Introsort- C++'s Sorting Weapon

Program Name- Introsort

Project Category- Sorting

Programming Paradigm Used- Quicksort, Heapsort, Insertion Sort

Pre-requisites-

Quicksort- http://quiz.geeksforgeeks.org/quick-sort/
Heapsort- http://quiz.geeksforgeeks.org/heap-sort/
Insertion Sort- http://quiz.geeksforgeeks.org/insertion-sort/

Explanation-

We will explain this algorithm through the below questions one by one.

What is Introsort?-

Simply putting, it is the best sorting algorithm around. It is a hybrid sorting algorithm, which means that it uses more than one sorting algorithms as a routine.

Which sorting algorithms introsort uses as its routine?-

Introsort being a hybrid sorting algorithm uses three sorting algorithm to minimise the running term – Insertion sort, Heapsort and Quicksort.

What is the algorithm?-

Introsort firstly begins with quicksort and if the recursion depth goes more than a particular limit it switches to heapsort to avoid quicksort's worse case $O(N^2)$ time complexity. It also uses insertion sort when the number of elements to sort is quite less.
So firstly it creates a partition. Now from here three cases arises-

1) If the partition size is such that there is a possibility to exceed the maximum depth limit then the introsort switches to heapsort.
We define the maximum depth limit as – 2*log(N)

2) If the partition size is too small then quicksort decays to insertion sort.
We define this cutoff as 16 (due to research). So if the partition size is less than 16 then we will do insertion sort.

3) If the partition size if under the limit and not too small (i.e- between 16 and 2*log(N)) , then it performs a simple quicksort.

Why it is better than simple quicksort or Why the need of introsort?-

Since quicksort can have a worse case $O(N^2)$ time complexity and it also increases the recursion stack space (O(logN) if tail recursion applied), so to avoid all these, we need to switch the algorithm from quicksort to another if there is a chance of worse case. So introsort solves this problem by switching to heapsort.

Also due to larger constant factor, quicksort can perform even worse than $O(N^2)$ sorting algorithm when N is small enough. So it switches to insertion sort to decrease the running time of sorting.

Also if a bad pivot-selection is done then the quicksort does no better than the bubble-sort.

Why insertion sort is used (and not bubble sort etc)?

Insertion sort offers two advantages-
1) It has a good locality of reference
2) It is an adaptive sorting algorithm, i.e- it outperforms all the other algorithms if the array elements are partially sorted.
3) It is a known and established fact that insertion sort is the most optimal comparison-based sorting algorithm for small arrays.

Why heap sort is used (and not merge sort etc)?

This is solely because of the memory requirements. Merge sort requires O(N) space whereas heapsort is an in-place O(1) space algorithm.

Why heap sort is not used in place of quicksort when the partition size is under the limit ?-

This question is same as why quicksort generally outperforms heapsort ?

The answer is -

Although heapsort also being O(NlogN) in average as well as worse case and O(1) space also, we still don't use it when the partition size is under the limit because the extra hidden constant factor in heapsort is quite larger than that of quicksort.

*Why the cut-off for switching from quick sort to insertion sort is 16 and that of switching from quick sort to heap sort is 2*logN ?-*

These values are chosen empirically as an approximate because of various tests and researches conducted.

Is Introsort stable ?-

Since quicksort is also not stable so introsort is also not stable.

Time Complexity-

1) Best Case - O(NlogN)
2) Average Case- O(NlogN)
3) Worse Case- O(NlogN)
where,
N = number of elements to be sorted

Space Complexity

Just like- quicksort it can have O(log N) auxiliary recursion stack space.

Points to Note-

Introsort is also called as "Introspective sort"

References-

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