Lớp: IT003.P12.CTTN

BÁO CÁO KẾT QUẢ THỬ NGHIỆM

Sinh viên thực hiện:ĐÀO ĐÌNH AN

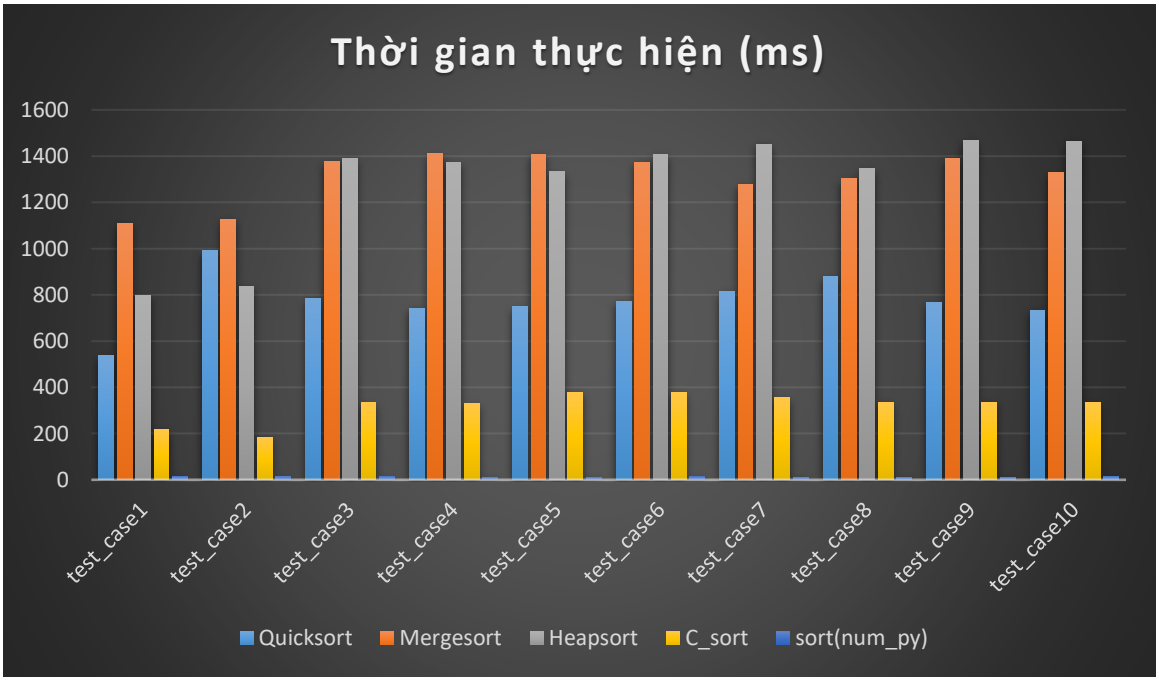
Nội dung báo cáo:Thực nghiệm các giải thuật sắp xếp

I. Kết quả thử nghiệm

1. Bảng thời gian thực hiện[1]

Dữ liệu	Thời gian thực hiện (ms)				
	Quicksort	Heapsort	Mergesort	sort (C++)	sort (numpy)
test_case 1	535.743	795.934	1108.7	217.084	12.392
test_case 2	991.33	836.157	1125.85	183.86	14.032
test_case 3	784.188	1390.12	1377.96	333.967	12.681
test_case 4	741.37	1373.8	1409.43	330.548	11.381
test_case 5	749.989	1331.45	1408.29	379.7	11.558
test_case 6	773.111	1405.84	1372.96	376.111	12.263
test_case 7	814.331	1451.82	1277.86	354.192	12.034
test_case 8	878.247	1345.9	1302.01	335.975	11.409
test_case 9	765.523	1467.68	1388.01	335.81	11.334
test_case 10	734.42	1464.19	1328.42	333.41	12.343
Trung bình	776.8282	1286.289	1309.947	328.0658	12.3427

2. Biểu đồ (cột) thời gian thực hiện



II. Kết luận:

- Hiệu năng cao nhất: sort (numpy) dẫn đầu với thời gian trung bình 12.3427 ms, nhanh vượt trội so với các thuật toán khác.

[1] Số liệu chỉ mang tính minh họa

- Hiệu năng tốt tiếp theo: sort (C++) đạt trung bình 328.0658 ms, nhanh hơn các thuật toán tự cài đặt.
- Hiệu năng trung bình: Quicksort có thời gian trung bình 776.8282 ms, tốt hơn Heapsort và Mergesort.
- Chậm nhất: Heapsort (1286.289 ms) và Mergesort (1309.947 ms) có hiệu năng thấp nhất.
    - ⇨ Nên xài hàm sort có sẵn trong c++ và python để tối ưu tốc độ chay.
    - ⇨ Nếu phải tự cài đặt thuật toán thì hãy ưu tiên quicksort(tối ưu pivot) để có thời gian chạy tối ưu .

III. Thông tin chi tiết – link github, trong repo gibub cần có:

Link github: https://github.com/nanad017/DSA_sort/