

Treball de Recerca:
Algoritmes d'ordenació

Joan Coma Bages

Curs 2020/2021

Índex

1	Introducció	3
2	Què és ordenar un vector?	4
3	Selection sort	5
3.1	Com funciona?	5
3.2	Pseudocodi	5
3.3	Implementació	5
3.4	Rendiment	6
4	Insertion sort	7
4.1	Com funciona?	7
4.2	Pseudocodi	7
4.3	Implementació	7
4.4	Rendiment	8
5	Bubble sort	9
5.1	Com funciona?	9
5.2	Pseudocodi	9
5.3	Implementació	9
5.4	Rendiment	10
6	Shell sort	11
6.1	Com funciona?	11
6.2	Pseudocodi	11
6.3	Implementació	11
6.4	Rendiment	12
7	Quick sort	13
7.1	Com funciona?	13
7.2	Pseudocodi	13

7.3	Implementació	13
7.4	Rendiment	14
8	Merge sort	15
8.1	Com funciona?	15
8.2	Pseudocodi	15
8.3	Implementació	15
8.4	Rendiment	16
9	Comparació dels algoritmes	17

1. Introducció

2. Què és ordenar un vector?

3. Selection sort

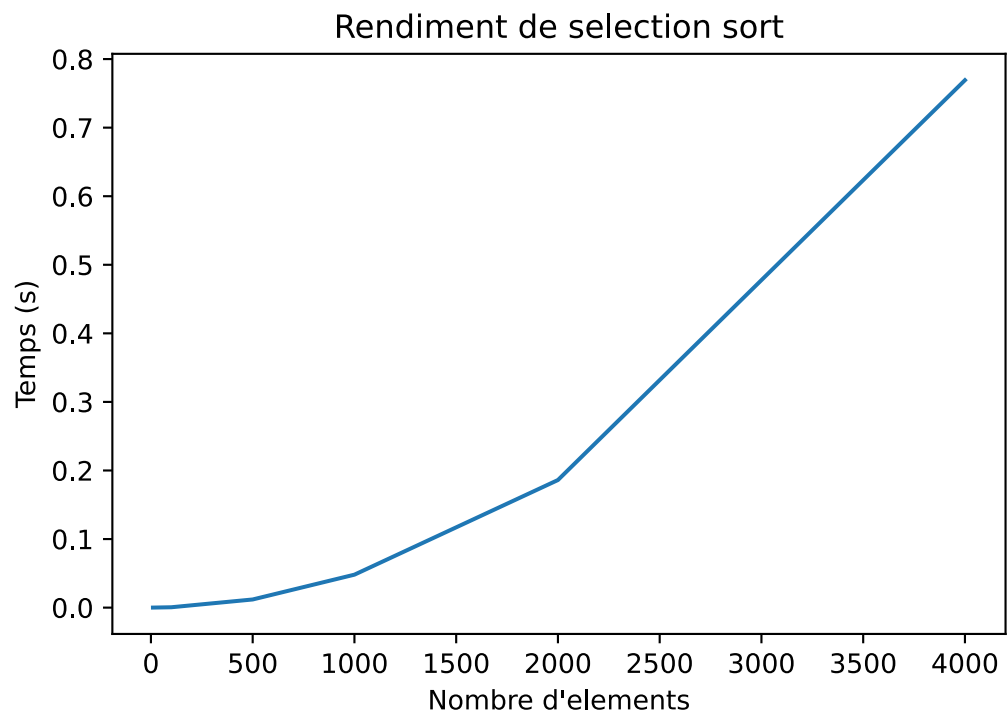
3.1. Com funciona?

3.2. Pseudocodi

3.3. Implementació

```
1  #!/usr/bin/env python3
2  import utils
3
4  def sort(array):
5      for i in range(len(array)-1):
6          low = i
7          for j in range(i, len(array)):
8              if array[low] > array[j]:
9                  low = j
10
11         array[i], array[low] = array[low], array[i]
12     return array
13
14 if __name__ == "__main__":
15     array = utils.numbers()
16     print(sort(array))
```

3.4. Rendiment



4. Insertion sort

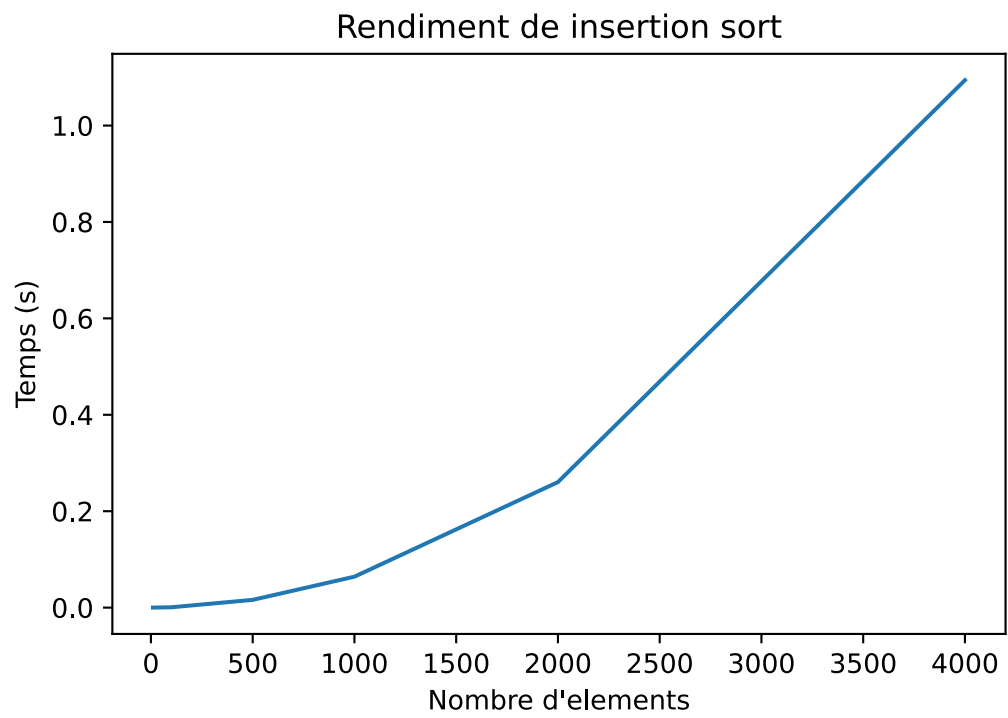
4.1. Com funciona?

4.2. Pseudocodi

4.3. Implementació

```
1  #!/usr/bin/env python3
2  import utils
3
4  def sort(array):
5      for i in range(len(array)):
6          for j in range(len(array[:i])):
7              if array[i] < array[j]:
8                  array[i], array[j] = array[j], array[i]
9
10     return array
11
12 if __name__ == "__main__":
13     array = utils.numbers()
14     print(sort(array))
```


4.4. Rendiment



5. Bubble sort

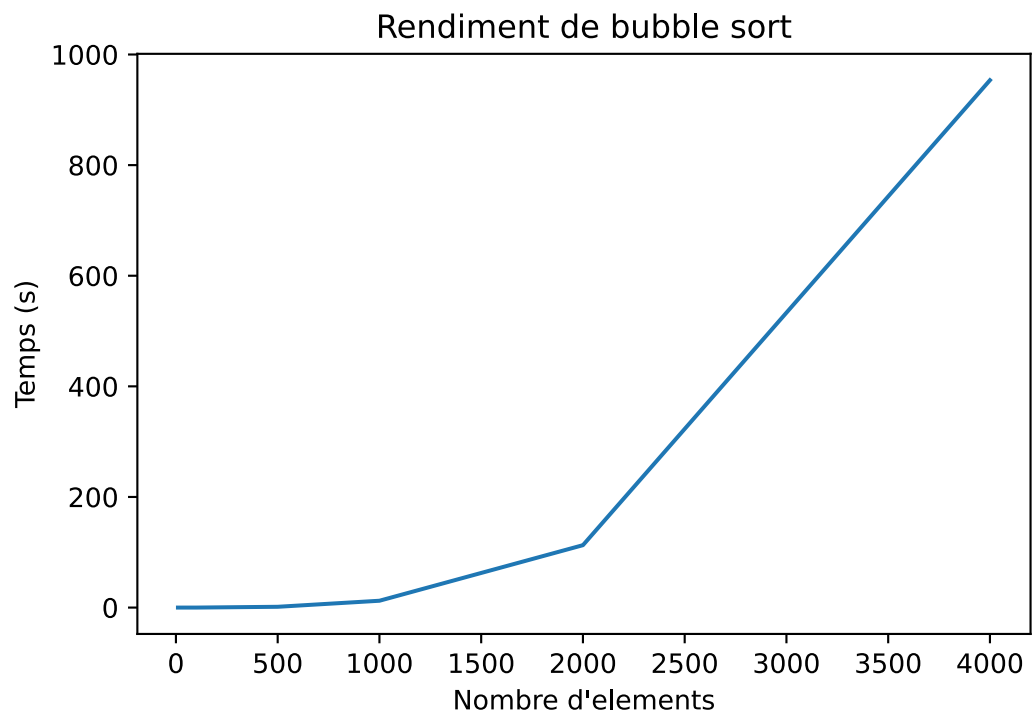
5.1. Com funciona?

5.2. Pseudocodi

5.3. Implementació

```
1  #!/usr/bin/env python3
2  import utils
3
4  def sort(array):
5      while True:
6          for i in range(len(array)-1):
7              if array[i] > array[i+1]:
8                  array[i], array[i+1] = array[i+1], array[i]
9                  break
10
11         # aquest else s'executa si el bloc anterior s'ha
12         # executat sense cap break, és a dir, si mai es compleix la
13         # condició array[i] > array[i+1]
14         else:
15             break
16     return array
17
18 if __name__ == "__main__":
19     array = utils.numbers()
20     print(sort(array))
```

5.4. Rendiment



6. Shell sort

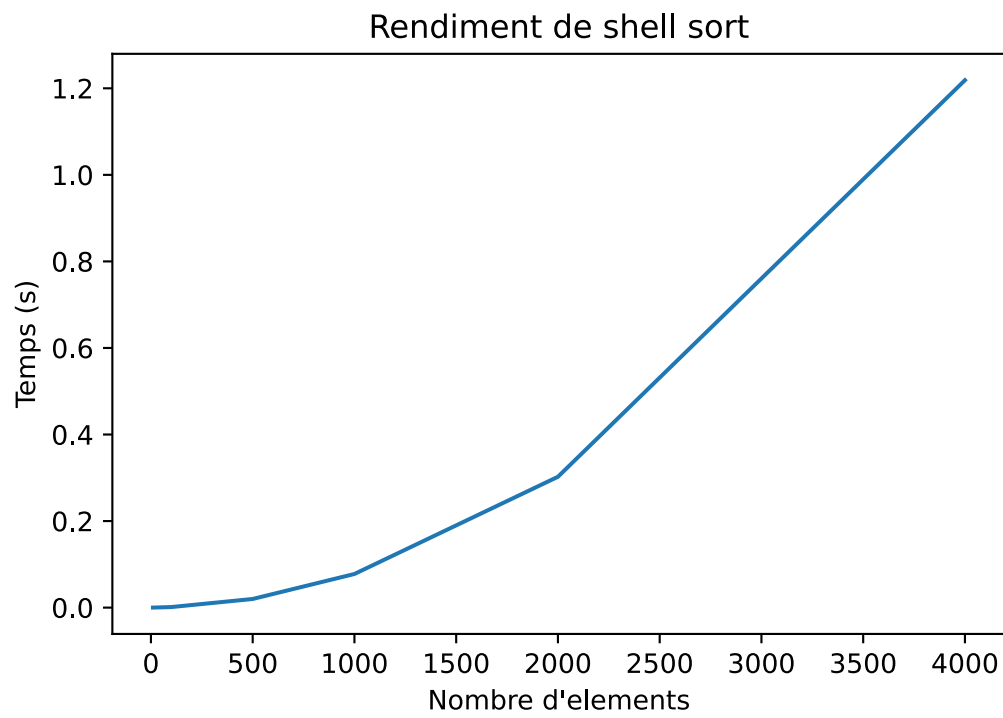
6.1. Com funciona?

6.2. Pseudocodi

6.3. Implementació

```
1  #!/usr/bin/env python3
2  import utils
3  import insertion
4
5  def sort(array):
6      step = len(array) // 2
7      while step != 0:
8          for offset in range(step):
9              sorted = insertion.sort(array[offset::step])
10             for i in range(len(sorted)):
11                 array[step*i + offset] = sorted[i]
12             step //= 2
13         return sorted
14
15 if __name__ == "__main__":
16     array = utils.numbers()
17     print(sort(array))
```

6.4. Rendiment



7. Quick sort

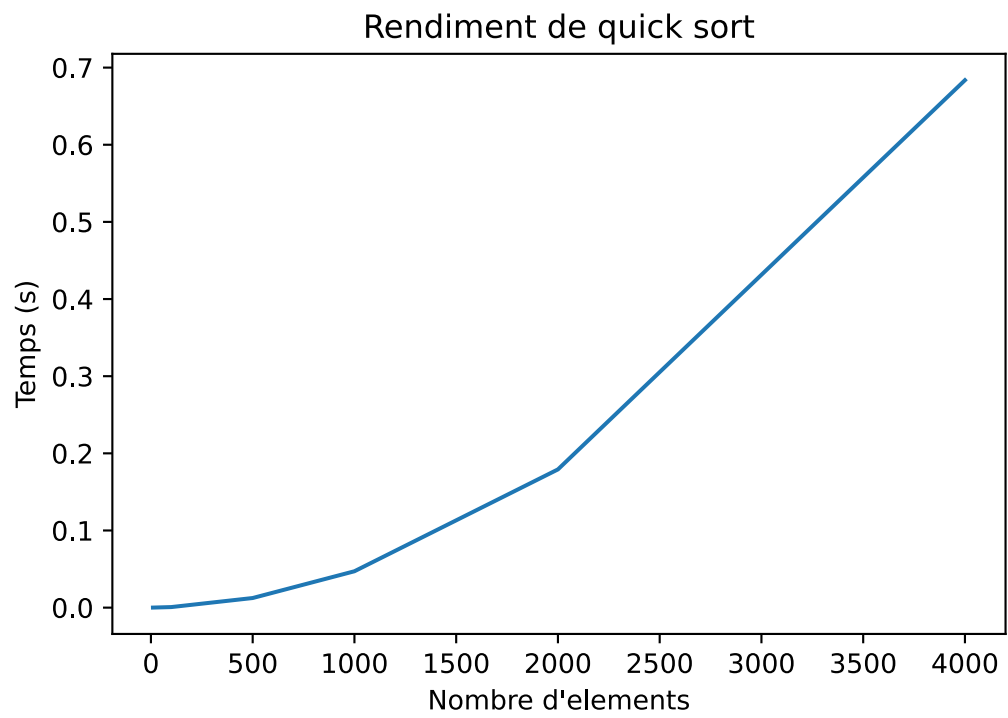
7.1. Com funciona?

7.2. Pseudocodi

7.3. Implementació

```
1  #!/usr/bin/env python3
2  import utils
3
4  def sort(array):
5      if len(array) < 2:
6          return array
7
8      pivot = array[0] # agafo el primer element com a pivot,
9      pot ser qualsevol
10     lower, higher = [], []
11
12     for i in array[: -1]:
13         if i < pivot:
14             lower.append(i)
15         else:
16             higher.append(i)
17
18     lower = sort(lower)
19     higher = sort(higher)
20
21     return lower + [pivot] + higher
22
23 if __name__ == "__main__":
24     array = utils.numbers()
25     print(sort(array))
```

7.4. Rendiment



8. Merge sort

8.1. Com funciona?

8.2. Pseudocodi

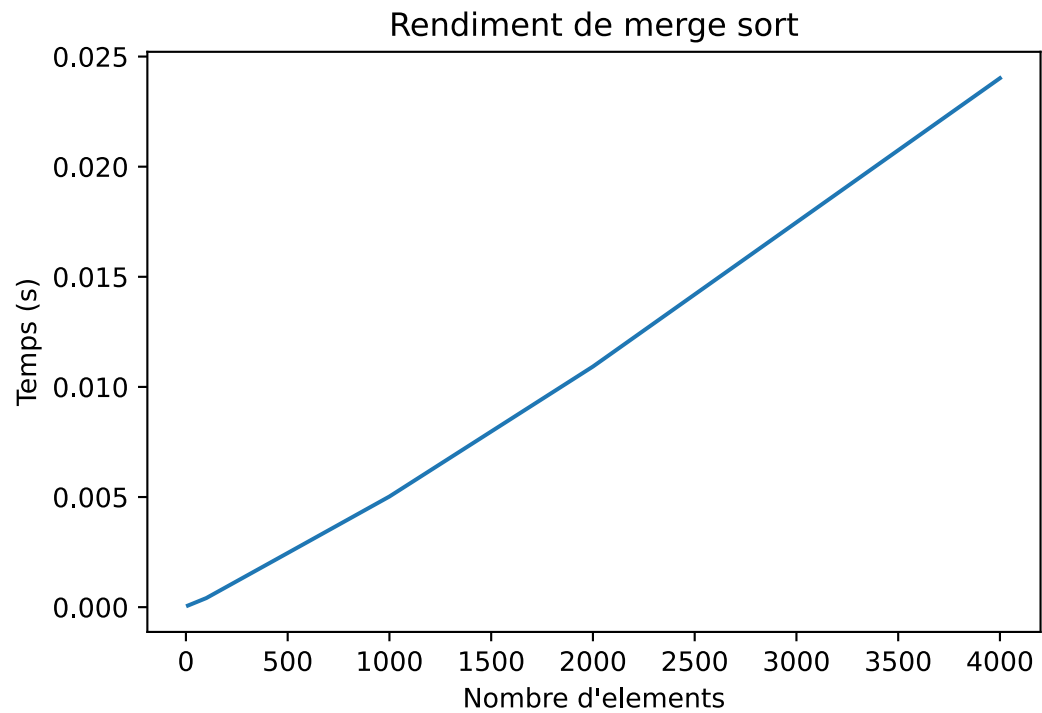
8.3. Implementació

```
1  #!/usr/bin/env python3
2  import utils
3
4  def sort(array):
5      if len(array) < 2:
6          return array
7      mid = len(array)//2
8      left = sort(array[:mid])
9      right = sort(array[mid:])
10
11     merged = []
12     l = 0
13     r = 0
14
15     while l < len(left) and r < len(right):
16         if left[l] < right[r]:
17             merged += [left[l]]
18             l += 1
19         else:
20             merged += [right[r]]
21             r += 1
22
23     if l < len(left):
24         merged += left[l:]
25     if r < len(right):
26         merged += right[r:]
27
28     return merged
29
```



```
30 if __name__ == "__main__":  
31     array = utils.numbers()  
32     print(sort(array))
```

8.4. Rendiment



9. Comparació dels algoritmes

