

ДЗ 1

а) Скриншоты, демонстрирующие работу эмулятора с программами.

1)

```
add-int01.s
1      li      a7 5          # Системный вызов №5 — ввести десятичное число
2      ecall
3      mv      t0 a0         # Сохраняем результат в t0
4      ecall
5      add     a0 t0 a0       # Прибавляем ко второму число первое
6      li      a7 1          # Системный вызов №1 — вывести десятичное число
7      ecall
8      li      a7 10         # Системный вызов №10 — останов программы
9      ecall
10
```

```
Messages  Run I/O
22
44
66
-- program is finished running (0) --
Clear
```

2)

```
add-int01.s  hello01.s
1      .text
2      la a0, string        # buffer
3      li a7, 4              # syscall write (4)
4      ecall
5      li a0, 0              # exit code
6      li a7, 10             # syscall exit
7      ecall
8      .data
9      string: .asciz "Hello! It works!!!\n"
10
```

```
Messages  Run I/O
Hello! It works!!!
-- program is finished running (0) --
```

3)

The screenshot shows a code editor with tabs for 'add-int01.s', 'hello01.s', 'riscv1.asm', and 'hello02.s'. The 'hello02.s' tab is active, displaying the following assembly code:

```

1      .data
2  hello:
3      .asciz "Hello, world!"
4      .text
5  main:
6      li a7, 4
7      la a0, hello
8      ecall
9

```

Below the code editor, the 'Messages' and 'Run I/O' tabs are visible. The 'Run I/O' tab shows the output of the program:

```

Hello, world!
-- program is finished running (dropped off bottom) --

```

4)

The screenshot shows a code editor with tabs for 'add-int01.s', 'hello01.s', 'riscv1.asm', 'hello02.s', and 'hello03.s'. The 'hello03.s' tab is active, displaying the following assembly code:

```

1      .text
2      la a0, string      # buffer
3      li a7, 4           # syscall write (4)
4      .data
5      string: .asciz "Hello! It works!!!\n"
6      .text
7      ecall
8      li a0, 0           # exit code
9      li a7, 10          # syscall exit
10     ecall
11

```

Below the code editor, the 'Messages' and 'Run I/O' tabs are visible. The 'Run I/O' tab shows the output of the program:

```

Hello! It works!!!
-- program is finished running (0) --

```

5)

The screenshot shows a code editor with tabs for 'add-int01.s', 'hello01.s', 'riscv1.asm', 'hello02.s', 'hello03.s', and 'hello-ru.s'. The 'hello-ru.s' tab is active, displaying the following assembly code:

```

1      .text
2      la a0, string      # buffer
3      li a7, 4           # syscall write (4)
4      ecall
5      li a0, 0           # exit code
6      li a7, 10          # syscall exit
7      ecall
8      .data
9      string: .asciz "Привет. Русский язык выглядит так!!!\n"
10

```

Below the code editor, the 'Messages' and 'Run I/O' tabs are visible. The 'Run I/O' tab shows the output of the program:

```

Привет. Русский язык выглядит так!!!
-- program is finished running (0) --

```

6)

```

Edit Execute
add-int01.s hello01.s riscv1.asm hello02.s hello03.s hello-ru.s add-int02.s
3      arg02: .asciz "Input 2nd number: "
4      result: .asciz "Result = "
5      ln: .asciz "\n"
6      .text
7      la      a0, arg01      # Подсказка для ввода первого числа
8      li      a7, 4          # Системный вызов №4
9      ecall
10     li      a7 5           # Системный вызов №5 — ввести десятичное число
11     ecall      # Результат — в регистре a0
12     mv      t0 a0          # Сохраняем результат в t0
13
14     la      a0, arg02      # Подсказка для ввода второго числа
15     li      a7, 4          # Системный вызов №4
16     ecall
17     li      a7 5           # Системный вызов №5 — ввести десятичное число
18     ecall      # Результат — в регистре a0
19     mv      t1 a0          # Сохраняем результат в t1
20
21     la      a0, result      # Подсказка для выводимого результата
22     li      a7, 4          # Системный вызов №4
23     ecall
24     add      a0 t0 t1      # Складываем два числа
25     li      a7 1           # Системный вызов №1 — вывести десятичное число
26     ecall
27
28     la      a0, ln          # Перевод строки
29     li      a7, 4          # Системный вызов №4
30     ecall
31
32     li      a7 10          # Системный вызов №10 — останов программы
33     ecall
34

```

```

Messages Run I/O
Input 1st number: 13
Input 2nd number: 34
Result = 47

-- program is finished running (0) --

```

- b) Описание, какие команды являются псевдокомандами,
+ анализ результатов компиляции одной из программ.

Псевдокоманды из программ:

| | | |
|------|-----------------|-----------------|
| • li | addi x17,x0,5 | 1: li a7 5 |
| • mv | add x5,x0,x10 | 3: mv t0 a0 |
| • la | auipc x10,64528 | 7: la a0, hello |
| | addi x10,x10,-4 | |

Анализ результатов компиляции 1-ой программы:

| Basic | Source |
|----------------|--|
| addi x17,x0,5 | 1: li a7 5 # Системный вызов №5... |
| ecall | 2: ecall # Результат — в реги... |
| add x5,x0,x10 | 3: mv t0 a0 # Сохраняем результат... |
| ecall | 4: ecall # Регистр a7 не менял... |
| add x10,x5,x10 | 5: add a0 t0 a0 # Прибавляем ко второ... |
| addi x17,x0,1 | 6: li a7 1 # Системный вызов №1... |
| ecall | 7: ecall |
| addi x17,x0,10 | 8: li a7 10 # Системный вызов №1... |
| ecall | 9: ecall |

- 1) для выполнения команды **li a7 5** вызывается команда **addi x17, x0, 5**
- 2) для выполнения команды **mv t0 a0** вызывается команда **add x5, x0, x10**
- 3) для выполнения команды **li a7 1** вызывается команда **addi x17, x0, 1**
- 4) для выполнения команды **li a7 10** вызывается команда **addi x17, x0, 10**

команды li и mv в данной программе являются псевдокомандами, так как для их выполнения вызываются другие команды

с) Описание типов форматов команд для одной из представленных программ.

Первая программа

| Edit | Execute | |
|------|---------|---|
| | | add-int01.s hello01.s riscv1.asm hello02.s hello03.s hello-ru.s add-int02.s |
| 1 | | li a7 5 # Системный вызов №5 — ввести десятичное число |
| 2 | | ecall # Результат — в регистре a0 |
| 3 | | mv t0 a0 # Сохраняем результат в t0 |
| 4 | | ecall # Регистр a7 не менялся, тот же системный вызов |
| 5 | | add a0 t0 a0 # Прибавляем ко второму число первое |
| 6 | | li a7 1 # Системный вызов №1 — вывести десятичное число |
| 7 | | ecall |
| 8 | | li a7 10 # Системный вызов №10 — останов программы |
| 9 | | ecall |
| 10 | | |

| Basic | Source |
|----------------|---|
| addi x17,x0,5 | 1: li a7 5 # Системный вызов №5 — ввести десятичное число |
| ecall | 2: ecall # Результат — в регистре a0 |
| add x5,x0,x10 | 3: mv t0 a0 # Сохраняем результат в t0 |
| ecall | 4: ecall # Регистр a7 не менялся, тот же системный вызов |
| add x10,x5,x10 | 5: add a0 t0 a0 # Прибавляем ко второму число первое |
| addi x17,x0,1 | 6: li a7 1 # Системный вызов №1 — вывести десятичное число |
| ecall | 7: ecall |
| addi x17,x0,10 | 8: li a7 10 # Системный вызов №10 — останов программы |
| ecall | 9: ecall |

- 1) **addi x17,x0,5** - тип I (immediate). Операнды – регистр-регистр-непосредственное значение.
- 2) **add x5,x0,x10** - тип R (register). В роли операндов используются три регистра – регистр назначения, первый аргумент и второй аргумент.
- 3) **add x10,x5,x10** - тип R (register). В роли операндов используются три регистра – регистр назначения, первый аргумент и второй аргумент.
- 4) **addi x17,x0,1** - тип I (immediate). Операнды – регистр-регистр-непосредственное значение.
- 5) **addi x17,x0,10** - тип I (immediate). Операнды – регистр-регистр-непосредственное значение.

d) Описание, какие системные вызовы используются в изученных программах.

- 1 - вывод целого числа
- 4 - вывод нуль терминированной строки
- 5 - чтение целого числа
- 10 - завершение программы с кодом 0