1 **Reference Counting.** The following extends the code from the previous question by adding a procedure `saveIfMax` that is implemented in a separate module. Add calls to `inc_ref` and `dec_ref` to use referencing counting to eliminate all dangling pointers and memory leaks in this code while creating no *coupling* between `saveIfMax` and the rest of the code (i.e., `saveIfMax` can not know about what the rest of the code does and neither can the rest of the code know what `saveIfMax` does). Do not implement reference counting nor worry about storing the reference count itself; just add calls to `inc_ref` and `dec_ref` in the right places, **which may require slightly rewriting portions of the code**.

```
int* copy (int* src) {
  int* dst = malloc (sizeof (int));
        inc_ref (dst);
  *dst = *src;
  return dst;
}
int foo() {
  int  a = 3;
  int* b = copy (&a);
  saveIfMax (b);
  int temp = *b;
  dec_ref (b);


  return *b;
        temp
}
```

```
int* max;
void saveIfMax (int* x) {
  if (max==NULL || *x > *max) {
    if (max != NULL) {
      dec_ref (max);
    }
    max = x;
        inc_ref (max);
  }
}
```

2 **Static Control Flow.** Give SM213 assembly code for the following C statements. Assume that i is a global variable of type int.

2a
```
if (i==0)
    i = 1;
else
    i = 2;
```

2b
```
while (i!=0)
    i -= 1;
```

2a
```
    ld $i, r0
    ld (r0), r0
    beq r0, L0
    ld $2, r1
    br L1
L0: ld $1, r1
L1: st r1, (r0)
    halt
```

2b
```
    ld $i, r0
    ld (r0), r1
L0: beq r1, L1
    dec r1;
    br L0
L1: st r1, (r0)
    halt
```

2b
```
ld $i, r0
```