1.
a. 0x04
b. 0x208
c. 0x12
d. 0xFC
e. 0xAA
f. 0x12
g. 0x02
h. 0x12

2.

a. x = 1 + x
b. y = x + 2z
c. x = 8 + 5y
d. z = 12 + 8x
e. y = 9 + x + 4z

3.
a. The purpose of this function is to print out all of the prime numbers between the values a and b.
b. The variables a and b represent the lower and upper ends of the range of numbers.
c. At the very end of the while loop, the value of a would be 51.
d.

```
void mystery (int*a, int*b){
int i, flag;
while (*a <= *b){
flag = 0;
for (i = 2; i <=*a/2; ++ i){
if (*a% i == 0){
flag = 1;
break;
}
}
if (flag == 0)
printf ("%d", *a);
}
}
```

e. mystery (&c, &d);

c = 51, d = 50

It is not a good idea because the values would also be changed for the rest of the code as well. So if we would not be able to use the original values of a and b.

f.

```
for (i = 2; i <= sqrt (a/2); i++){
if (a % i == 0){
flag = 1;
break;
}
}
```

g. I is assigned the right type because we need it to be an integer value so we can iterate through the loop correctly. Since a and b are declared as ints, there is no reason for i to be any larger or smaller than an int, because it will never go outside of that range.

4.

a. x
b. y > x
c. y > z
d. y
e. z
f. z > x
g. z

5.

0x100:
0x100: 2070
0x102: 6020
0x104: 2073
0x106: 6063
0x108: 6123
0x10a: 7600010000
0x10f: 30f101000000
0x121: 6160
0x117: 7523010000
0x11c: 6060
0x11e: 7039010000

0x123:
0x123: 6120
0x125: 7039010000

0x12a:
0x12a: 30f101000000
0x130: 6171
0x132: 7239010000
0x137: 6160

0x139:
0x139: 90

6.

```
.pos 0
        irmovl $0, %eax
        irmovl $1, %ecx
L1:     irmovl $1, %edi
        andl %eax, %edi
        jne L2
        addl %ecx, %eax
L2:     irmovl $1, %edx
        addl %edx, %ecx
        irmovl $100, %edx
        subl %ecx, %edx
        jge L1
        halt
```

This program gets the sum of every even number between the values 1 and 100.