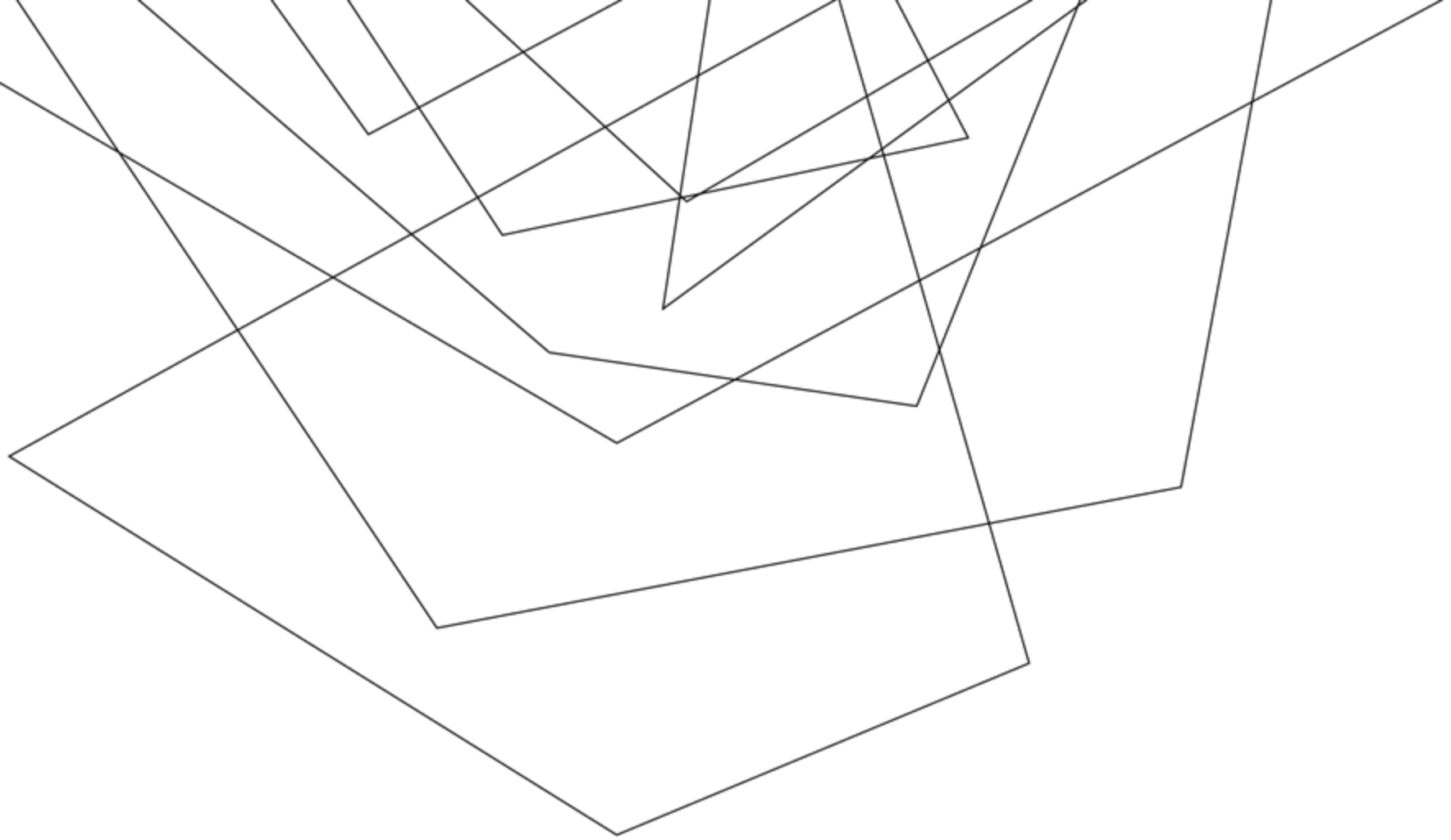


CMPT 225 D100 LAB10

TA

TOPICS FOR TODAY

Sorting

- Insertion sort
- Quicksort
- Heap sort

Comparison Sorting

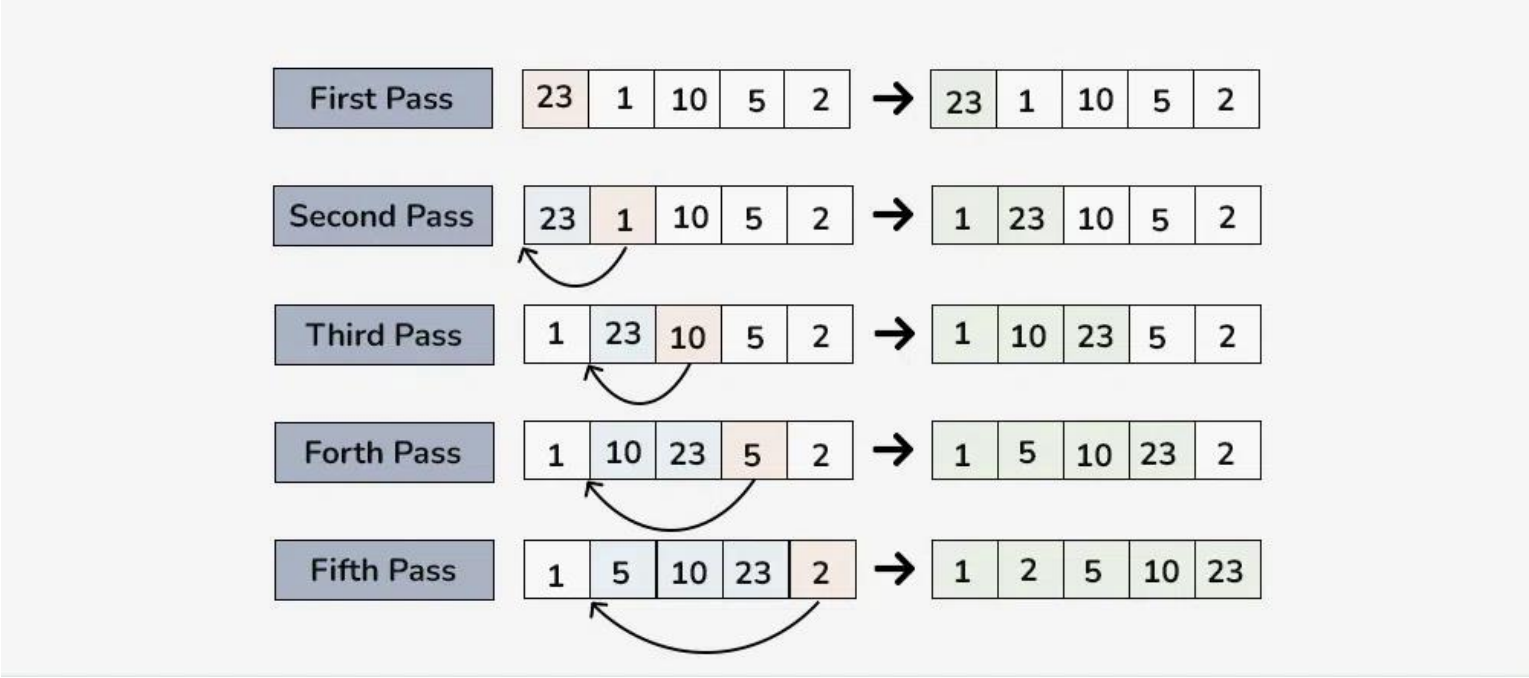
Sorting algorithms need to using comparisons in order to be general enough to sort all types of data

In C++ the STL implement insertion sort, quicksort and heapsort for this purpose. Each with their own pros and cons.

	In-place	Stable	Average Time complexity	Worst-Case Time complexity
Insertion Sort	Yes	Yes	$O(n^2)$	$O(n^2)$
Quicksort	Yes	No	$O(n \log n)$	$O(n^2)$
Heap Sort	Yes	No	$O(n \log n)$	$O(n \log n)$

Insertion Sort

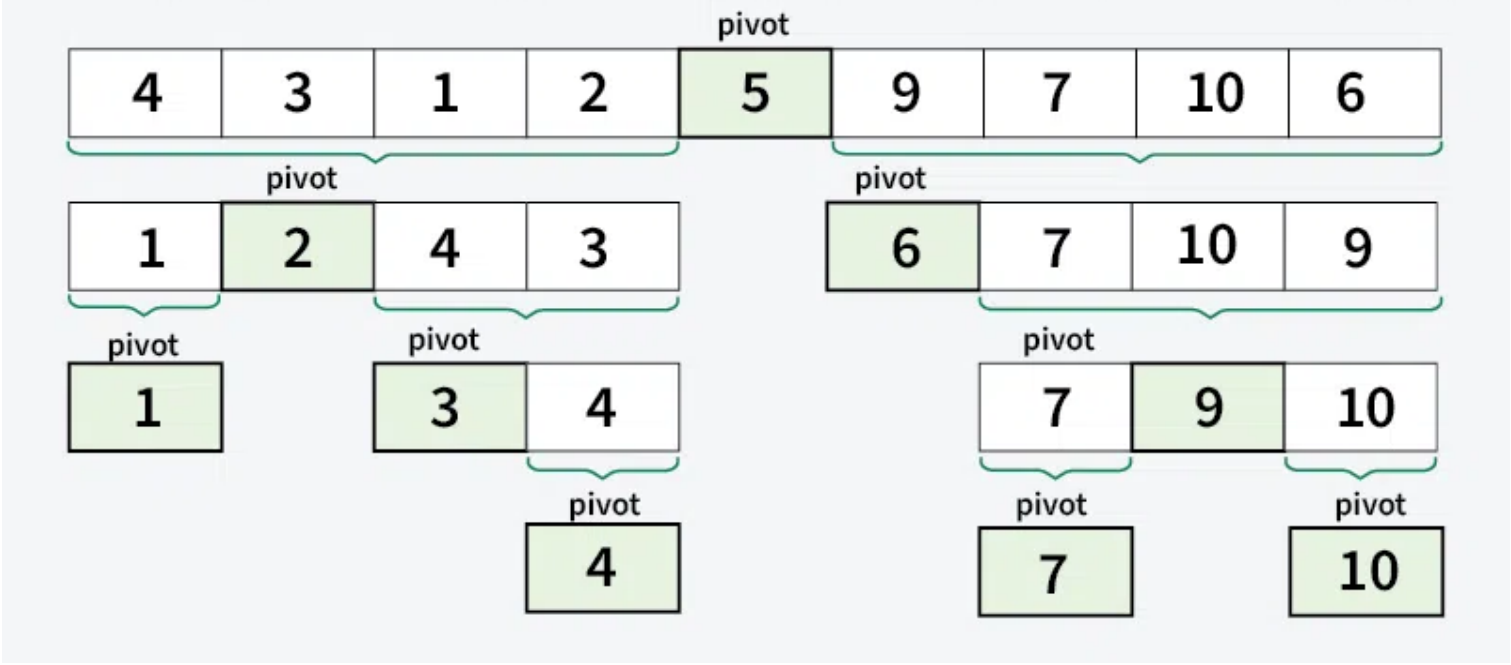
Insertion sort builds a final sorted array one item at a time by repeatedly taking an element from **the unsorted section** and inserting it into the correct position in **the sorted section**



https://www.geeksforgeeks.org/dsa/insertion-sort-algorithm/

Quicksort

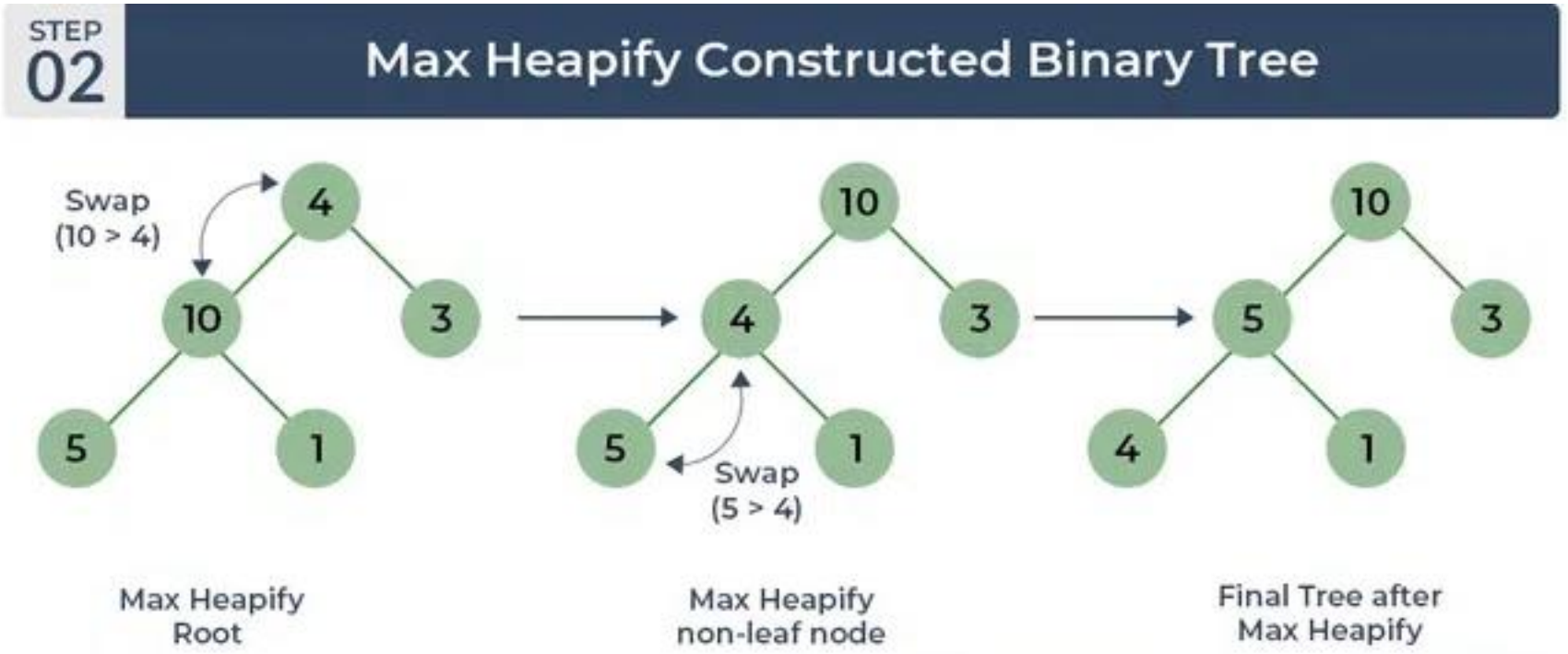
Quicksort employs a **divide-and-conquer** strategy, working by selecting a "**pivot**" element from an array or list, partitioning the other elements based on whether they are smaller or larger than the pivot, and then recursively sorting those partitions



https://www.geeksforgeeks.org/dsa/quick-sort-algorithm/

Heap Sort

Heap sort organizes elements into a **binary heap** and then repeatedly extracts the **largest element** to build a sorted array.



https://www.geeksforgeeks.org/c/c-program-for-heap-sort/

Exercise

- The goal today is to complete a test each of these algorithms for their best use case.
- Download the **sorting.cpp** file
  - Read the all functions to understand how they work. There is code for `insertionSort`, `quickSort` and `heapSort` with their necessary helper functions. You should understand this code.
  - Complete the code in **sorting.cpp**
    - Look for the comment: // YOUR WORK IS HERE
  - You will have to fill the three vectors of size n = 100000 so that
    - firstType gives Insertion Sort the best time of the three algorithms
    - secondType gives quicksort the best time of the three algorithms
    - thirdType gives heapsort the best time of the three algorithms

- Since the array is large, it will take a minute or so to run the main.

Expected output

```
+----------------+----------------+-----------+
| Algorithm      | Input Type     | Time      |
+----------------+----------------+-----------+
| Insertion Sort | First          |      0 ms |
| Quicksort      | First          |  10504 ms |
| Heapsort       | First          |     27 ms |
```

```
+----------------+----------------+-----------+
| Algorithm      | Input Type     | Time      |
+----------------+----------------+-----------+
| Insertion Sort | Second         |   8790 ms |
| Quicksort      | Second         |     13 ms |
| Heapsort       | Second         |     24 ms |
```

```
+----------------+----------------+-----------+
| Algorithm      | Input Type     | Time      |
+----------------+----------------+-----------+
| Insertion Sort | Third          |  18090 ms |
| Quicksort      | Third          |  10801 ms |
| Heapsort       | Third          |     18 ms |
```

Note:

- 1 second = 1000 milliseconds
- This program took 48 seconds to complete, it can be slow
- You should replace “first”, “second” and “third” with more descriptive titles