Simón Correa y David Gómez

Taller #6

1. Cálculo de complejidad

```
Abeja temp;
        if (index < 0 || index > size)
{                                   //c_1  = 4
            throw new IndexOutOfBoundsException("Index : "+
index);     //
        } else if (index == size)
{                                       //c_3 = 2
            add(e);
        } else
{                                                           //c_4 = 1
            for (int i = index; i < size+1; i++)
{                       //c_5 = 1
                temp = elements[i];
                elements[i] = e;
                e = temp;                                   //T(n) = c_5
+c_6(n+1)          c_6 = 8
            }
            size++;
        }
```

La ecuación de recurrencia sería:

$$T(n) = \begin{matrix} c\_1 + c\_2 \\ c\_3 \\ c\_4 + c\_5 + c\_6(n+1) \end{matrix}$$

Para calcular la complejidad:

O(c_4 + c_5 + c_6(n+1)	//Por Def. de O
O(c_6(n+1))	//Por Regla de la Suma
O(n+1)	//Por Regla del producto
O(n)	//Por Regla de la Suma

Siendo así, la complejidad de agregar n Abejas es O(n).

Concluyendo observamos que, al agregar millones de abejas, la complejidad se vuelve lineal en el peor de los casos, por lo que, aunque no comprometa en gran medida el rendimiento, si se sigue incrementando el número de abejas por agregar, la eficiencia del algoritmo puede no ser suficiente, o podría verse afectada.