Homework 2

Problem 1

Consider the following function with potential load-store interactions:

```
void foo (int *src, int *dest, int n)
{
    int i;
    for ( i = 0; i < n; i++ )
       dest[i] = src[i] + i;
}
```

Performance of arithmetic operations is list below:

Instruction	Latency	Cycles/Issue
Load / Store (Cache hit)	2	1
Load / Store (Cache miss)	18	1
Integer Add	1	0.33
Integer Multiply	4	1
Integer Divide	36	36
Double/Single FP Multiply	5	2
Double/Single FP Add	3	1
Double/Single FP Divide	38	38

Suppose a is an array of length 500 and all data are always available in cache.
1) What is the expected CPE for call foo(a+1, a , 499) and foo(a, a+1 , 499) .
2) What is the expected CPE without cache?

Problem 2

Suppose we have a block of code shown as below. After running the code, you found that the performance is not very good. Try to optimize the function using any technologies you've learned in the ICS class.

```
void test( )
{
   int x = 1, y = 2;
   int a[100];
   for(int i = 0; i < 100; i++)
   {
      if(i % 2 == 0) a[i] = x;
      else  a[i] = y;
   }
}
```

Problem 3

The assembly code generated for the compiled loop of combine3 is shown below:

```
combine3: data_t = float, OP = *
i in %rdx, data in %rax, dest in %rbp

1 .L498:                          loop:
2   movss (%rbp), %xmm0             Read product from dest
3   mulss (%rax, %rdx, 4), %xmm0    Multiply product by data[i]
4   movss %xmm0, (%rbp)             Store product at dest
5   addq $1, %rdx                   Increment i
6   cmpq %rdx, %r12                 Compare i:limit
7   jg .L498                        If >, goto loop
```

Illustrate the code above with data-flow graph like figure 5.14(a) or (b) in textbook. You can use “store” to identify operation in line 4.