1 Functions - 1

1. Consider the following definition for function `foo`.

```
1        int foo(int a) {
2                return a*a;
3        }
```

   (a) What is the value of `result` when the following code is executed?

```
1        int result = foo(4);
```

   Solution: 16

   (b) What is the value of `result` when the following code is executed?

```
1        int result = foo(foo(4));
```

   Solution: 256

   (c) What is the value of `result` when the following code is executed?

```
1        int result = foo(foo(4) - 6);
```

   Solution: 100

   (d) What is the value of `result` when the following code is executed?

```
1        int result = foo(foo(4) - foo(3));
```

   Solution: 49

2. Consider the following definition for function `foo`.

```
1  int foo(int a, int b) {
2          if(a > b) {
3                  return a;
4          }
5          else {
6                  return b;
7          }
8  }
```

   (a) What is the value of `result` when the following code is executed?

```
1        int result = foo(4, 8);
```

Solution: 8

(b) What is the value of `result` when the following code is executed?

```
int result = foo(13, 8);
```

Solution: 13

(c) What is the value of `result` when the following code is executed?

```
int result = foo(foo(9, 5), 7);
```

Solution: 9

(d) Without knowing the values of `a, b, c, d, e, f`, what is it that you can tell about the value of `result` when the following code executes?

```
int result = foo(foo(foo(a, b), foo(c, d)), foo(e
    , f));
```

Solution: `result` stores the highest of a, b, c, d, e, f.

(e) Without knowing the values of `a, b, c, d, e, f`, what is it that you can say about the value of `result1, result2` (comparatively) when the following code executes?

```
int result1 = foo(foo(foo(a, b), foo(c, d)), foo(e, f));
int result2 = foo(foo(foo(foo(foo(a, b), c), d), e), f);
```

Solution: `result1` and `result2` will be equal.

3. Write a function that when passed two floating-point variables, returns the smaller of the two.

Solution:

```
float smaller(float a, float b) {
        if(a < b) {
                return a;
        }
        else {
                return b;
        }
}
```

4. Write a function that when passed two floating-point variables, returns `true` if they are both positive, and `false` otherwise.

Solution:

```
boolean bothPositive(float a, float b) {
        if(a > 0 && b > 0) {
                return true;
        }
        else {
                return false;
        }
}
```

5. Write a function that when passed two integers, returns `true` if they are both even, and `false` otherwise.

Solution:

```
boolean bothEven(float a, float b) {
        if(a % 2 == 0 && b % 2 == 0) {
                return true;
        }
        else {
                return false;
        }
}
```

6. Write a function that when passed two integers, returns the highest integer by which they are both divisible. Such an integer is called the *greated common divisor*. For example, greatest common divisor of 40 and 24 is 8, that of 32 and 27 is 1, that of 12 and 12 is 12, that of 24 and 48 is 24.

Solution: Solution 1: very inefficient

```
int gcd(int a, int b) {
        int result = 1;
        int smaller = a; //assume a < b
        if(b < a) {
                smaller = b;
        }
        for(int i=1; i < smaller; i++) {
                if(a % i == 0 && b % i == 0) {
                        result = i;
                }
        }
        return result;
}
```

Solution 2: better

```
int gcd(int a, int b) {
        int result = 1;
        int smaller = a; //assume a < b
        if(b < a) {
                smaller = b;
        }
        for(int i=smaller; i > 1; i--) {
                if(a % i == 0 && b % i == 0) {
                        return i;
                }
        }
        return 1; //no common divisor
}
```

Solution 3: Euclid, you beauty :)

```
int gcd(int a, int b) {
        if(a < b) {
                int temp = a;
                a = b;
                b = temp;
        }
        //so we are certain that a >= b

        while(b != 0) {
                int temp = a % b;
                a = b;
                b = temp;
        }

        return a;
}
```