Algorithmics	Student information	Date	Number of session
	UO: 275725		3.1
	Surname: Gómez Menéndez		
	Name: Laura		

Escuela de Ingeniería Informática
Universidad de Oviedo

Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

Activity 1. [Basic recursive models.]

A brief explanation for each of the given classes indicating how you calculated the complexity of that class.

- **Subtraction1**

```
public class Subtraction1{
   public static long rec1(int n) {
          long cont = 0;
          if (n<=0)
                 cont++;
          else {
                 cont++;  // O(1)=O(n^0)
        rec1(n-1);
          }
          return cont;
   }
```

In this method we only have one subproblem (rec(n-1)), we can say that a = 1.

We know that b = 1 because the number that substracts n in the subproblem is 1.

As there is no loops, k = 0.

In conclusion, as a = 1, the complexity is O(n^(k+1)) that is **O(n).**

- **Subtraction2**

```
public static long rec2(int n) {
          long cont = 0;
          if (n<=0)
                 cont++;
          else {
                 for (int i=0;i<n;i++)
                        cont++; // O(n)
                 rec2(n-1);
          }
          return cont;
   }
```

In this method we only have one subproblem (rec2(n-1)), we can say that a = 1.

We know that b = 1 because the number that substracts n in the subproblem is 1.

As there is one loop, k = 1.

In conclusion, as a = 1, the complexity is O(n^(k+1)) that is **O(n^2).**

Algorithmics	Student information	Date	Number of session
	UO: 275725		3.1
	Surname: Gómez Menéndez		
	Name: Laura		

- **Subtraction3**

```
public static long rec3(int n) {
		long cont = 0;
		if (n<=0)
			cont++;
		else {
			cont++;  //O(1)
			rec3(n-1);
			rec3(n-1);
		}
		return cont;
	}
```

In this method we have two subproblems (rec3(n-1)), we can say that a = 2.

We know that b = 1 because the number that substracts n in the subproblem is 1.

As there is no loops, k = 0.

In conclusion, as a > 1, the complexity is O(a^(n/b)) that is **O(a^n).**

- **Division1**

```
public static long rec1 (int n) {
		long cont = 0;
		if (n<=0) cont++;
		else {
			for (int i=1;i<n;i++)
				cont++ ;  //O(n)
		rec1(n/3);
		}
		return cont;
	}
```

In this method we have one subproblem (rec1(n/3)), we can say that a = 1.

We know that b = 3 because the number that divides n in the subproblem is 3.

As there is one loop, k = 1.

In conclusion, as a < b^k, the complexity is O(n^k) that is **O(n).**

Algorithmics	Student information	Date	Number of session
	UO: 275725		3.1
	Surname: Gómez Menéndez		
	Name: Laura		

- **Division2**

```
public static long rec2 (int n) {
		long cont = 0;
		if (n<=0) cont++;
		else {
			for (int i=1;i<n;i++)
				cont++ ;  //O(n)
			rec2(n/2);
			rec2(n/2);
		}
		return cont;
	}
```

In this method we have two subproblems (rec2(n/2)), we can say that a = 2.

We know that b = 2 because the number that divides n in the subproblem is 2.

As there is one loop, k = 1.

In conclusion, as a = b^k, the complexity is O((n^k)*logn) that is **O(nlogn).**

- **Division3**

```
public static long rec3 (int n) {
		long cont = 0;
		if (n<=0)
			cont++;
		else {
			cont++ ; // O(1)
			rec3(n/2);
			rec3(n/2);
		}
		return cont;
	}
```

In this method we have two subproblems (rec3(n/2)), we can say that a = 2.

We know that b = 2 because the number that divides n in the subproblem is 2.

As there no loops, k = 0.

In conclusion, as a > b^k, the complexity is O(n^(log_b(a))) that is **O(n^(log_2(2))).**

Algorithmics	Student information	Date	Number of session
	UO: 275725		3.1
	Surname: Gómez Menéndez		
	Name: Laura		

A brief explanation for each of the two new classes indicating how you calculate the complexity to get the requested one.

- **Subtraction4**

I was asked to write a recursive method by subtraction with a complexity O(3^(n/2)) .

With that information I can know that a > 1, and a = 3, so I had to write three subproblems.

I can also know that b = 2, so the number that divides n in the subproblems is 2. The variable k does not matter at all, so I did not write loops and k = 0 in my class.

```
public static long rec1(int n) {
		long cont = 0;
		if (n<=0)
			cont++;
		else {
			cont++;
		rec1(n-2);
		rec1(n-2);
		rec1(n-2);

		}
		return cont;
	}
```

- **Division4**

This time I was asked to create a new recursive method by division with a complexity O(n^2) and a number of subproblems = 4. The number of subproblems is the same as the variable a, so a = 4. With that information I can know that a < b^k, and that k = 2 what means that I had to write two loops.

I had to look for a number whose square was bigger than 4, so I chose b = 3 (4<9), what means that the number that divides n in the subproblems Is 3.

```
public static long rec2 (int n) {
		long cont = 0;
		if (n<=0) cont++;
		else {
			for (int i=1;i<n;i++)
				for(int j=i;j<n;i++)
					cont++ ;  //O(n)
			rec2(n/3);
			rec2(n/3);
			rec2(n/3);
			rec2(n/3);
		}
		return cont;}
```

Algorithmics	Student information	Date	Number of session
	UO: 275725		3.1
	Surname: Gómez Menéndez		
	Name: Laura		