Eric Chen
11381898

1.
```
a. 11111001      Negative number, so we invert the bits and add 1
                 = 00000110 inverted
                 +        1
                   00000111
= 0*27+0*26+0*25+0*24+0*23+1*22+1*21+1*20
= 4+2+1
= 7
= -7
```

b.
```
= 0*27+1*26+0*25+1*24+1*23+0*22+0*21+0*20
= 26+24+23
= 64+16+8
= 88
```

2.
```
a. -3   = 1100
b. 5    = 0101
```

3.
Instruction set

```
Repeat until we hit the rightmost of the list:
        Compare right
        Note who is larger, set larger kid as largest
Repeat until we hit the rightmost of the list
        Tell kid who the largest one is
```

Program

Assuming each “kid” is a node with fields to store a name(string), a number value(int), and who is largest(string) so at the end they “all know who is largest”

So each kid node has variables: current_kid_name(string), current_kid_value(int), and kid_with_largest_value(string).

```
while (there is still another kid to the right) {  /* find the largest
value */
        if (current_kid_value >  right_kid_value)
        largest_name_string = current_kid_name            /*
largest_name_string is a variable that stores the name of the kid with
the largest name */
        move to next kid
}
```

```
while (there is still another kid to the right) {  /* second
transversal to “tell” each */
        kid_with_largest_value = largest_name_string       /* set
kid_with_largest_value for each kid node to who has the largest
value.*/
                                                           /* this
“tells” each kid which node is the largest */
        move to next kid
}
```