BÁO CÁO THỰC HÀNH

**Môn học: Lập Trình Ứng Dụng Mạng**

**Buổi báo cáo: Lab 04**

**Tên chủ đề:** Multithread trong java

*GVHD: Trần Mạnh Hùng*

*Ngày thực hiện: 26/04/2024*

**THÔNG TIN CHUNG:**

*(Liệt kê tất cả các thành viên trong nhóm)*

Lớp: NT109.O21.MMCL

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Email** |
| 1 | Nguyễn Văn Anh Tuấn | 21522757 | 21522757@gm.uit.edu.vn |

1. **ĐÁNH GIÁ KHÁC:**

|  |  |
| --- | --- |
| **Nội dung** | **Kết quả** |
| Tổng thời gian thực hiện bài thực hành trung bình | 4 tiếng |
| Link Video thực hiện  *(nếu có)* |  |
| Ý kiến *(nếu có)*  + Khó khăn  + Đề xuất … | Không có |
| Điểm tự đánh giá | 10/10 |

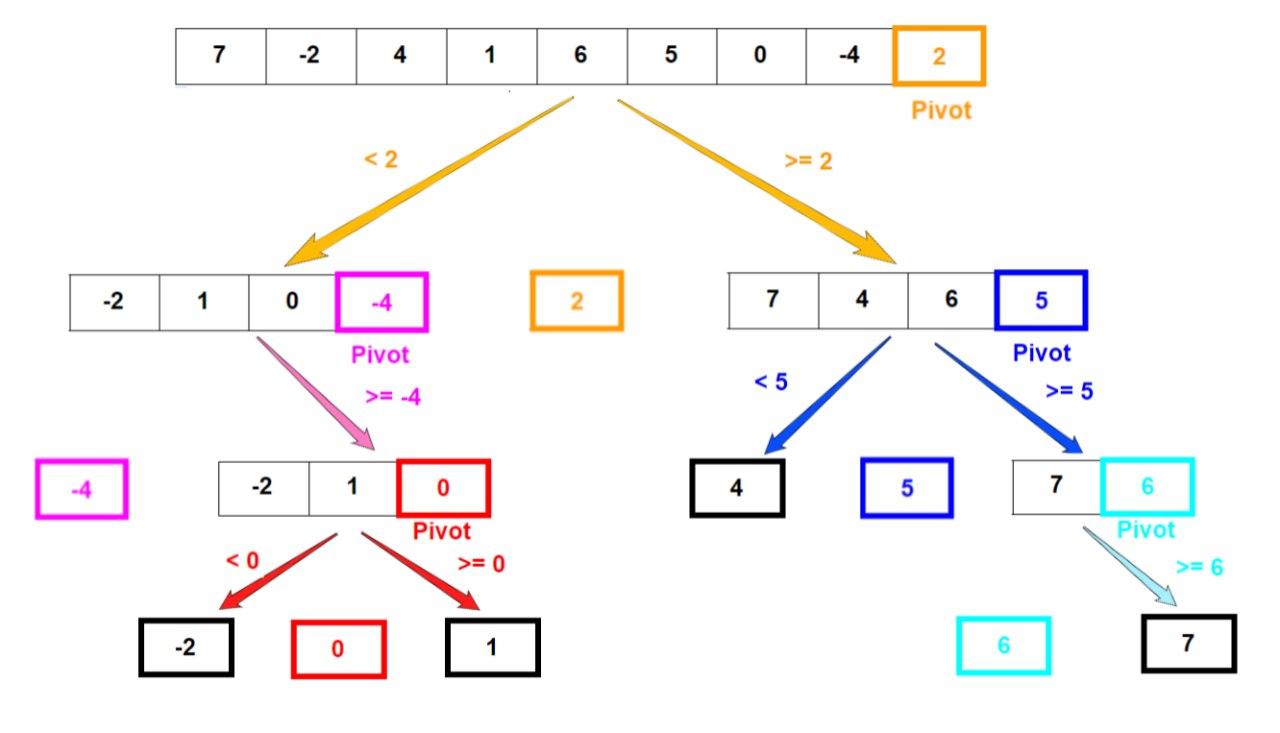
BÁO CÁO CHI TIẾT

Tổng quan:

Sử dụng thư viện Thread và Runable để lập trình đa luồng trong java. Giải quyết bài toán sắp xếp mảng tăng dần bằng đơn luồng và 4 luồng song song. So sánh thời gian giữa 2 cách. Làm rõ khi nào nên sử dụng extend Thread và implement Runable.

1. Chương trình sắp xếp mảng tăng dần:

Trước khi áp dụng lập trình đa luồng vào bài toán ta phải xác định và lựa chọn thuật toán sắp xếp phù hợp, có thể áp dụng song song. Phải đảm bảo thuật toán khi sắp xếp, các phần tử không phụ thuộc vào nhau. Từ điều kiện trên ta có thể chọn ra một số thuật toán phù hợp như MergeSort, QuickSort.



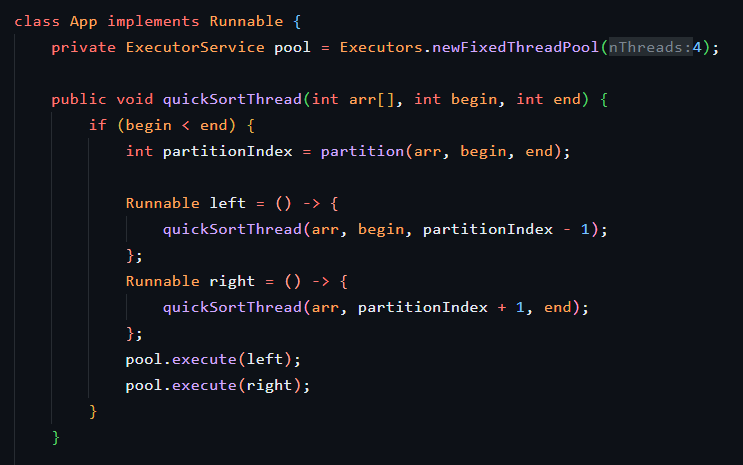
*Hình 1.1: QuickSort*

**Ý tưởng của thuật toán QuickSort:**

Trước tiên chọn một phần tử làm chốt ( pivot ), sau đó sắp xếp các phần tử nhỏ hơn chốt nằm bên trái và lớn hơn bên phải. Tiếp tục áp dụng đệ quy QuickSort cho 2 mảng được chia.

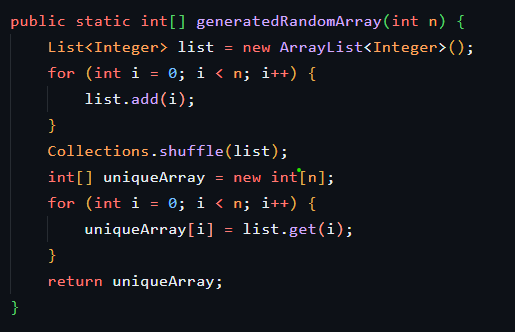
**Áp dụng song song vào bài toán:**

Sau khi chia làm 2 mảng và đệ quy QuickSort cho cả 2 thì chỉ có thể là mảng bên trái hoặc bên phải sử dụng luồng để chạy. Mảng còn lại phải chờ tới khi mảng kia chạy xong mới được gọi. Từ hình 1.1 ta có thể thấy được 2 mảng chạy không phụ thuộc kết quả từ mảng còn lại nên ta có thể áp dụng song song ở chỗ này. Với mỗi lần gọi đệ quy QuickSort cho mảng được chia ta sẽ tạo một thread mới để chạy cùng lúc.



*Hình 1.2*

Với yêu cầu từ bài toán ta khởi tạo pool với số thread có thể sử dụng là 4. Sau đó, tạo 2 biến Runable left, right để đệ quy QuickSort cho mảng bên trái và phải. Dùng pool.execute() để chạy 2 biến trên.



*Hình 1.3: Hàm khởi tạo random arrays*

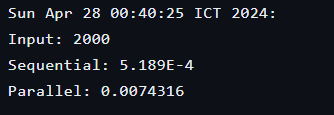
Hàm generatedRandomArray() sẽ khởi tạo array từ 1 -> n sau đó đảo vị trí của các phần tử sử dụng phương thức shuffle() của thư viện Collections.



*Hình 1.4: Hàm main*

Khởi tạo uniqueArray bằng hàm generatedRandomArray() sau đó gán uniqueArray2 để có thể kiểm tra tốc độ thực thi của 2 cách đơn luồng và đa luồng một cách chính xác nhất.

**Kết quả với input = 2000 ( Hình 1.5 )**

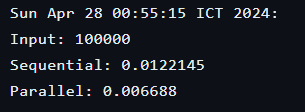


*Hình 1.5*

Ta có thể thấy tốc độ của đơn luồng nhanh hơn tốc độ của đa luồng tới ~14 lần. Tại sao đơn luồng lại có thể nhanh hơn đa luồng như vậy? Đó là tại vì để có thể chạy đa luồng máy tính phải xử lý các công việc khởi tạo luồng, cấp phát bộ nhớ và những công việc này chỉ chạy trên một luồng duy nhất. Một chương trình không thể nào chạy song song 100%.

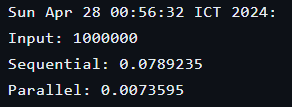
Do đó việc chạy trên càng nhiều luồng thì thời gian khởi chạy sẽ càng tăng và vì input đầu vào chỉ có 2000 là một công việc quá dễ dàng với QuickSort nên trong lúc khởi tạo luồng thì công việc của đơn luồng đã hoàn thành. Để có thể so sánh một cách trực quan hơn về tốc độ của cả 2 cách, ta buộc phải tăng số lượng input đầu vào.

**Kết quả với input = 100000 ( Hình 1.6 )**



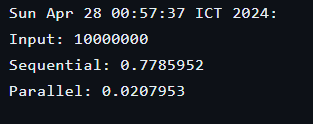
*Hình 1.6*

**Kết quả với input = 1000000 ( Hình 1.7 )**



*Hình 1.7*

**Kết quả với input = 10000000 ( Hình 1.8 )**



*Hình 1.8*

2. Khác nhau giữa extends Thread và implements Runable

Thread và Runable đều cần thiết cho Multi-Threading. Runable giúp xác định cái gì sẽ chạy trong luồng và Thread dùng để khởi chạy một luồng mới. Extends Thread cũng đồng thời implements Runable vì Class Thread mặc định đã implements Runable.

Khi extends Thread thì mọi câu lệnh ta chạy đều nằm ở trong một thread. Nhưng nếu sử dụng implements Runable thì có nhiều sự lựa chọn hơn như bỏ câu lệnh ta muốn chạy vào một ExecutorService hoặc có thể đưa nó vào một luồng xác định và chạy bất cứ lúc nào muốn.

Việc chỉ implements Runable giúp tiết kiệm khoảng trống để ta có thể kế thừa một class khác vì Java chỉ cho phép kế thừa không nhiều hơn 1 lớp.

3. Kết luận:

**Những gì học được sau bài thực hành này:**

* Biết được cách lập trình multithreading trong Java.
* Hiểu rõ hơn về lớp Thread.
* Cách hoạt động của Runable, ExecutorService.

**HẾT**