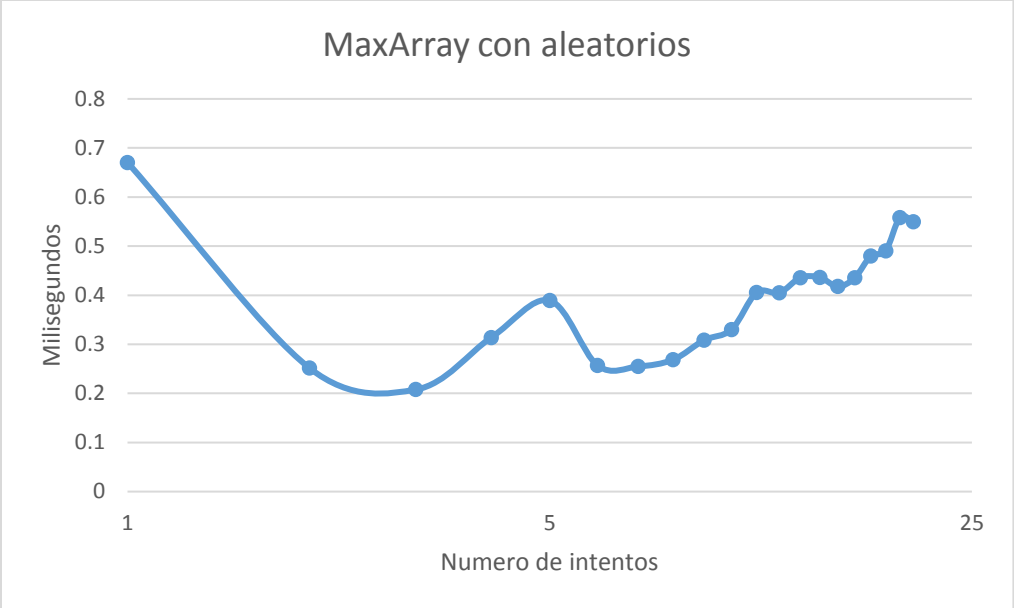
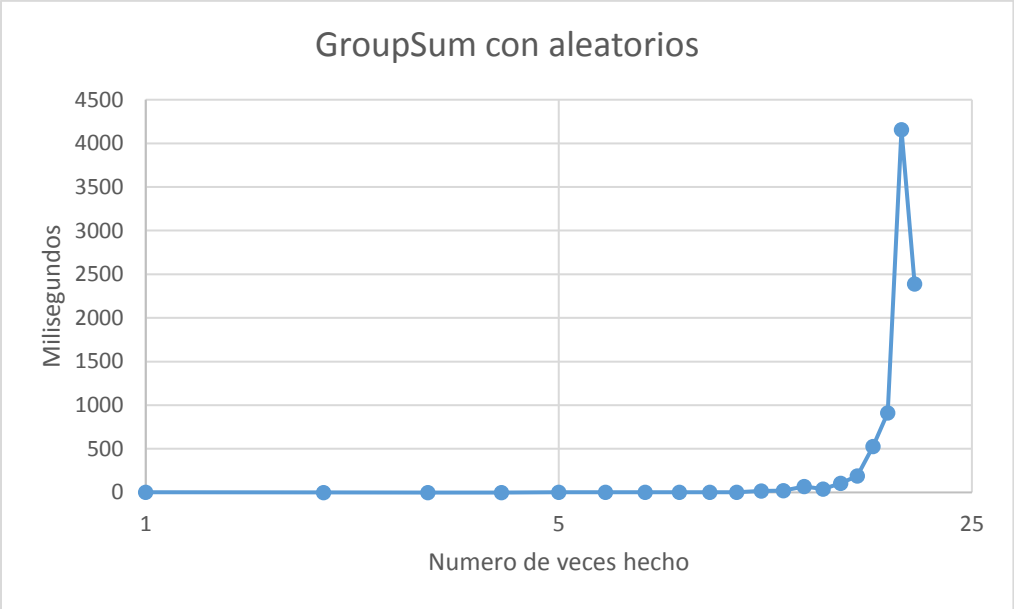
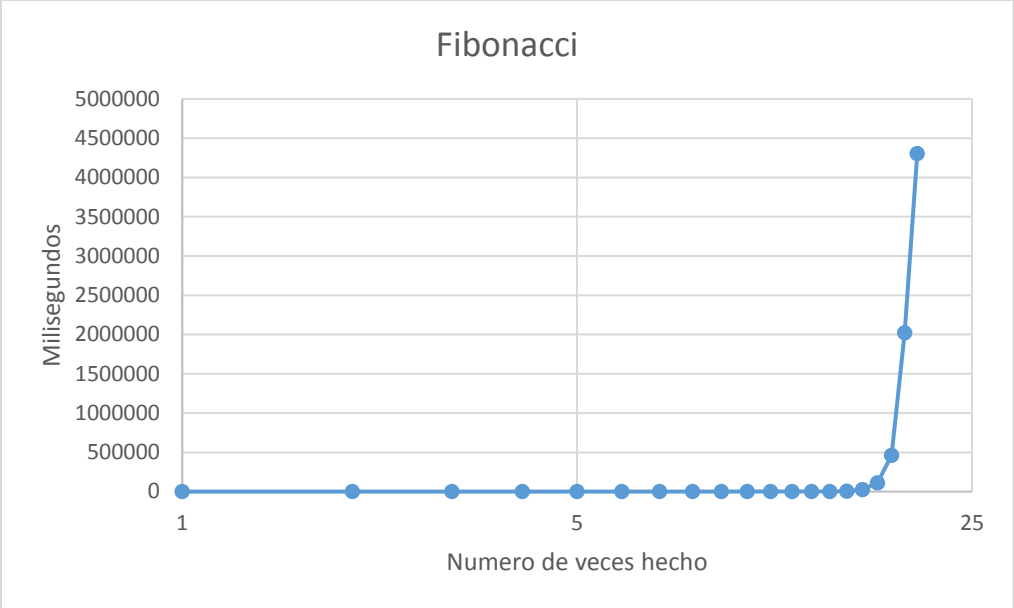
1. Algoritmos ArrayMax, GroupSum y Fibonacci implementados

```java
//ArrayMax
public static int arrayMax(int[] array, int n) {
    int max = array[n];
    int temp;
    if(n!= 0){
        temp = arrayMax(array, n-1);
        if(temp>max) max = temp;
    }
    return max;
}

//groupSum
public static boolean groupSum(int start, int[] nums, int target) {
    if(start == nums.length){
        return target==0;
    }
    else{
        return groupSum(start+1,nums,target-nums[start])||
        groupSum(start+1,nums,target);
    }
}

//fibonacci
public static long fibonacci(int n) {
    if(n==0)    return 0;
    if(n==1)    return 1;
    else return fibonacci(n-2)+fibonacci(n-1);
}

public static int[] randarray(int size){
    int[] array = new int[size];
    for(int i = 0; i<size;i++){
        array[i] = (int)(Math.random()*500);
    }

    return array;
}
```

2. El tamaño del algoritmo en arrayMax y en groupSum está dado por “int n” e “int target” que básicamente un parámetro para el tamaño del arreglo, entre más largo sea este, más largo por asi decirlo será el algoritmo. En el caso de Fibonacci, el tamaño también está dado por “int n” pues este parámetro especifica el número de veces que se tiene que repetir el algoritmo.
3. Los valores apropiados para el primer problemas serian
   Array Max: 700, 1400, 2100, 2800, 3500, 4200, 4900, 5600, 6300, 7000, 7700, 8400, 9100, 9800, 10500, 11200, 11900, 12600, 13300, 14000
   Group Sum: 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32
   Fibonacci: 3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48, 51, 54, 57, 60,
4. Graficas:


MaxArray con aleatorios
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Milisegundos
1
5
25
Numero de intentos


GroupSum con aleatorios
4500
4000
3500
3000
2500
2000
1500
1000
500
0
Milisegundos
1
5
25
Numero de veces hecho


Fibonacci
5000000
4500000
4000000
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
Milisegundos
1
5
25
Numero de veces hecho