Algorithms

STL Sorting

Mostafa S. Ibrahim
Teaching, Training and Coaching for more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

Stl Sorting

- As you know, we can use the built-in sorting function:
  - sort(vec.begin(), vec.end())
- What is the internally used algorithm?
  - The problem is each algorithm has some pros and cons.
- STL uses IntroSort which is a **hybrid** of QuickSort, HeapSort and InsertionSort.
  - For very small arrays ⇒ Use Insertion sort
  - Otherwise, the default algorithm is QuickSort
  - If QuickSort seems like it won't perform well, it switches to HeapSort
- Python uses Timsort which is a **hybrid** of a hybrid stable sorting algorithm, *derived* from merge sort and insertion sort, designed to perform well on *many kinds of real-world data*

Stl Sorting: Wrong usage

- The code below using C++ std::Sort will RTE: Why?

```
11 bool compare_right(int a, int b) {
12     return a < b;
13 }
14 bool compare_wrong(int a, int b) {
15     return a <= b;
16 }
17
18 int main() {
19     // 10^7 elements of value 5
20     vector<int> v(10000000, 5);
21     // It will take much time, then RTE!
22     sort(v.begin(), v.end(), compare_wrong);
23     // It takes reasonable time with compare_right
24     //sort(v.begin(), v.end(), compare_right);
25     cout << "bye\n"<<flush; // 90% won't be printed
```

Stl Sorting: Logical Ordering

- If you will provide a comparison function (e.g. for your Employee class), it must behave logically the same as E1 < E2 NOT E1 <= E2
- If this is not satisfied, in some cases/implementations,
  - your code will go in an infinite recursion, then RTE
- **Current, Internally**, here is what is expected
  - If compare (E1, E2) = True, then E1 < E2
    - Then compare (E2, E1) = must be false
  - If compare (E1, E2) = False, there are 2 cases
    - If comparing (E2, E1) = True, then E2 < E1
    - If comparing (E2, E1) = False, then E1 == E2 (no one is smaller than another)
  - Don’t miss up with the current STL implementation!
- STL allows us to compare whatever **objects** that are **comparable**
  - Built-in (int, double, string) or user-defined (Employee class)

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."