\_exit

slli a3, a2, 2 #сдвиг

add a3, a0, a3 #a[i+1]

addi a4, a3, -4 #a[i]

li a5, 0 #j = 0

addi a5, a2, 0 #j = i

lw t2, 0(a4) #t2 = a[i]

lw t3, 0(a3) #t3 = a[i+1]

bltu t2, t3, loop1\_exit #if a[i] < a[i+1] goto loop1\_exit

loop2:

bgeu zero, a5, zero\_insert #if 0 >= j goto zero\_insert

slli a7, a5, 2 #сдвиг

add a7, a0, a7 #a[j+1]

lw t4, -4(a7) #a[j]

lw t5, 0(a7) #a[j+1]

sw t4, 0(a7) # a[j+1] = a[j]

bltu t3, t4, loop2\_exit #if savedNum < a[j] goto loop2\_exit

insert:

sw t3, 0(a7) #a[j] = savedNum

bgeu t3, t4 loop2\_exit #if savedNum >= a[j] goto loop2\_exit

zero\_insert:

sw t3, 0(a0) #a[0] = savedNum

loop2\_exit:

bgeu zero, a5, loop1\_exit #if 0 >= j goto loop1\_exit

addi a5, a5, -1 #j--

bltu t3, t4, loop2 #if savedNum < a[j] goto loop2

loop1\_exit:

addi a2, a2, 1 #i++

bltu a2, a1, loop1 # if i < array\_length goto loop1

loop\_main\_exit:

ret

Вывод:

