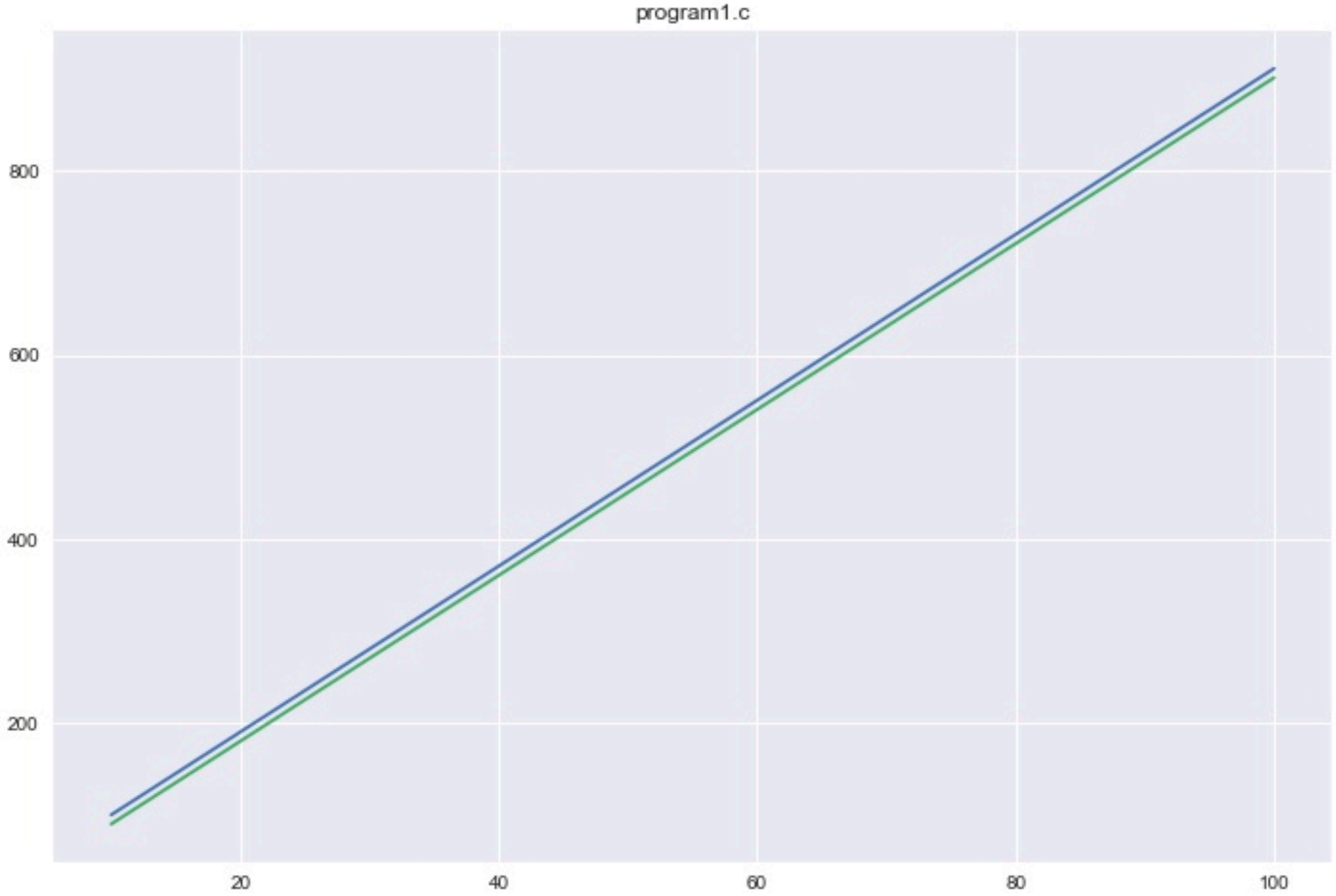
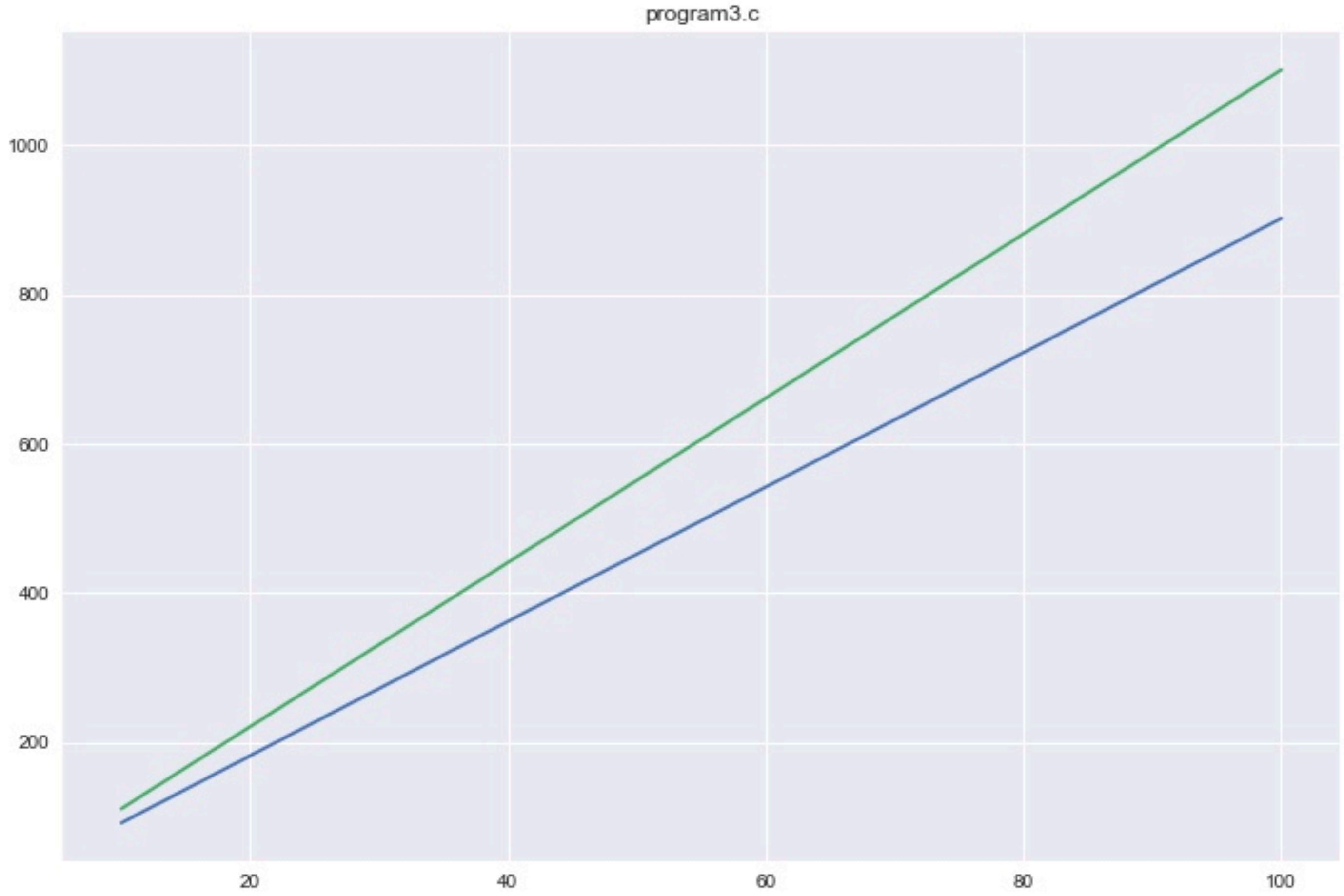
# Project 1B

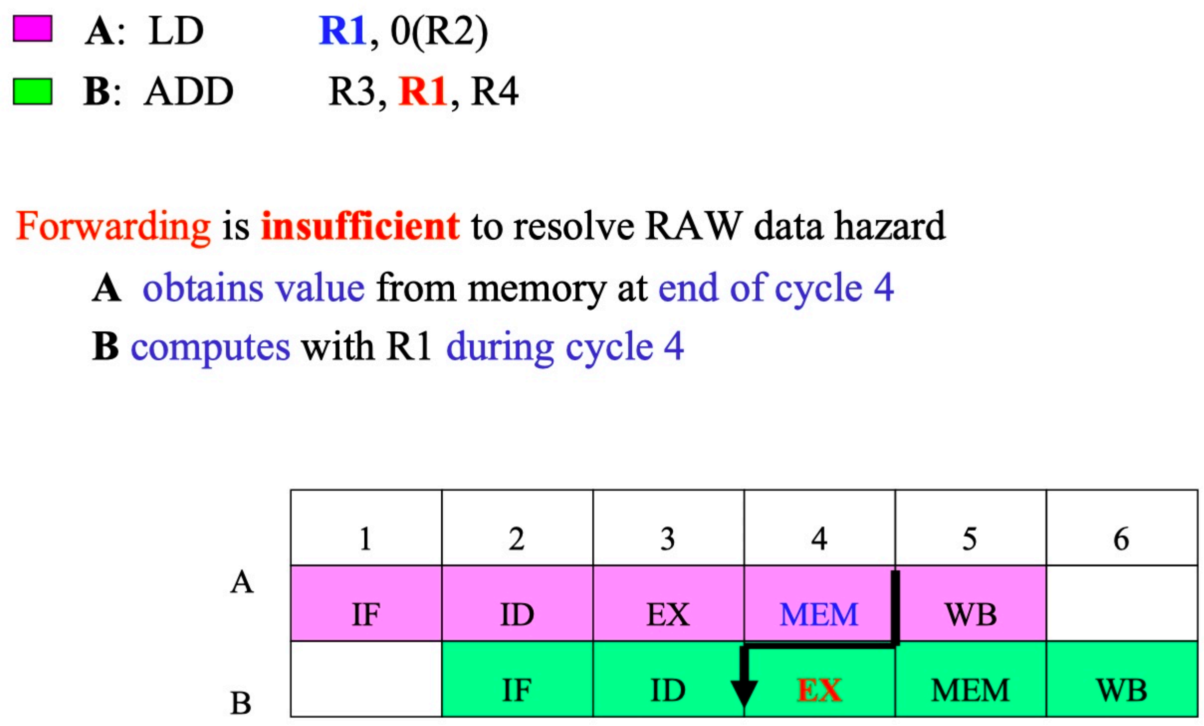
## e. Run each program for NUM_ITERATIONS equal to 10, 20, 40, 80 …





## f. briefly explain your implementation of forwarding, and why you believe that the results in the plots are correct.

```c
// issue.c
checkDataHazard()
{
  if (PR[1].opCode == LOAD && (PR[1].destReg == PR[0].srcReg1 || PR[1].destReg == PR[0].srcReg2)) {
    stallIF = TRUE;
  }else
  {
    stallIF = FALSE;
  }
}
```



🟪 **A**: LD R1, 0(R2)

🟩 **B**: ADD R3, R1, R4

Forwarding is **insufficient** to resolve RAW data hazard

**A** obtains value from memory at end of cycle 4

**B** computes with R1 during cycle 4

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| A | IF | ID | EX | MEM | WB | |
| B | | IF | ID | EX | MEM | WB |

We need to stall when necessary, so if the previous operation is load, i.e. A. And it's result is one of the oprand. We need to stall it.

```
// exec.c
forwardEX_MEMOp1 = PR[3].destReg == PR[1].srcReg1 && PR[3].INSTRUCTION != 0;
forwardEX_MEMOp2 = PR[3].destReg == PR[1].srcReg2 && PR[3].INSTRUCTION != 0;
forwardMEM_WBOp1 = PR[4].destReg == PR[1].srcReg1 && PR[4].INSTRUCTION != 0;
forwardMEM_WBOp2 = PR[4].destReg == PR[1].srcReg1 && PR[4].INSTRUCTION != 0;
```

When the operand is in PR[3] or PR[4]'s results. We need to forward them to the current ID to get the right value.