Jonathan Tso
CS 261
HW 7

Q1.

lp: $\left(\frac{40}{200} \cdot \frac{1}{1} \cdot 1\right) = .05$

rp: $\left(\frac{5}{200} \cdot 3\right) = .075$

mp: $\left(\frac{60}{200} \cdot \frac{35}{100} \cdot 2\right) = .21$

CPI: $1 + C_B/C_i$ where C_i = cycle instructions

C_B = cycle bubbles

$C_B = lp + rp + mp = 0.335$

CPI = $1 + .335 = 1.335$

Q2.

```
main:
irmovq $256, %rsp   F D E M W
irmovq $2, %rdx     F D E M W
pushq  %rdx         F D E M W    #1 b/c %rdx needs to update
irmovq $10, %rbx    F D E M W
popq   %rax         F D E M W    #2 b/c you are updating rsp in pop and need to get new rsp from memory
                                 and then forward to next instruction
addq   %rbx, %rax   F D E M W    #3 S1 stall by 1 to get to memory
subq   %rax, %rax   F D E M W    #4 b/c %rax needs to update
je     end          F D          jump does not complete b/c of condition codes
call proc           F D E M W
ret (via proc)      F D E M W
halt                F D #  #  #  halt stops after decode
```

Q3. Optimized Version

```
void inner_product(vec_ptr u, vec_ptr v, data_t *dest) {
    long i;
    long length = vec_length(u);
    data_t *udata = get_vec_start(u);
    data_t *vdata = get_vec_start(v);
    data_t *sum = (data_t) 0;
    data_t *sum2 = (data_t) 0;
    data_t *sum3 = (data_t) 0;
    data_t *sum4 = (data_t) 0;
    data_t *sum5 = (data_t) 0;
    data_t *sum6 = (data_t) 0;
    long length2 = length-5;
  for (i=0; i< length2; i+=6) {

        sum = sum + udata[i] * vdata[i];

        sum2 = udata[i+1] * vdata[i+1];

        sum3 = udata[i+2] * vdata[i+2];

        sum4 = udata[i+3] * vdata[i+3];

        sum5 = udata[i+4] * vdata[i+4];

        sum6 = udata[i+5] * vdata[i+5];
    }

  for (; i< length; i++) {

        sum = sum + udata[i] * vdata[i];

    }

    *dest = sum+sum2+sum3+sum4+sum5+sum6;

}
```

For ideal nonhazards, we have it such that CPE will be approximately (5+i)/(i+1) where i is the number of elements after the first.

For non-ideal hazards, we may hit cycle delays as well as running through the second for loop, which can increase CPE.

Thus, for increasing i, we can get to the approximate CPE of 1.06