Algorithms in Python Programming

Week 10A: Sorting algorithms in Python

Sorting algorithms:

There are many sorting algorithms that widely implemented and used in programming languages.

- Selection sort
- Insertion sort
- Merge sort, bubble sort, heap sort, quick sort and **timsort.**.

As you know already, we use **sort()** and **sorted()** functions in Python for arranging elements in ascending or descending order.

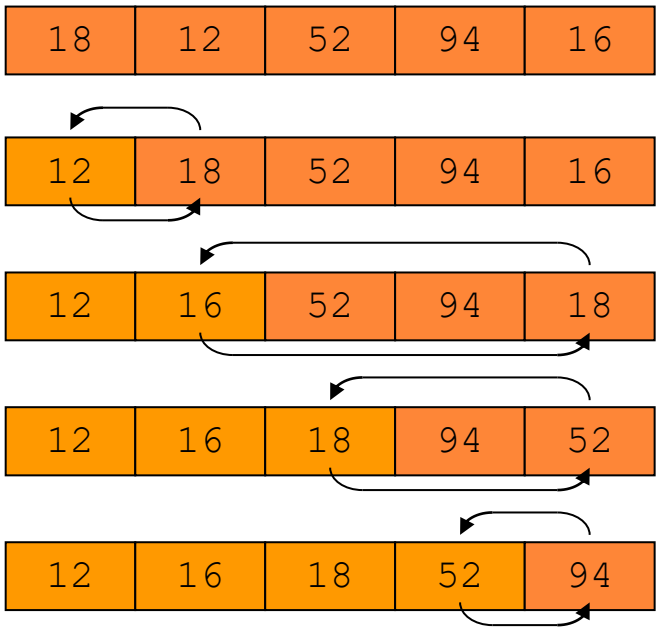
What sorting technique does Python use?
Python uses **Timsort** which is derived from **merge sort** and **insertion sort** to perform sorting on many kinds of **real-world data**.

Sorting algorithms: Selection sort

The selection sort algorithm gets its name, because it works on the successive selection of items in a particular order.

The algorithm works as follows:

- Find the minimum value in the list
- Swap it with the value in the first position
- Repeat the steps above for the remainder of the list (starting at the second position and advancing each time)



This is the order of elements, before the selection sort commences.

It finds 12 as the smallest number, and swaps with index 0.

It finds 16 as the smallest number, and swaps with index 1.

It finds 18 as the smallest number, and swaps with index 2.

It finds 52 as the smallest number, and swaps with index 3. The array is now sorted.

Selection sort: Python program

sort1_selection.py

```
1  def selectionSort(l1):
2      for i in l1:
3          mini = l1.index(i)
4          for j in range(mini+1,len(l1)):
5              if l1[j]<l1[mini]:
6                  swap = l1[j]
7                  l1[j]=l1[mini]
8                  l1[mini] =swap
9      return l1
10 #main program
11 list1 = [24,7,64,41,36,78,93,65,55,12]
12 print(selectionSort(list1))
```

Shell

```
Python 3.7.9 (bundled)
>>> %Run sort1_selection.py
 [7, 12, 24, 36, 41, 55, 64, 65, 78, 93]
```

sort1_selection2.py

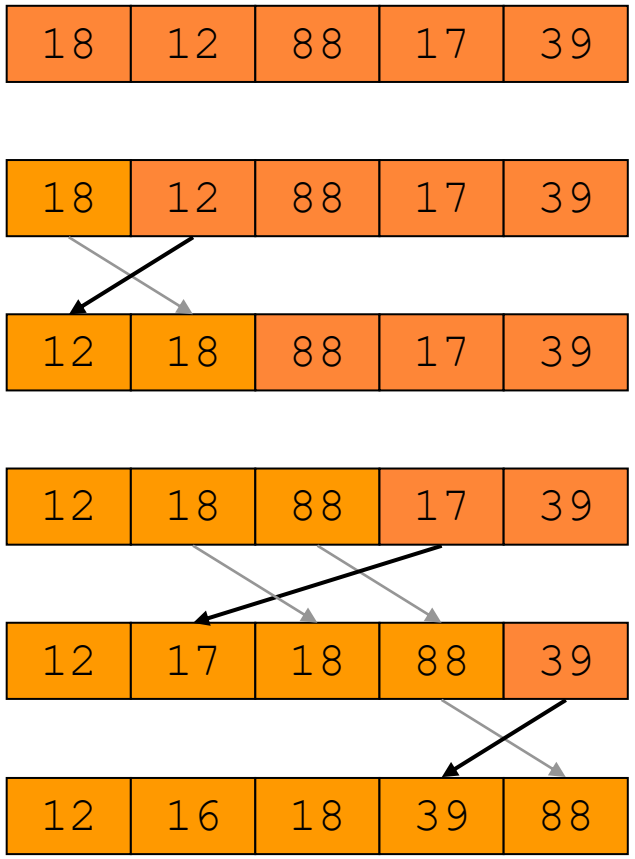
```
1  def selectionSort(l1):
2      for i in l1:
3          mini = l1.index(i)
4          for j in range(mini+1,len(l1)):
5              if l1[j]<l1[mini]:
6                  swap = l1[j]
7                  l1[j]=l1[mini]
8                  l1[mini] =swap
9      return l1
10 #main program
11 list1 = ['dd','b','xy','c','ab','mn','cz','y','az','wy']
12 print(selectionSort(list1))
```

Shell

```
Python 3.7.9 (bundled)
>>> %Run sort1_selection2.py
 ['ab', 'az', 'b', 'c', 'cz', 'dd', 'mn', 'wy', 'xy', 'y']
>>>
```

Python's string type uses the Unicode Standard **for representing characters**, which lets Python programs work with all these different possible characters. Unicode **(https://www.unicode.org/)** is a specification that aims to list every character used by human languages and give each character its own unique code.

Sorting algorithms: Insertion sort



This is the list of elements, before the insertion sort commences.

18 is introduced as the first item in the sorted sublist

12 is introduced into the sublist; 18 must make way for it.

88 is introduced into the sublist; it does not need to be moved.

17 is introduced into the sublist; 18 and 88 must move to make way.

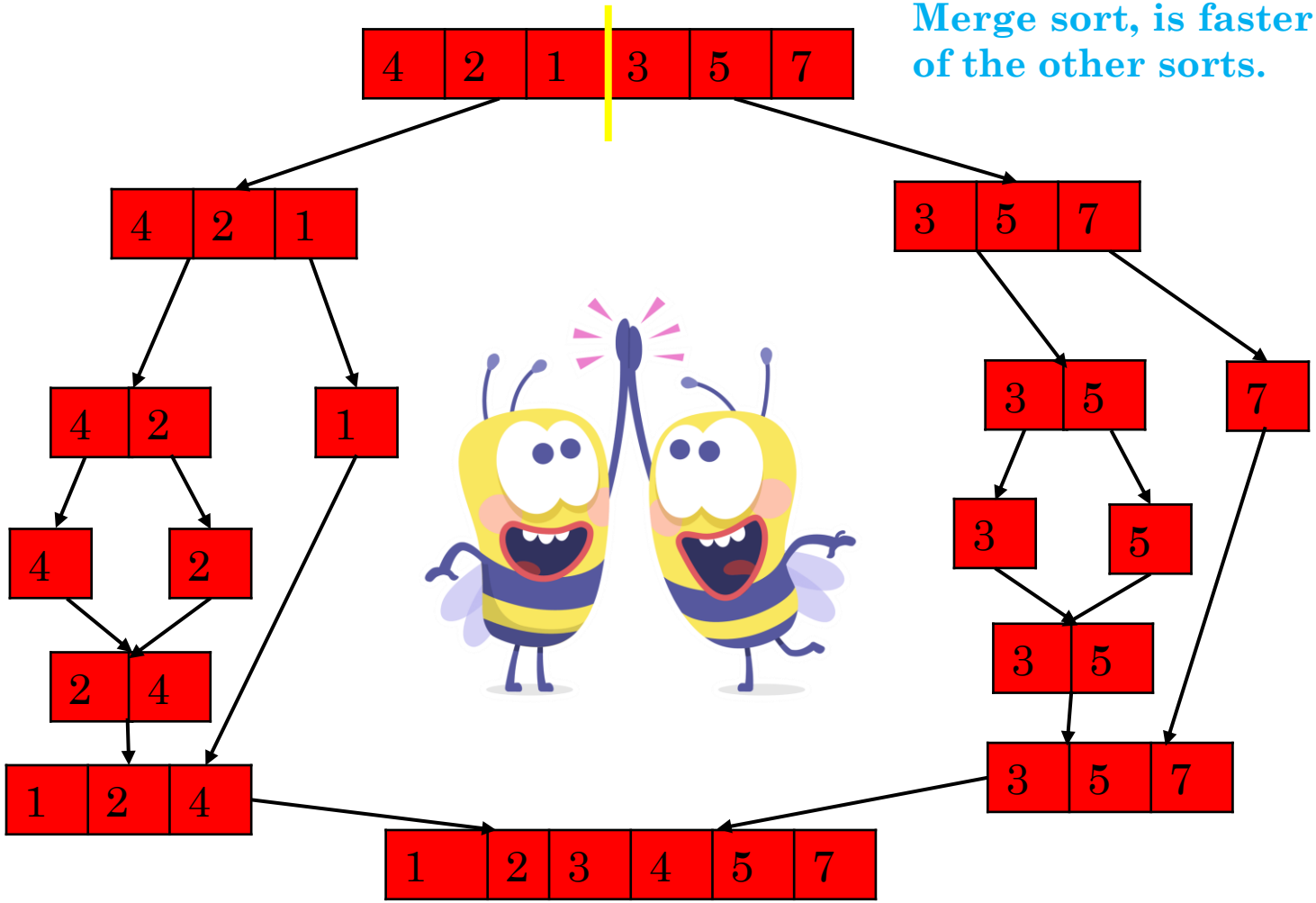
39 is introduced into the sublist; 88 must make way.

Insertion sort is effective on small lists

Sorting algorithms: Merge sort

- It is an comparison based sorting algorithm
- It split the list into sublists until sublist size is 1.
- Then merges those sublists to produce a sorted list.

Merge sort, is faster than most of the other sorts.



What sorting algorithm does Python use?

Python uses **Timsort** which is derived from **merge sort** and **insertion sort** to perform on many kinds of **real-world data**.

If the list size is small, then Python uses insertion sort to perform sorting.

Check the link :→ **self study:**
https://dev.to/brandonskerritt/timsort-the-fastest-sorting-algorithm-you-ve-never-heard-of-2ake

Pseudo code

It is a simple language that explain the description of steps (mechanics of the code) for a program or algorithm or how the system works or to be developed.

Python program

```
1 age = int(input("Enter your age:"))
2
3 if age>18:
4     print("Eligible for Army training")
5 else:
6     print("NOT eligible for Army training")
7
```

Pseudo code

Step 1: Prompt user to get age in number

Step 2: If age is above 18 then
display “eligible for army training”

else then
display “not eligible for army training”

Step 3: End of the task

x = 0 → **set x as/to 0**

f1 = open("asd.txt","w") → **create a new text file asd.txt for writing/storing**

Are description listed in the slides 3 and 5 part of pseudo code?

Task: Write pseudo code for Timsort() Python