LECTURE 8 MATH AND MAKING DECISIONS CLICKER QUESTION SOLUTIONS

1) Which is the correct way to call the static method numberOfFish on the Ocean class?

a) OceanInstance.numberOfFish();
b) Ocean.numberOfFish(static);
c) Ocean.numberOfFish();
d) OceanInstance.numberOfFish(static);

ANSWER C: A static method is called on the class itself, not an instance of the class.

2) Which if statement will run if the Grinch does not steal Christmas and Horton does hear a who?

a) if (!grinch.stole() && horton.hears()){}
b) if (grinch.stole() && !horton.hears()){}
c) if (grinch.stole() && horton.hears()){}
d) if (!grinch.stole() && !horton.hears()){}

ANSWER A: the not operator "!" is used to ensure that grinch.stole() returns false, so we know the Grinch does "not" steal Christmas, while horton.hears() is checked to be true within the if statement, so we know he did hear.

3) Which print statement will be printed out?

```
int x = 10;
if (x < 10) {
        if ((x+10)>15) {
                System.out.println("x + 10 is greater than 15");
        } else {
                System.out.println("x is less than 10");
        }
} else if (x <= 15) {
        if ((x+2) > 13) {
                System.out.println("x + 2 is greater than 13");
        } else {
                System.out.println("x is less than or equal to 11");
        }
} else {
        System.out.println("x is greater than 15");
}
```

ANSWER D: x is equal to 10, so it skips the first if statement because it is not less than 10. It satisfies the else condition x<=15 because 10 is less than or equal to 15. However, it fails the if statement ((x+2) > 13) because 10 + 2 = 12 and 12 is not greater than 13. Thus, the else statement "x is less than or equal to 11" is printed.

4) What is a valid way to have overloaded methods?

a) Two methods that are absolutely identical.
b) Two methods that are the same, except in their return type.
c) Two methods that have the same name, but different parameters and/or return types.
d) Two methods that are the same, except one contains an error

ANSWER C: A method is overloaded if it has the same name, but has different parameters and/or return types. Thus, Java can differentiate between the two methods, and run the correct one depending on the method signature.