# HW10/Laypanov

## №5

. Assume that `x11` is initialized to `11` and `x12` is initialized to `22` . Suppose you executed the code below on a version of the pipeline that does not handle data hazards. What would the final values of register `x15` be? Assume the register file is written at the beginning of the cycle and read at the end of a cycle. Therefore, an `ID` stage will return the results of a `WB` state occurring during the same cycle.

```
addi x11, x12, 5
add  x13, x11, x12
addi x14, x11, 15
add  x15, x11, x11
```

So, referring to task we have that our pipeline do not handle data hazards and register file is written at the beginning of the cycle and is read at the end of a cycle

So, when **add x13, x11, 5** gets to **EX** stage, the **addi x11, x12, 5** is on **MEM** stage and the **new value of x11 is not yet saved to register**

same goes to **addi x14, x11, 15**: when this operation is on **EX** stage, the **addi x11, x12, 5 is on WB stage** and the value will update on next cycle, so, **x14** will be computed with the initial value of **x11**

But for our **x15** value of **x11** will be already updated so **x15** will be computed as it should be

**x11 = 11**
**x12 = 22**
**addi x11, x12, 5** → x11 = 22 + 5 = 27
**add x13, x11, x12** → x13 = 11 + 22 = 33
**addi x14, x11, 15** → x14 = 11 + 15 = 26
**add x15, x11, x11** → x15 = 27 + 27 = 54

Answer, x15 = 54

№6

Add NOP instructions to the code below so that it will run correctly on a pipeline that does not handle data hazards.

```
addi x11, x12, 5
add  x13, x11, x12
addi x14, x11, 15
add  x15, x13, x12
```

Generally, our aim to fix hazard is to make code save new value of **x11** before execution of add operations of **x13** and **x14** - to be more precise, **add x13, x11, x12** should be on stage **EX** after **addi x11, x12, 5** passes through stage **WB**

Same goes to **addi x15, x13, x12** as this operation should be on stage **EX** after **add x13, x11, x12** passes through stage **WB**

So, solution is following:

```
1 addi x11, x12, 5
2 nop
3 nop
4 add  x13, x11, x12
5 nop
6 addi x14, x11, 15
7 add  x15, x13, x12
```

so, as u can see in picture 2, add x13, x11, x12 is on EX stage and addi x11, x12, 5 is already passed through WB stage
same goes to add x15, x13, x12(EX stage) and x13, x11, x12(passed through WB) in 3 picture