Jessie George
Section 1
HW4

Problem 1

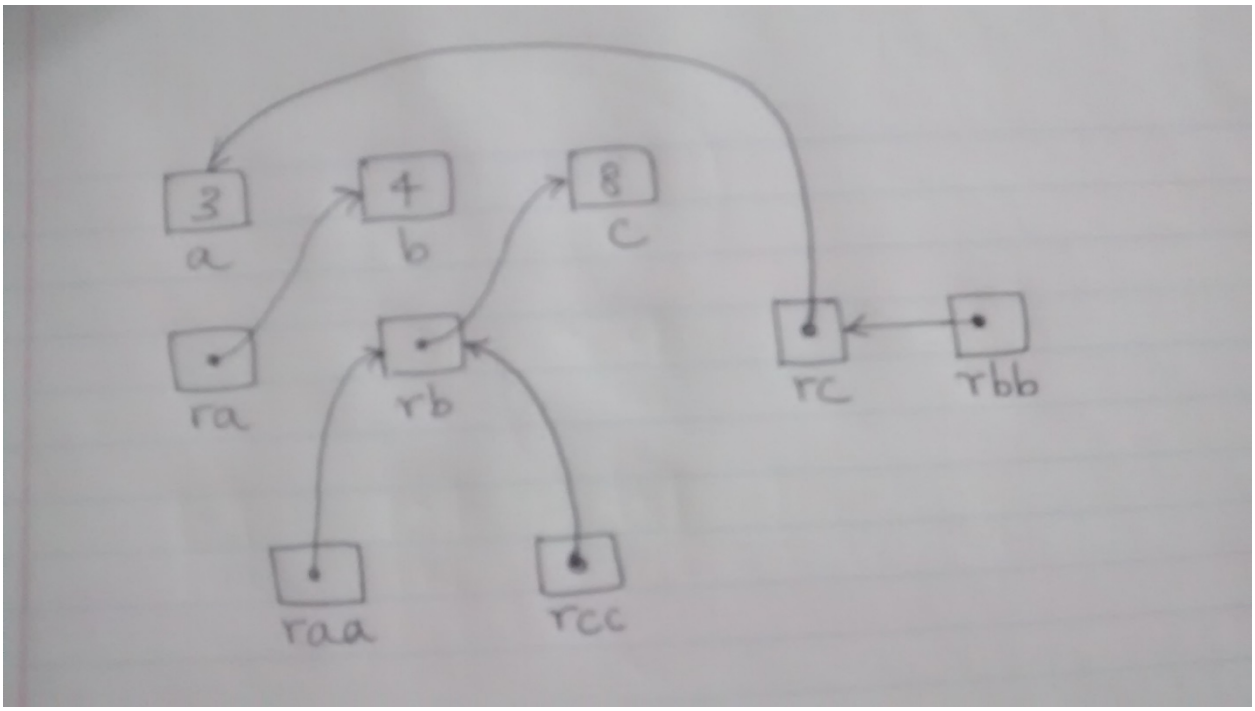
1.1
Excluding rFP, the minimal number of physical registers = 4.
Strategy:
r0 maps to rFP
Use a new register to load a constant, unless the same constant has been loaded in a previous register, then just reuse that register.
If a value of one register is stored to another place and then loaded to another new register, just reuse the same register instead of making a new register.
If a register is never used again, it can be overwritten. (Except always keep rFP)

1.2
```
loadI 1024 -> rFP
loadI 5 -> rA
storeAI rA -> rFP, 0
loadI 1 -> rB
loadI 2 -> rC
loadAI rFP, 0 -> rA
add rC, rA -> rD
add rB, rD -> rC
loadI 5 -> rA
mult rC, rA -> rB
loadI 8 -> rD
sub rB, rD -> rC
storeAI rC -> rFP, 4
outputAI rFP, 4
```

Problem 2

1. ```
   int *ra, *rb, *rc;
   int **raa, **rbb, **rcc;
   ```

2.
   ```



3. Output:
   3 4 8
   4 8
   8 3 8

4. a = **ra;
   //error: invalid type argument of unary ‘*’ (have ‘int’)

Problem 3

1. Yes there’s a safety issue because you will get a Null Pointer Exception at current_cell = current_cell->next
   because current_cell has already been freed, you can’t use null->next.

2. /*DEALLOCATE LIST*/
```
listcell *next = NULL;
for(current_cell = head; current_cell!=NULL; current_cell = next)
{
        next = current_cell->next;
        free(current_cell);
}
```

Problem 4

Constant propagation would not be safe if you alias a or b or both. For example:

```
begin
        int a, b, c;
        int *p = &a;
        a = 5;
        b = 7;
        *p = 100;
        c = a + b;  /* in this case c = 107 not 12 */
        print c;
end.
```