BÁO CÁO

I. Mô tả ý tưởng các hàm sinh dữ liệu

- **Ý tưởng chung:** sinh dữ liệu theo 4 xu hướng và lần lượt ghi vào 4 file txt các phần tử nằm trên 1 dòng và cách nhau bởi khoảng trắng.
- **Dữ liệu có xu hướng sắp xếp ngẫu nhiên:** phát sinh dữ liệu ngẫu nhiên có kích thước 10^6 mỗi phần tử có giá trị từ 0 đến 10^9 và ghi vào file NgauNhien.txt.
- **Dữ liệu có xu hướng gần như có thứ tự tăng dần:** phát sinh dữ liệu có kích thước 10^6 các phần tử tăng dần từ 0 đến 10^6 và hoán đổi ngẫu nhiên 10 lần trong dãy và ghi vào file DuLieuThuTu.txt.
- **Dữ liệu có thứ tự tăng dần:** phát sinh dữ liệu có kích thước 10^6 các phần tử tăng dần từ 0 đến 10^6 và ghi vào file DuLieuTangDan.txt.
- **Dữ liệu có thứ tự giảm dần:** phát sinh dữ liệu có kích thước 10^6 các phần tử giảm dần từ 10^6 đến 0 và ghi vào file DuLieuGiamDan.txt.

II. Đánh giá thuật toán

1. Selection Sort

- Độ phức tạp thời gian:
  - Tốt nhất: $O(n^2)$
  - Xấu nhất: $O(n^2)$
  - Trung bình: $O(n^2)$
- Độ phức tạp không gian: O(1)

2. Heap Sort

- Độ phức tạp thời gian:
  - Tốt nhất: O(nlogn)
  - Xấu nhất: O(nlogn)
  - Trung bình: O(nlogn)
- Độ phức tạp không gian: O(1)

3. Merge Sort

- Độ phức tạp thời gian:
  - Tốt nhất: O(nlogn)
  - Xấu nhất: O(nlogn)
  - Trung bình: O(nlogn)
- Độ phức tạp không gian: O(n)

4. Quick Sort

- Độ phức tạp thời gian:
  - Tốt nhất: O(nlogn) khi chọn pivot là trung vị
  - Xấu nhất: $O(n^2)$ khi chọn pivot là phần tử lớn nhất hoặc nhỏ nhất
  - Trung bình: O(nlogn)

- Độ phức tạp không gian: O(1), trong trường hợp xấu nhất là O(n)

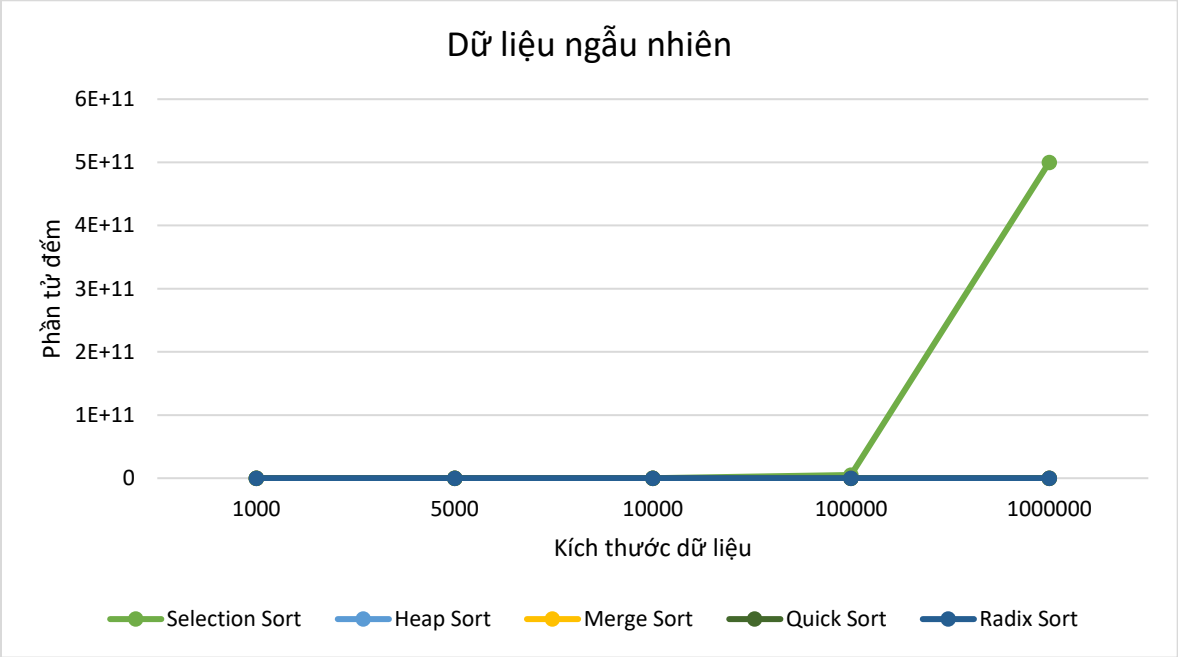
5. Radix Sort

- Độ phức tạp thời gian: O(n*k) trong đó n là số phần tử, k là số chữ số của phần tử lớn nhất. Độ phức tạp phụ thuộc vào giá trị của phần tử.
- Độ phức tạp không gian: O(n+k) trong đó n là số phần tử, k là số chữ số phần tử lớn nhất.

III. Thực nghiệm và nhận xét

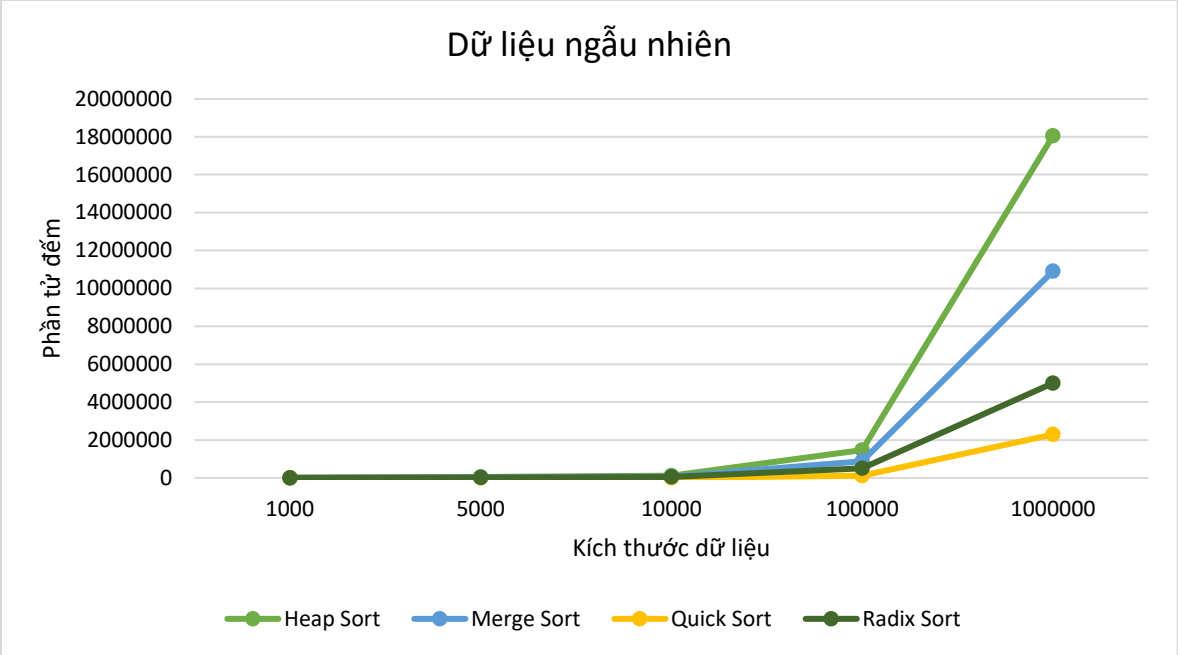
1. Dữ liệu ngẫu nhiên

	n=1000	n=5000	n=10000	n=100000	n=1000000
Selection Sort	499500	12497500	49995000	4999950000	499999500000
Heap Sort	8090	52043	114253	1474709	18047690
Merge Sort	4937	30015	65465	877956	10910853
Quick Sort	523	3180	6727	112117	2290859
Radix Sort	5000	25000	50000	500000	5000000



- Trong mọi kích thước dữ liệu thuật toán Selection Sort có thời gian chạy chậm nhất, thuật toán Quick Sort có thời gian chạy nhanh nhất.
- Theo thống kê các thuật toán phân thành 2 nhóm:

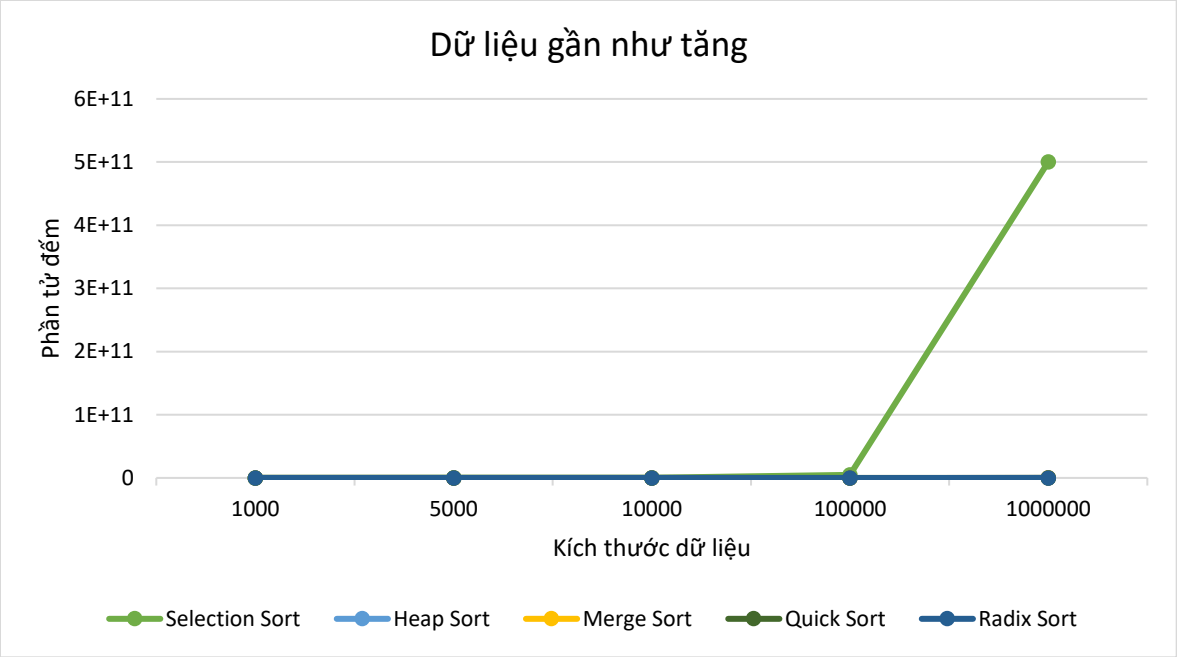
- Nhóm 1: Selection Sort với phần tử đếm tăng lên rất nhanh.
- Nhóm 2: Heap Sort, Merge Sort, Quick Sort, Radix Sort. Với với kích thước dữ liệu đủ nhỏ ($n \leq 10000$) phần tử đếm không có sự khác biệt đáng kể, tuy nhiên khi dữ liệu bắt đầu lớn ($n \geq 100000$) dần có sự khác biệt giữa Heap Sort với 3 thuật toán còn lại



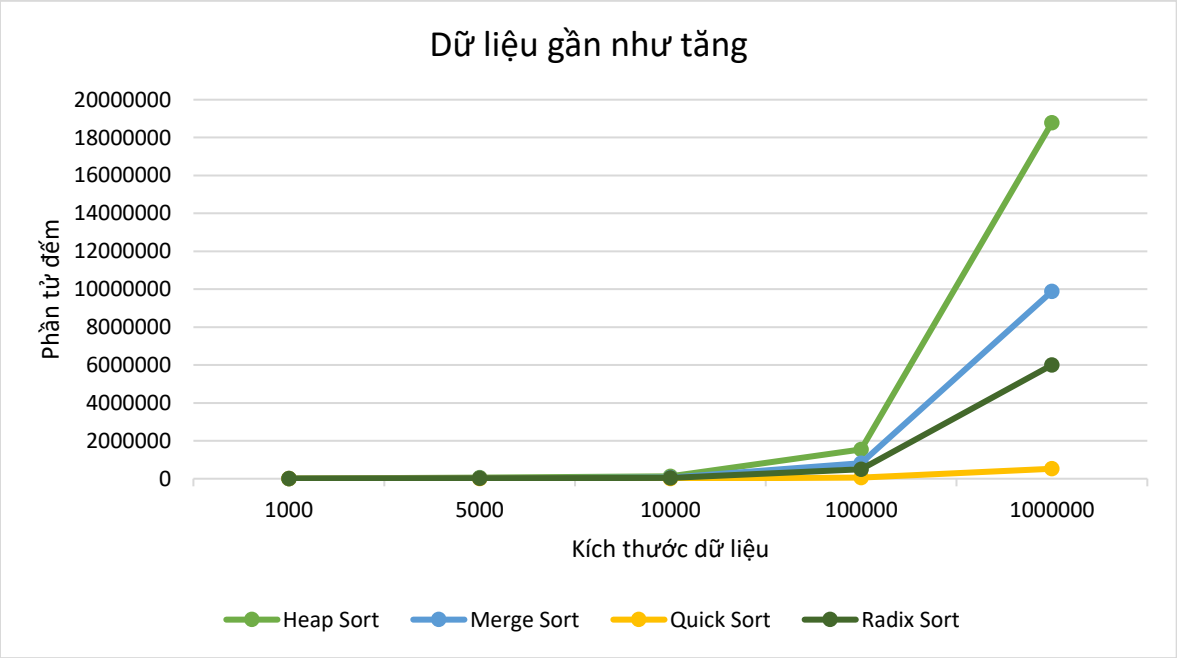
  - Thuật toán Quick Sort chạy nhanh nhất do pivot được chọn là trung vị
  - Heap Sort chạy chậm nhất trong nhóm này.
  - Merge Sort chậm do thao tác để sao chép qua mảng phụ trong quá trình sắp xếp.

2. Dữ liệu có xu hướng gần như tăng dần

	n=1000	**n=5000**	**n=10000**	**n=100000**	**n=1000000**
Selection Sort	499500	12497500	49995000	4999950000	499999500000
Heap Sort	8706	55524	121200	1550745	18788327
Merge Sort	4932	29804	64608	815024	9884992
Quick Sort	511	2952	5904	65535	524287
Radix Sort	5000	25000	50000	500000	6000000



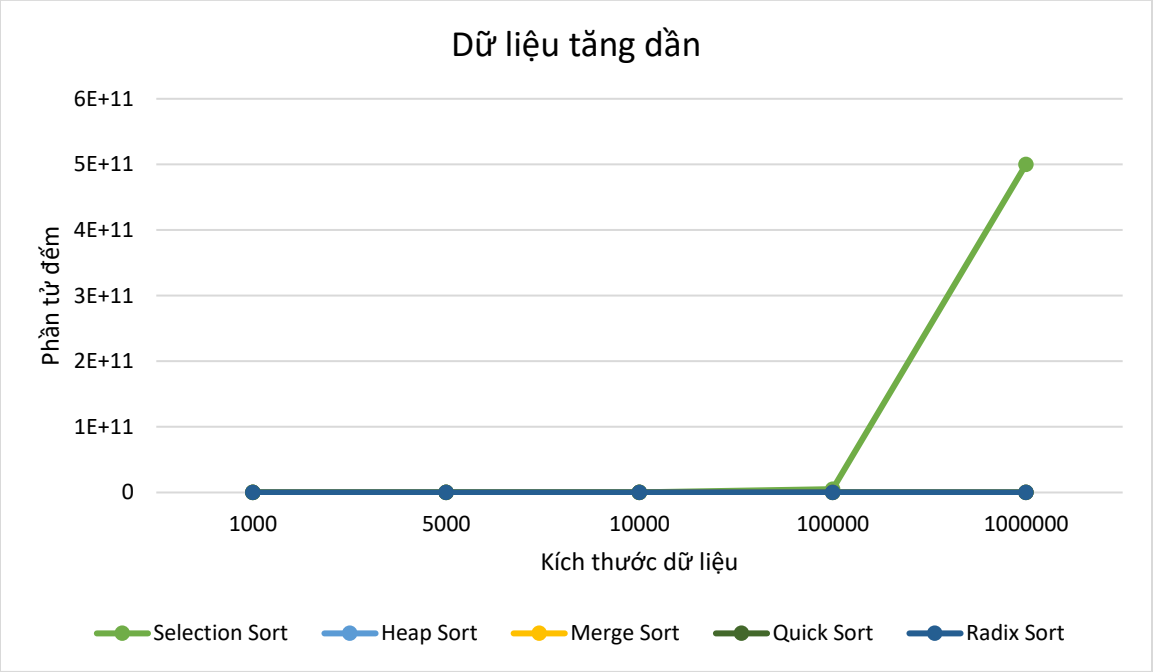
- Selection Sort là thuật toán chạy chậm nhất mặc dù mảng gần như tăng dần do vẫn duyệt qua từng cặp phần tử đôi một và thực hiện so sánh do đó số thao tác đếm gần như tăng lên theo số m



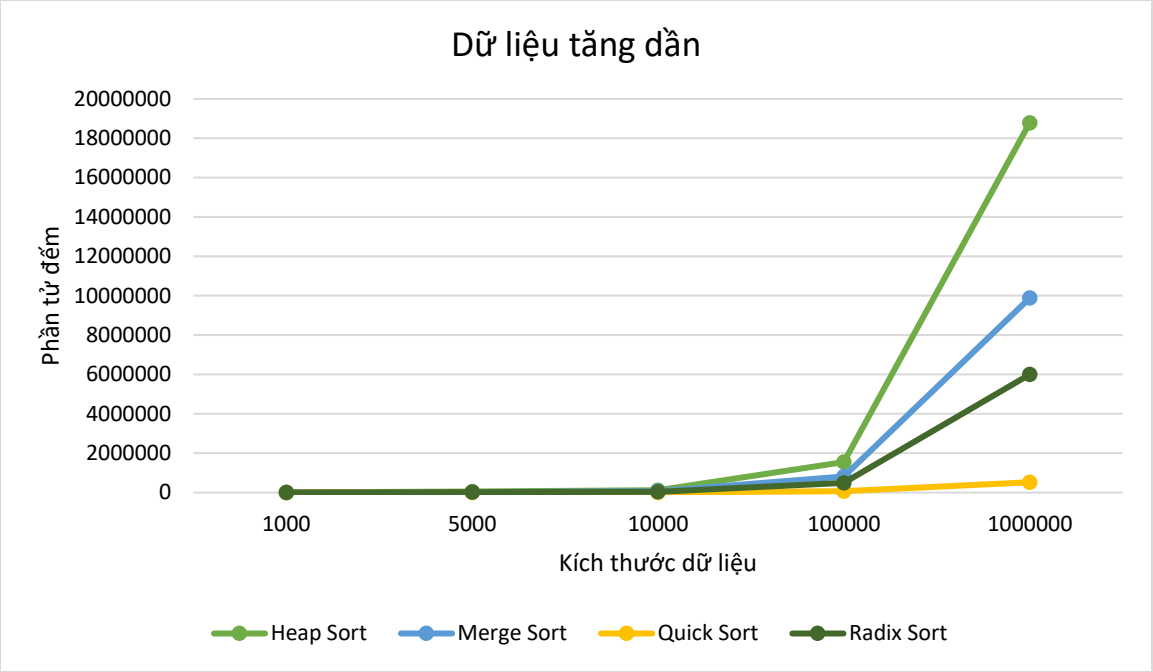
- Heap Sort chậm nhất vì phải mất nhiều thao tác khi gặp dữ liệu có kích thước lớn
- Quick Sort là thuật toán chạy nhanh nhất vì pivot được chọn là phần tử trung vị.

3. Dữ liệu tăng dần

	n=1000	n=5000	n=10000	n=100000	n=1000000
Selection Sort	499500	12497500	49995000	4999950000	499999500000
Heap Sort	8709	55933	121957	1550855	18787793
Merge Sort	4932	29804	64608	815024	9884992
Quick Sort	511	2952	5904	65535	524287
Radix Sort	3000	20000	40000	500000	6000000

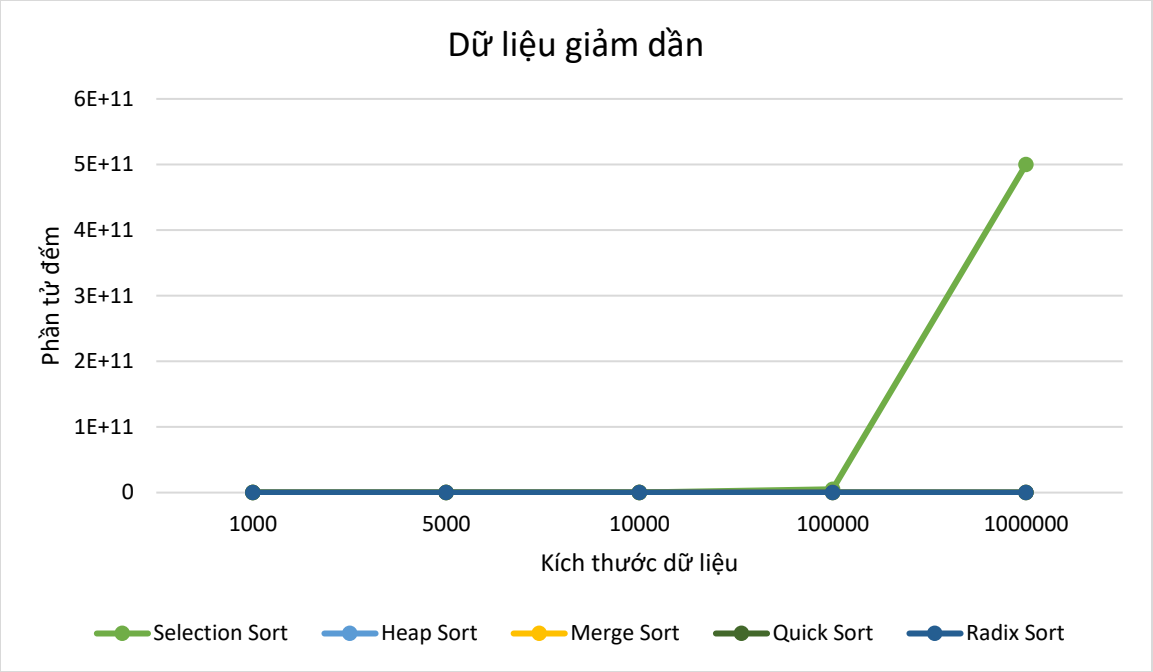


- Tương tự trường hợp dữ liệu có xu hướng gần như tăng dần, chiếm nhiều thời gian nhất vẫn là Selection Sort, thuật toán nhanh nhất vẫn là Quick Sort.

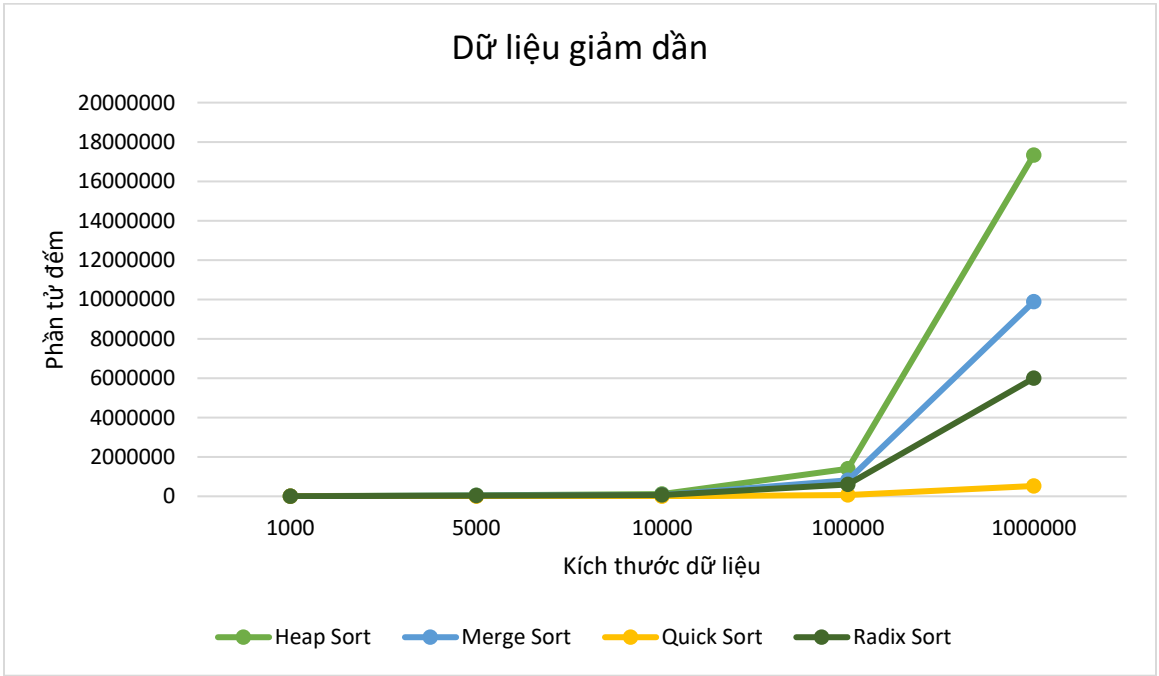


4. Dữ liệu giảm dần

	n=1000	n=5000	n=10000	n=100000	n=1000000
Selection Sort	499500	12497500	49995000	4999950000	499999500000
Heap Sort	7317	48437	106697	1397435	17333409
Merge Sort	4932	29804	64608	815024	9884992
Quick Sort	511	2952	5904	65535	524287
Radix Sort	6000	30000	60000	600000	6000000



- Thuật toán nhanh nhất vẫn là Quick Sort và chậm nhất vẫn là Selection Sort.
- Ở nhóm 2 thuật toán Merge Sort chạy chậm do tốn thao tác để sao chép qua mảng phụ trong quá trình sắp xếp. Heap Sort tốn nhiều thao tác để sắp xếp dữ liệu có kích thước lớn.



IV. Tài liệu tham khảo:

- Các thuật toán Selection Sort, Heap Sort, Quick Sort, Merge Sort, Radix Sort trên trang GeeksforGeeks (ngày tham khảo 26/12/2023):

- Selection Sort
- Heap Sort
- Quick Sort
- Merge Sort
- Radix Sort