

Lớp: KHTN2021

## BÁO CÁO KẾT QUẢ THỬ NGHIỆM

Thời gian thực hiện: 11/03 – 16/03/2022

**Sinh viên thực hiện:** Huỳnh Phạm Đức Lâm

**Mã số sinh viên:** 21521050

### Nội dung báo cáo:

Cài đặt các thuật toán sắp xếp và chạy trên các bộ dữ liệu sinh ra ngẫu nhiên, ghi lại thời gian thực hiện trên từng bộ dữ liệu. Trong đó:

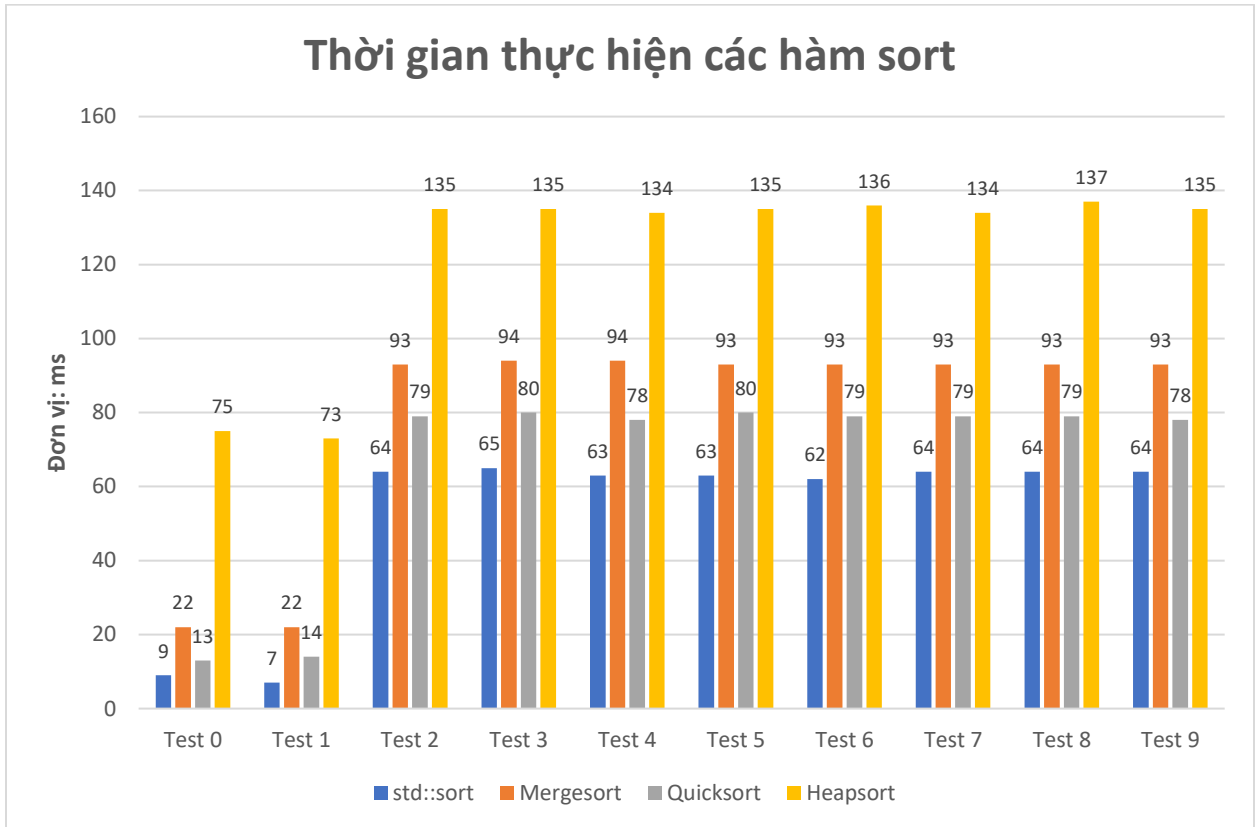
- Các thuật toán sắp xếp: Quicksort, Mergesort, Heapsort và `std::sort` (trong thư viện `algorithm`).
- Dữ liệu đầu vào: Gồm 10 file input từ 0 đến 9, mỗi file chứa 1 triệu số thực trong đoạn  $[0;1]$  được sinh ngẫu nhiên. Trong đó, input0 có thứ tự tăng dần, input1 có thứ tự giảm dần, các input còn lại có thứ tự ngẫu nhiên.
- Môi trường thử nghiệm:
  - CPU: i5-9400F @ 2.9GHz
  - Ngôn ngữ: C++
  - Compiler: GCC 11.2 with O3 optimization

### I. Kết quả thử nghiệm:

#### 1. Bảng thời gian thực hiện

Input	Thời gian thực hiện (ms)			
	<code>std::sort</code>	Mergesort	Quicksort	Heapsort
0	9	22	13	75
1	7	22	14	73
2	64	93	79	135
3	65	94	80	135
4	63	94	78	134
5	63	93	80	135
6	62	93	79	136
7	64	93	79	134
8	64	93	79	137
9	64	93	78	135
Trung bình	52.5	79	65.9	122.9

## 2. Biểu đồ thời gian thực hiện:



## II. Kết luận:

- Theo thời gian trung bình thì ta có: `std::sort` < Quicksort < Mergesort << Heapsort. Quicksort và `std::sort` chênh lệch không đáng kể cả ở test đã sắp xếp và các test ngẫu nhiên. Heapsort có thời gian thực thi chậm nhất (gấp 2.5 lần so với `std::sort`).
- Thời gian chạy ở các test sắp xếp nhanh hơn đáng kể so với các test ngẫu nhiên (2 đến 7 lần).
- Với việc sử dụng Introsort (Quicksort với đoạn lớn, Selection sort với đoạn nhỏ, Heapsort khi đến các depth cao) và chỉ cần import thư viện thì `std::sort` sử dụng dễ dàng và nhanh nhất.

## III. Thông tin chi tiết

<https://github.com/Darklul03/IT003-Sorting>