Problem 1

In this problem, you will port code to a simple **3-issue VLIW machine,** and schedule it to improve performance.

Details about the 3-issue VLIW machine with 3 fully pipelined functional units:

- Integer ALU with 1 cycle latency to next Integer/FP
- Integer ALU with 2 cycle latency to next Branch
- Memory Unit with 3 cycle latency
- Floating Point Unit with 3 cycle latency (it can complete one add or one multiply per clock cycle)
- Branch completed with 1 cycle delay slot (branch solved in ID stage)
- In the Register File, it is possible to read and write at the same address at the same clock cycle

C Code:

```
for(int i=0; i<N; i++)
        C[i] = A[i]*A[i] + B[i];
```

Assembly Code:

```
loop: ld   f1, 0(r1)
      ld   f2, 0(r2)
      fmul f1, f1, f1
      fadd f1, f1, f2
      st   f1, 0(r3)
      addi r1, r1, 4
      addi r2, r2, 4
      addi r3, r3, 4
      bne  r3, r4, loop
```

Problem 1.A

Considering **one iteration** of the loop (except for the last one), **schedule** the assembly code for the 3-issue VLIW machine in the following table.

Schedule the assembly code by using the **list-based scheduling,** but do not use any software pipelining or loop unrolling. You do not need to write in NOPs (can leave blank).

	Integer ALU	Memory Unit	FPU
C0			
C1			
C2			
C3			
C4			
C5			
C6			
C7			
C8			
C9			
C10			
C11			
C12			

How long is the critical path?

What performance did you achieve in FP ops per cycle?

What performance did you achieve in cycles per loop iteration?

What code efficiency did you achieve?

What loop overhead did you achieve?

Problem 1.B

Based on the solution obtained in Problem 1.A, **reschedule to further optimize** the assembly code for the 3-issue VLIW machine in the following table.

	Integer ALU	Memory Unit	FPU
C0			
C1			
C2			
C3			
C4			
C5			
C6			
C7			
C8			
C9			
C10			
C11			

How long is the Critical Path?
What performance did you achieve in FP ops per cycle?
What performance did you achieve in cycles per loop iteration?
What code efficiency did you achieve?
What loop overhead did you achieve?

_____

Problem 1.C

Unroll two iterations of the loop (so two iterations of the original loop are done for each branch in the new assembly code)

Unrolled Assembly Code (no scheduled)

```
loop:ld   f1, 0(r1)
```

Considering **one iteration of the unrolled loop, schedule** the assembly code for the 3-issue VLIW machine in the following table **by using list-based scheduling:**

	Integer ALU	Memory Unit	FPU
C0			
C1			
C2			
C3			
C4			
C5			
C6			
C7			
C8			
C9			
C10			
C11			
C12			
C13			

How long is the Critical Path?
What performance did you achieve in FP ops per cycle?
What performance did you achieve in cycles per loop iteration?
What code efficiency did you achieve?
What loop overhead did you achieve?

Problem 1.D

Based on the solution obtained in Problem 1.A, **reschedule to further optimize** the assembly code for the 3-issue VLIW machine in the following table.

	Integer ALU	Memory Unit	FPU
C0			
C1			
C2			
C3			
C4			
C5			
C6			
C7			
C8			
C9			
C10			
C11			
C12			

How long is the Critical Path?
What performance did you achieve in FP ops per cycle?
What performance did you achieve in cycles per loop iteration?
What code efficiency did you achieve?
What loop overhead did you achieve?