Module 3 Assignment

Q1.
Adding b-1 to a recursively until b == 0.

Pseudocode:
```
sum(a, b){
   if b == 0: return a;
   return 1 + sum(a, b-1);
}
```

Q2.
Accessing each element in the array recursively, and calculate + adding val/# of elem

Pseudocode:
```
avg(arr){
   if (arr.length() == 1) return arr[0];
   return helper(arr, 0);


}


helper(arr, i){
   if (i==arr.length) return 0;
   return ((float)arr[i])/arr.length() + helper(arr, i+1);
}
```

Q3.
If n = 1, it’s a simple comparison, so no recursive call is made.
If n = 2, the midpoint is the first element, if it doesn’t match the target val, the recursive call is made 1, and the process is reduced to n = 1 case. Same for n =3.
If n = 4, .. 7, after one comparison, the recursive call is made and the process can be reduced to n= 2 or 3 cases. 1 + 1 = 2
If n = 8 … 15, dividing the array by half can reduce the process to n = 4 … 7 cases. So 1 +2 = 3
…
For any n, the process can be divided to n/2, n/4, … 2, 1, so the max # of calls would be log2(n)

n	Max # of calls
1	0
2	1
3	1
4	2
5	2
6	2
7	2
8	3
n	log2(n) => O(log2(n))

Q4.

```
int gcd(int x, int y){
   if (y<= x && x%y == 0) return y;
   if (x < y) return gcd(y,x);
   return gcd(y, x%y);
}
```

Q5.

```
int gfib(int f0, int f1, int n){
   if ( n == 0 ) return f0;
   if ( n == 1 ) return f1;
   return gfib(f0, f1, n-1) + gfib(f0, f1, n-2);
}
```

Q6.

```
a(2,2) = a(1, a(2,1))
            a(2,1) = a(1, a(2,0))
                        a(2,0) = a(1,1)
                                a(1,1) = a(0, a(1,0))
                                            a(1,0) = a(0,1)
                                                    a(0,1) = 1+1 = 2
                                            a(1,0) = 2
                                a(1,1) = a(0, 2)
                                        a(0,2) = 2+1 = 3
                                a(1,1) = 3
                        a(2,0) = a(1,1) = 3
           a(2,1) = a(1, a(2,0)) = a(1,3)
                                a(1,3) = a(0, a(1,2))
                                            a(1,2) = a(0, a(1,1))
                                                        a(1,1) = 3
                                            a(1,2) = a(0, 3) = 4
                                a(1,3) = a(0, 4) = 5
           a(2,1) = a(1,3) = 5
a(2,2) = a(1,a(2,1)) = a(1,5)
                    a(1,5) = a(0, a(1,4))
                                a(1,4) = a(0, a(1,3))
                                            a(1,3) = 5
                                a(1,4) = a(0,5) = 6
                    a(1,5) = a(0, 6) = 7
a(2,2) = a(1,5) = 7
```

Q7.

```
int rec(int n){
   while (f(n) == FALSE){
       n = g(n);
   }
   return 0;
}
```