CMPSCI 187 (Spring 2019) Lab 06: Recursion

The lab is due by 8:00 pm today. Please make sure that you complete your lab assignment in time.

- Go to ***File -> Make a Copy*** to make an editable copy of this Google Doc for your account
- Follow the instructions below to complete the lab
- When you are done, go to ***File -> Download As -> PDF Document***
- Log in to Gradescope and submit your PDF. Remember to submit to **Lab 06**, please **Do NOT submit to Project 06**.

Section A: Fill in the Blanks [8 points].

1. **Describe the three conditions of recursion [3 pts].**

Base case
Must move towards base case
Recursive call to itself

2. **Given the following recursive method, what would `dot(4)` print out? [1 pts]**

```
public void dot(int n) {
	if(n>0) {
		System.out.print(n);
		dot(n-1);
	}
}
```

4321

3. **Given the following recursive method, what would `foo(4)` print out? [1 pts]**

```
public void foo(int n) {
	if(n>0) {
		foo(n-1);
		System.out.print(n);
	}
}
```

1234

4. **Given the following recursive method, what would `bar(4)` print out? The answer is not trivial. Think carefully. Hint: what would `bar(1)` print out? How about `bar(2)` and `bar(3)`? If you have already figured out what `bar(n-1)` prints out, can you quickly figure out what `bar(n)` prints out? [3 pts]**

```
public void bar(int n) {
	if(n>0) {
		bar(n-1);
		System.out.print(n);
		bar(n-1);
	}
}
```

121312141213121

Section B: Programming [9 pts].

In this section, you must implement each method **using recursion**, even though some may be easy to implement using other approaches. These are good exercises to train you to think recursively. To fulfill this goal, you are **NOT allowed** to use loops anywhere in your code (such as `for`, `while`, or `do` statements). In addition, you are **NOT allowed** to use anything from the `java.util.Math` class. You will receive a zero if you violate any of these requirements.

1. **Write a method to return true if val is even and false otherwise.** Without using recursion this could be simply: `return val%2==0`. Here you must use recursion to implement this method, and cannot use %. You must correctly handle all integers, positive or negative. Hint: what are the base cases? What should isEven(0) return? What about isEven(1)? How do you make sure every recursive call makes progress towards the base case? NOTE: A negative number is even if its negation is even. [3 pts]

```
public boolean isEven(int val) {
	if(val==1||val==-1){return false;}
	if(val==0){return true;}
	else{
		if(val<0){
		isEven(val+2);}
		else{
		isEven(val-2);}
	}
}
```

2. **Write a method to return the sum of all integers between 0 and n.** Note that n can be positive or negative. For example, if n is 5, this method returns the sum from 0 to 5. if n is -10, this method returns the sum from -10 to 0. Your code must NOT contain *, /, *= or /=. [3 pts]

```
public int sumN(int n) {
	if(n==0){reutrn 0;}
	else if(n<0){
		return n + sumN(n+1);
	}
	else{
		return n + sumN(n-1);}

}
```

3. **Write a method to return 2 to the nth power, i.e. 2^n.** Remember, you must implement this recursively. No loop is allowed. You may NOT use anything from java.util.Math package. Return 0 if n is less than 0. [3 pts]

```
public int biPower(int n) {
	if(n==0){
	return 1;
	}
	if(n<0){
	return 0;
	}
	else {
	return 2 * biPower(n-1);}

}
```