# Selection Sort & HeapSort

- Airam Navas Simón
- Ruymán Rodríguez Martín
- Jorge Sierra Acosta

Repositorio Github

## Selection Sort -- $\Theta(n^2)$

```
selectionSort(A):
      for i = 0 to A.size
            min = selectMin(A, i, A.size)
            swap(A[min], A[i]);
      return

selectMin(A, start, end):
      min = start;
      for i = start to end
            if A[i] < A[min]
                  min = i
      return min

swap(a, b):
      aux = a
      a   = b
      b   = aux
      return
```

# Selection Sort -- Θ($n^2$)

```
                                                  Repeticiones
selectionSort(A): ______________________________  1
      for i = 0 to A.size _______________________ n
            min = selectMin(A, i, A.size)________ t₁
            swap(A[min], A[i]); _________________ t₂
      return ____________________________________ 1

selectMin(A, start, end):
      min = start;
      for i = start to end
            if A[i] < A[min]
                  min = i
      return min

swap(a, b):
      aux = a
      a   = b
      b   = aux
      return
```

# Selection Sort -- Θ(n²)

```
                                                        Repeticiones
selectionSort(A): _____________________________________ 1
      for i = 0 to A.size _____________________________ n
            min = selectMin(A, i, A.size)______________ t₁
            swap(A[min], A[i]); _______________________ t₂
      return __________________________________________ 1

selectMin(A, start, end): _____________________________ 1
      min = start; ____________________________________ 1
      for i = start to end ____________________________ n
            if A[i] < A[min] __________________________ 1
                  min = i _____________________________ 1, 0
      return min ______________________________________ 1

swap(a, b):
      aux = a
      a   = b
      b   = aux
      return
```

# Selection Sort -- Θ(n²)

```
                                                  Repeticiones
selectionSort(A): _______________________________ 1
      for i = 0 to A.size _______________________ n
            min = selectMin(A, i, A.size)________ t₁
            swap(A[min], A[i]); _________________ t₂
      return ____________________________________ 1

selectMin(A, start, end): _______________________ 1
      min = start; ______________________________ 1
      for i = start to end ______________________ n
            if A[i] < A[min] ____________________ 1
                  min = i _______________________ 1, 0
      return min ________________________________ 1

swap(a, b): _____________________________________ 1
      aux = a ___________________________________ 1
      a   = b ___________________________________ 1
      b   = aux _________________________________ 1
      return ____________________________________ 1
```

# Selection Sort -- Θ(n²)

**Repeticiones**

```
selectionSort(A): ____________________________________ 1
      for i = 0 to A.size ____________________________ n
            min = selectMin(A, i, A.size) ____________ $t_1$
            swap(A[min], A[i]); ______________________ $t_2$
      return _________________________________________ 1
```

$$T(n) = nt_1c_4 + nc_5 + c_6$$

$$T(n) = n^2c_9 + nc_8 + c_7$$

$$T(n) \in \Theta(n^2)$$

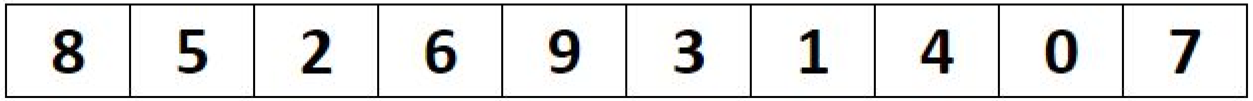
```
selectMin(A, start, end): ____________________________ 1
      min = start; ___________________________________ 1
      for i = start to end ___________________________ n
            if A[i] < A[min] _________________________ 1
                  min = i ____________________________ 1, 0
      return min _____________________________________ 1
```

$$t_1 = nc_2 + c_1$$

```
swap(a, b): __________________________________________ 1
      aux = a ________________________________________ 1
      a   = b ________________________________________ 1
      b   = aux ______________________________________ 1
      return _________________________________________ 1
```

$$t_2 = c_3$$

## Selection Sort -- Θ($n^2$)



| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 8 | 5 | 2 | 6 | 9 | 3 | 1 | 4 | 0 | 7 |

## HeapSort -- Θ(n log(n))

```
heapSort(A):
      for i = (A.size/2) to 1
            sink(A, i)
      for i = A.size to 2
            swap(A[1], A[i])
      sink(A, i)
      return
```

## HeapSort -- Θ(n log(n))

```
                                                        Repeticiones
heapSort(A): __________________________________________ $1$
     for i = (A.size/2) to 1 __________________________ $n/2$
           sink(A, i) _________________________________ $t_1$
     for i = A.size to 2 ______________________________ $n-1$
           swap(A[1], A[i])____________________________ $t_2$
     sink(A, i) _______________________________________ $t_1$
     return ___________________________________________ $1$
```

# HeapSort -- Θ(n log(n))

```
                                                      Repeticiones
heapSort(A): ________________________________________ 1
    for i = (A.size/2) to 1 _________________________ n/2
        sink(A, i) __________________________________ t₁
    for i = A.size to 2 _____________________________ n-1
        swap(A[1], A[i])_____________________________ t₂
    sink(A, i) ______________________________________ t₁
    return __________________________________________ 1
```

$$T(n) = nt_1c_3 + nc_4 + c_5$$

# HeapSort -- Θ(n log(n))

```
sink(A, i):
      x = A[i-1]
      while 2i <= A.size
            left = 2i
            right = 2i+1
            if (left == A.size) or (A[left] > A[right])
                  max = left
            else
                  max = right
            if (A[max] <= x)
                  break
            else
                  swap(A[i], A[max])
                  i = max
      return
```

# HeapSort -- Θ(n log(n))

```
                                                      Repeticiones
sink(A, i):                                                1
    x = A[i-1]                                             1
    while 2i <= A.size                                     logn
        left = 2i                                          1
        right = 2i+1                                       1
        if (left == A.size) or (A[left] > A[right])        1
            max = left                                     1
        else
            max = right                                    1
        if (A[max] <= x)                                   1
            break                                          1
        else
            swap(A[i], A[max])                             $t_2$
            i = max                                        1
    return
```

# HeapSort -- Θ(n log(n))

```
                                                      Repeticiones
sink(A, i): _________________________________________ 1
    x = A[i-1] ______________________________________ 1
    while 2i <= A.size ______________________________ logn
        left = 2i ___________________________________ 1
        right = 2i+1 ________________________________ 1
        if (left == A.size) or (A[left] > A[right]) _ 1
            max = left ______________________________ 1
        else
            max = right _____________________________ 1
        if (A[max] <= x) ____________________________ 1
            break ___________________________________ 1
        else
            swap(A[i], A[max]) ______________________ $t_2$
            i = max _________________________________ 1
    return
```

$$t_1 = \log(n)c_1 + c_2$$

## HeapSort -- $\Theta(n \log(n))$

$T(n) = nt_1c_3 + nc_4 + c_5$

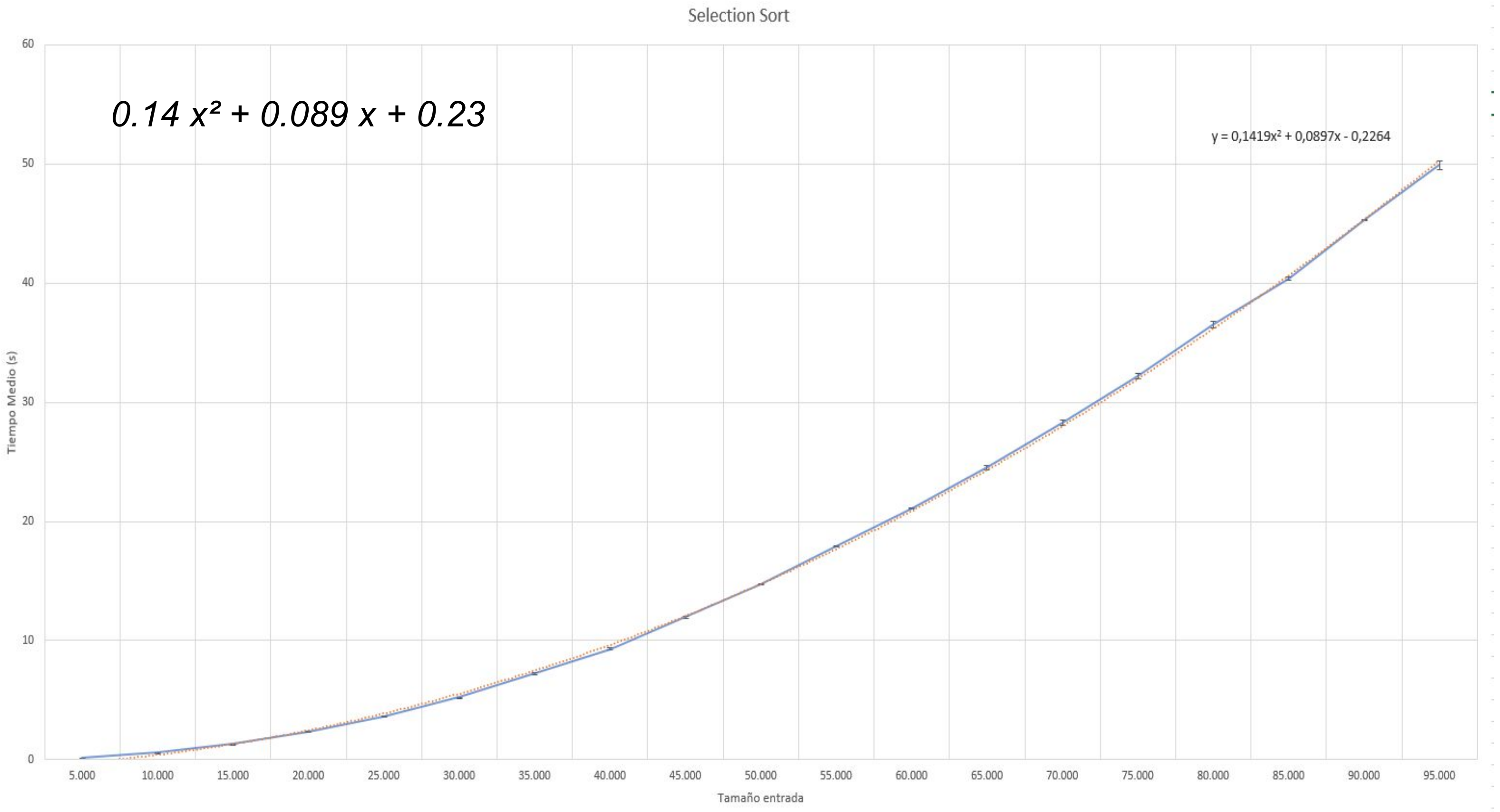
$t_1 = \log(n)c_1 + c_2$

$T(n) \in \Theta(n\log(n))$

## HeapSort -- Θ(n log(n))

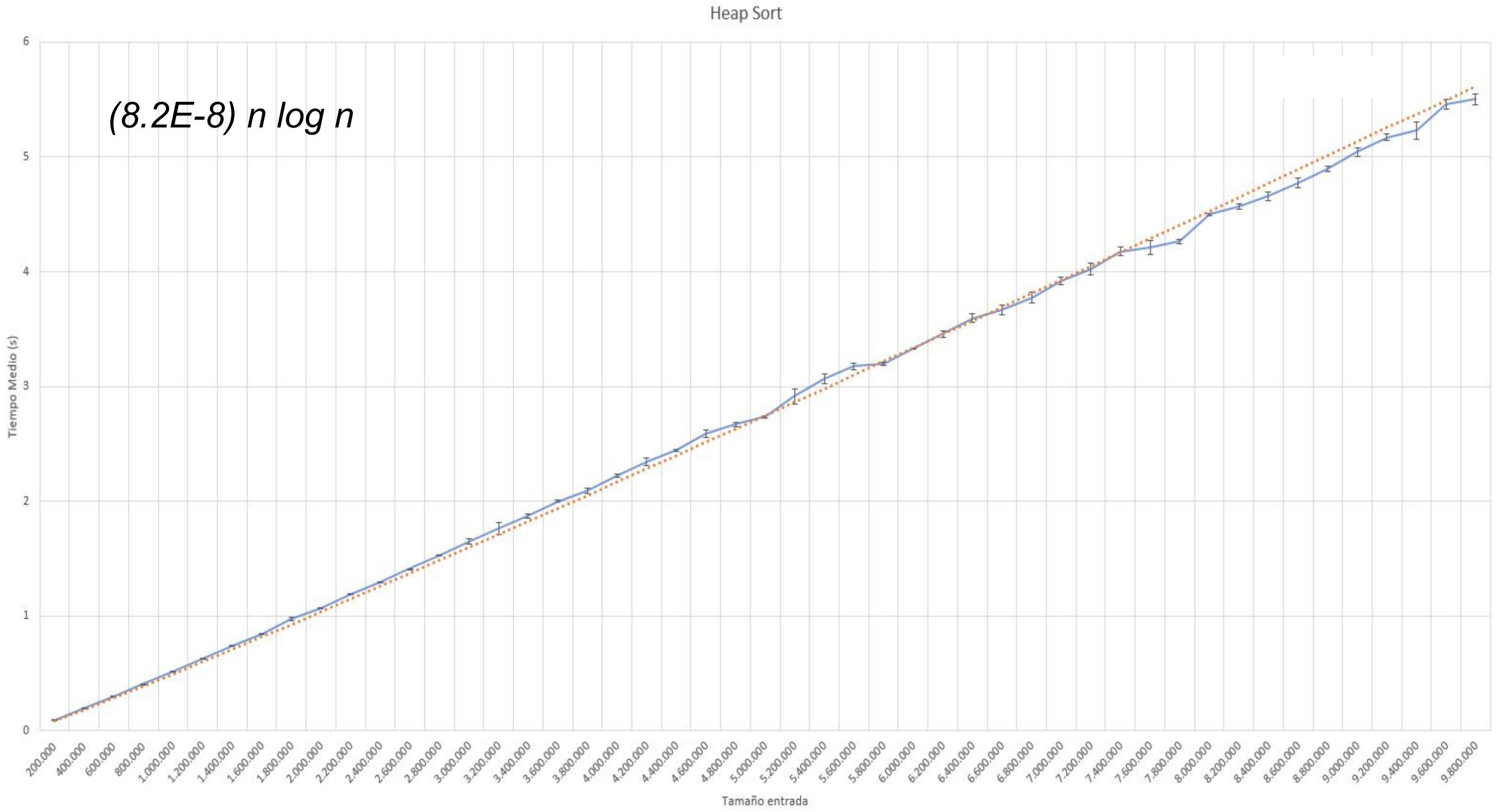
6 5 3 1 8 7 2 4

## Caso real -- Selection Sort -- $\Theta(n^2)$

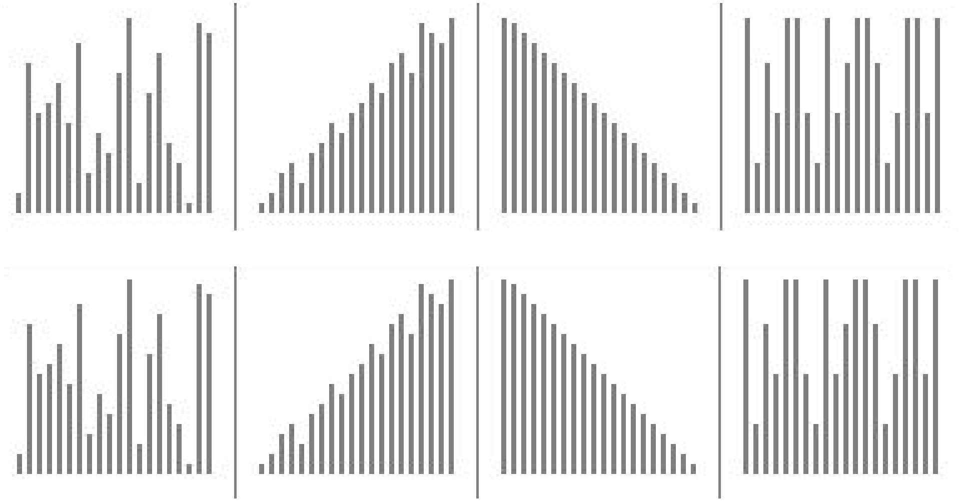


$0.14\ x^2 + 0.089\ x + 0.23$

## Caso real -- HeapSort -- Θ(n log(n))



# Comparación



## Bibliografía

https://www.toptal.com/developers/sorting-algorithms/

https://en.wikipedia.org/wiki/Selection_sort

https://en.wikipedia.org/wiki/Heapsort

PDF’s: Diapositivas Tema 1 DAA

FIN