Lớp: ATTN2024 - Đinh Quốc Thịnh

BÁO CÁO KẾT QUẢ THỬ NGHIỆM

Thời gian thực hiện: 25/02 – 04/03/2025

Sinh viên thực hiện:

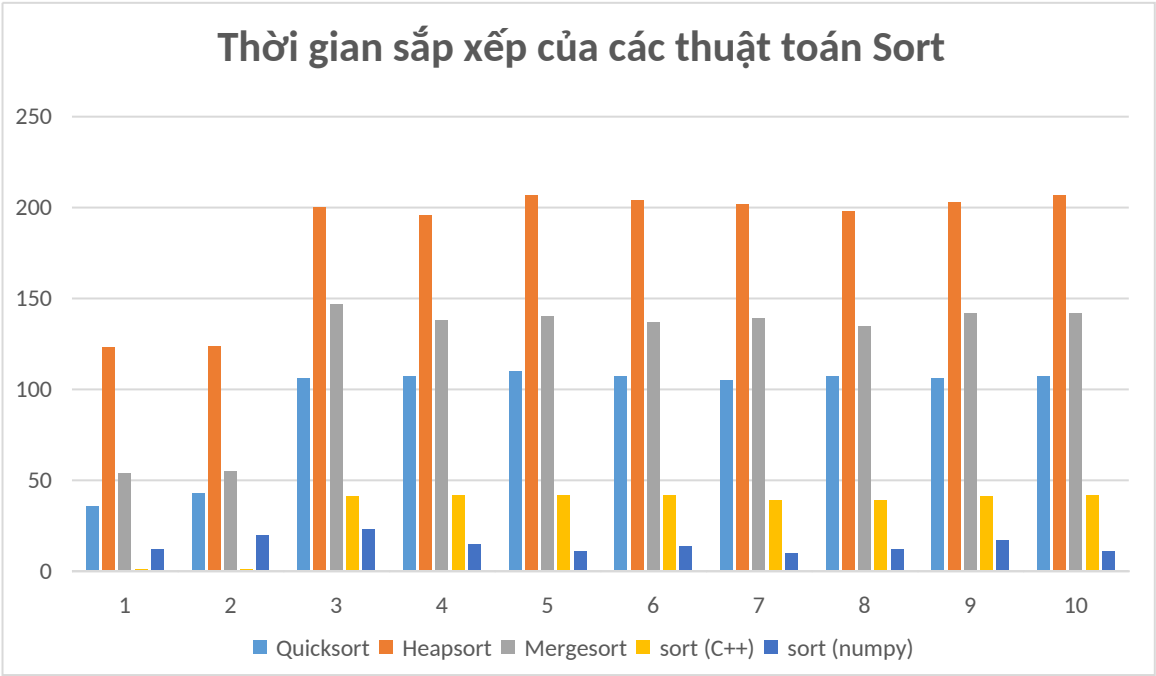
Nội dung báo cáo:

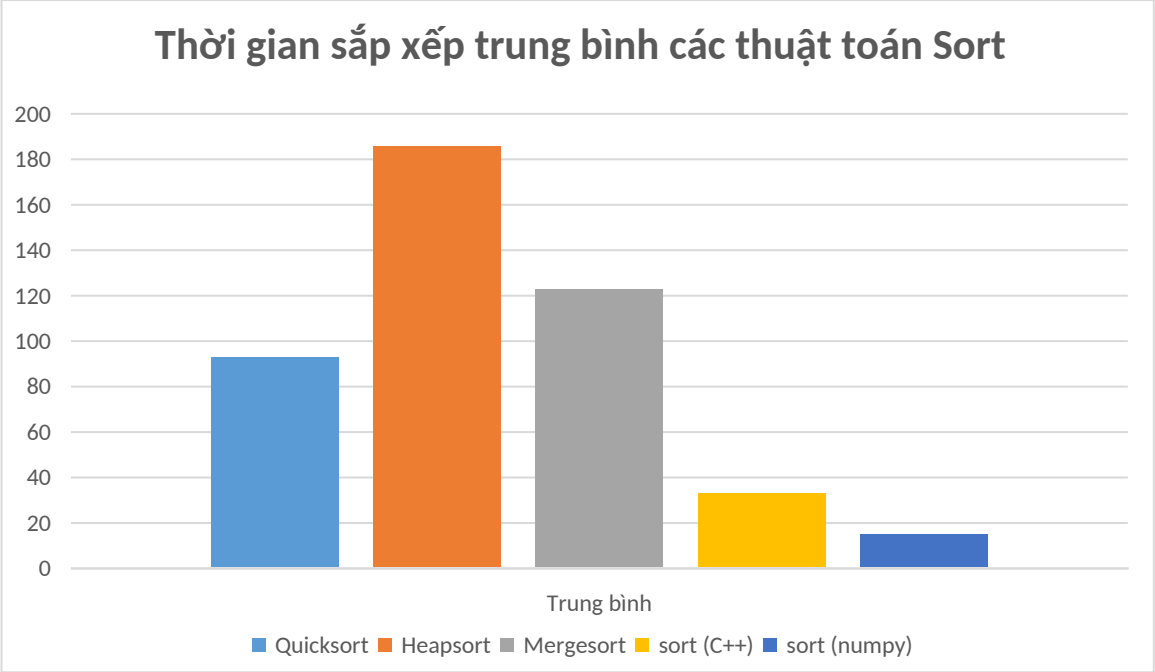
I. Kết quả thử nghiệm

1. Bảng thời gian thực hiện

Dữ liệu	Thời gian thực hiện (ms)				
	Quicksort	Heapsort	Mergesort	sort (C++)	sort (numpy)
1	36	123	54	1	12
2	43	124	55	1	20
3	106	200	147	41	23
4	107	196	138	42	15
5	110	207	140	42	11
6	107	204	137	42	14
7	105	202	139	39	10
8	107	198	135	39	12
9	106	203	142	41	17
10	107	207	142	42	11
Trung bình	93	186	123	33	15

2. Biểu đồ (cột) thời gian thực hiện





II. Kết luận:

- Tốc độ sắp xếp nhanh giảm dần của các loại thuật toán là: sort (numpy) > sort (c++) > QuickSort > MergeSort > HeapSort.

- Các thuật toán sắp xếp có sẵn trong các ngôn ngữ lập trình đã được thiết lập rất tối ưu và có tốc độ rất nhanh.

- QuickSort: Quicksort là thuật toán sắp xếp hiệu quả với độ phức tạp trung bình, hoạt động nhanh trong thực tế nhưng có thể chậm nếu chọn pivot không tốt.

- Heap Sort là thuật toán sắp xếp có độ phức tạp O(n log n) trong mọi trường hợp, không ổn định nhưng hoạt động tốt với dữ liệu lớn nhờ sử dụng cấu trúc cây heap.

- Merge Sort là thuật toán sắp xếp có độ phức tạp O(n log n) trong mọi trường hợp, ổn định, phù hợp với dữ liệu lớn nhưng tốn thêm bộ nhớ do sử dụng mảng phụ.

- Hàm Sort có sẵn trong thư viện <algorithm> là một trong những hàm sắp xếp nhanh và tối ưu, thường được cài đặt bằng **Introsort** (kết hợp QuickSort, HeapSort và InsertionSort).

- Hàm Sort (numpy) trong Python là hàm sắp xếp nhanh và hiệu quả, hỗ trợ nhiều thuật toán và hoạt động tốt với mảng số lớn.

=> Đối với mỗi loại thuật toán sắp xếp khác nhau sẽ có các trường hợp tốt nhất và tệ nhất, do đó ta nên dùng đúng loại thuật toán để tối ưu thời gian chạy, nếu không, ta có thể dùng các thuật toán có sẵn để dễ dàng sử dụng mà vẫn mang lại hiệu quả cao.