Ordenar

swap

```
1 void swap(reunion &r, int i, int j) {
2     pair<senial, hablante> aux = r[i];
3     r[i] = r[j];
4     r[j] = aux;
5 }
```

c_1	1
c_2	1
c_3	1

- $m = |r|$
- $T_{swap}(m) = c_1 + c_2 + c_3$
- $T_{swap}(m) \in O(1)$

tono

```
1 float tono(senial s){
2     float sumatoria = 0;
3     for(int i=0; i < s.size(); i++){
4         sumatoria = sumatoria + abs(s[i]);
5     }
6     return sumatoria / s.size();
7 }
```

c_1''	1
c_2''	n+1
c_3''	n
c_4''	1

- $n = |s|$
- $T_{tono}(n) = c_1'' + c_2'' * (n+1) + c_3'' * n + c_4''$
- $T_{tono}(n) \in O(n)$

insert

```
1 void insert(reunion &r, int i) {
2     int j = i;
3     while (j > 0 && tono(r[j].first) > tono(r[j - 1].first)) {
4         swap(r, j, j - 1);
5         j--;
6     }
7 }
```

c_1'''	1
$c_2''' * (n+n)$	m+1
c_3'''	m
c_4'''	m

- $m = |r|$
- $T_{insert}(m) = c_1''' + c_2''' * 2n * (m+1) + c_3''' * m + c_4''' * m$
- $T_{insert}(m) \in O(n * m)$

insertionSort

```
1 void insertionSort(reunion &r) {
2     int i = 0;
3     while (i < r.size()) {
4         insert(r, i);
5         i++;
6     }
7 }
```

c_1''''	1
c_2''''	m+1
$c_3'''' * (n * m)$	m
c_4''''	m

- $m = |r|$
- $T_{insertSort}(m) = c_1'''' + c_2'''' * (m+1) + c_3'''' * n * m * m + c_4'''' * m$
- $T_{insertSort}(m) \in O(n * m^2)$

ordenar

```
1 void ordenar(reunion &r, int freq, int prof) {
2     insertionSort(r);
3 }
```

| $c_1'''' * (n * m^2)$ | 1

- $m = |r|$
- $T_{ordenar}(m) = c_1'''' * (n * m^2)$
- $T_{ordenar}(m) \in O(n * m^2)$