HW4 Brent Nix

1. problem 7.2
    a. structural equivalence: A,B,D
    b. Strict name equivalence: A,B
    c. Loose name equivalence: A,B,C,D
2. problem 7.8

0	s	
2	c	▨
4	t	
6	d	▨
8	r	
10		
12		
14		
16	i	
18		
20	s_1	

9-0+1 = 10
10 * 20 = 200 Bytes

3. problem 7.16
8B float
4B int
1B char

/** Column-Major **/
(10-1+1)*(100-10+1) * 8 = 7280
offset A = -7280
offset i = -7284
offset x = -7292

($fp-7292)=($fp-7280)+((($fp-7284)-10)*(100-10+1)+(3-1))*8

```
add $s0,$fp,-7280         //base of A
lw $s1,-7280($fp)         //load i
subi $s1, $s1,10          //i-10
addi $s2,$zero,91
mult $s1, $s1,$s2         //(i-10)*91
addi $s1, $s1, 2          //+2
addi $s2, $zero, 8
mult $s1, $s1, $s2        // ()*8
add $s0, $s0, $s1         //add base
lw $s1,($s0)              //load A[3,i]
sw $s1,-7292($fp)         //store value in x
```

/** Row Pointer **/

base A offset = -(10 * 4)
i offset = -44
x offset = -52
($fp-52) = *(($fp-40) + (3-1)*8)+(($fp-44)-10)*8

```
addi $s0,$fp,-40                    //base of A
addi $s1,$zero, 2                   //(3-1)
addi $s2,$zero, 8
mult $s1,$s1,$s2                    //(3-1)*8
add $s0,$s0,$s1                     //add base
lw $s1,($s0)                        //load A[3]
lw $s0,-44($fp)                     //load i
addi $s0,$s0,10                     //i-10
mult $s0,$s0,$s2                    //(i-10)*8
add $s1,$s1,$s0                     //add Base(*a[3])
lw $s0,($s1)                        //load a[3,i]
sw $s0,-52($fp)                     //store a[3,i] in x
```

b)
blue is record r

-8816	k			
-8812	j			
-8808	x			
-8804	y			
-8800	A[0,0]			
-8720	A[0,1]			

record A[i,j]: 80 bytes

0	z
8	B[k]
80	

(20-10+1)*(10-1+1)*80 = 8800 Bytes

{($fp-8808) + 8 + [(2-1)*(10-1+1)+(($fp-8812)-10)]*80}+8+(($fp-8816)*1

r.A[2,j].B[k]

```
addi $s0,$fp,-8808       //r.
addi $s0,$s0,8           //r.A
addi $s1,$zero,10        //A[2]
lw $s2,-8812($fp)        //load j
```

```
subi $s2,$s2,10           //j offset
addi $s1,$s1,$s2          //[]
addi $s2,$zero,80
mult $s1,$s1,$s2          //[]*80
add $s0,$s0,$s1           //{}
addi $s0,$s0,8            //{}+8
lw $s1, -8816($fp)        //load k
add $s0, $s0, $s1         //r.A[2,j].B[k]
```

4. Dan’s program relies on integers for most of the calculations and they only have 4 bytes of accuracy (minus 1 for a signed int). if he uses doubles, this won’t be a problem until much higher numbers are used