Lista 12

zadanie 1

```
add x5, x6, x7
addi x5, x5, -5
```

zadanie 2

```
d = a + b + c;
```

Gdzie wartości a, b, c znajdużą się odpowiednio w rejestrach x2, x3, x4, a wynik d jest przychowywany w rejestrze x1.

zadanie 3

```
sub x30, x28, x29
add x31, x10, x30
lb x31, 0(x31)
sb x31, 8(x11)
```

zadanie 4

```
T[i] = T[8] + a;
```

Gdzie wartość ‘i’ znajduje się w rejestrze x6, a wartość ‘a’ w x5.

zadanie 5

opcode = 0x3 oznacza rodzaj LOAD
funct3 = 0x2 - operację lw
rs1 = 27 - rejestr źródłowy x27
rd = 3 - rejestr na wartość x3
imm = 0x4 - offset o 4
Zatem instrukcja to:

```
lw x3, 4(x27)
```

Stosujemy format I zatem kod binarny to:

```
000000000100 11011 010 00011 0000011
```

zadanie 6

Stosujemy format S
opcode = 0b0100011
funct3 = 0b010 (bo rozmiar to 4)
rs1 = 0b11110
rs2 = 0b00101
imm = 0b000000100000
Kod binarny:

```
0000001 00101 11110 010 00000 0100011
```

zadanie 7

opcode = 0110011
Zatem rodzaj to OP, czyli stosujemy format R.
rd = 0b00001 - rejestr x1
funct3 = 0b000 - operacja add
rs1 = 0b00001 - rejestr x1
rs2 = 0b00001 - rejestr x1
funct7 = 0b0000000
Otrzymujemy zatem:

```
add x1, x1, x1
```

zadanie 8

a)

```
slli x7, x5, 4
```

Wartość x7 po powyższej operacji:
szesnastkowo: 0x000aaaa0
binarnie: 0b0000 0000 0000 1010 1010 1010 1010 0000

```
or x7, x7, x6
```

Wartość x6:
szesnastkowo: 0x12345678
binarnie: 0b0001 0010 0011 0100 0101 0110 0111 1000

Wartość x7 po powyższej operacji:
binarnie: 0b0001 0010 0011 1110 1111 1110 1111 1000
szesnastkowo: 0x123efef8

b)

```
srli x7, x5, 3
```

Wartość x7 po powyższej operacji:
binarnie: 0b0000 0000 0000 0000 0001 0101 0101 0101

```
andi x7, x7 ,0xfef
```

Wartość imm:
szesnastkowo: 0xfef
binarnie: 0b1111 1110 1111
uzupełnione do 32 bitów: 0b1111 1111 1111 1111 1111 1111 1110 1111

Wartość x7 po powyższej operacji:
binarnie: 0b0000 0000 0000 0000 0001 0101 0100 0101
szesnastkowo 0x00001545