**BÁO CÁO DỰ ÁN: MÔ PHỆNG CÁC THUẬT TOÁN LẬP LỊCH CPU**

**I. Giới thiệu**

Lập lịch CPU là một trong những chức năng quan trọng nhất của hệ điều hành, đòi hỏi phải tối ưu hóa quy trình xử lý tiến trình nhằm đạt hiệu suất cao nhất.

Dự án này nhầm mô phệ các thuật toán lập lịch CPU phổ biến như:

- First-Come-First-Serve (FCFS): Lần lượt xử lý tiến trình theo thứ tự đến.

- Shortest Job Next (SJN): Động thời ưu tiên tiến trình có thời gian thực thi ngắn nhất.

- Round Robin (RR): Xây dựng tính công bằng nhờ việc cung cấp thời gian xử lý cố định cho mỗi tiến trình theo vòng.

Mục tiêu của dự án bao gồm việc hiểu quá trình thực thi các thuật toán và phân tích tác động của chúng đến các chỉ số hiệu năng như thời gian chờ trung bình, thời gian hoàn thành, và tỷ lệ sử dụng CPU.

**II. Mục tiêu của dự án**

1**. Mô phệ các thuật toán lập lịch CPU nhằm đánh giá hiệu quả lập lịch.**

2. **Trực quan hóa quá trình thực thi các tiến trình bằng giao diện người dùng thân thiện.**

3. **Tăng cường hiểu biết về lập trình hướng đối tượng (OOP).**

4. **Đảm bảo tuân thủ nguyên tắc OOP trong việc thiết kế và thực thi.**

**III. Kiến thức nền tảng**

1. **Thuật toán lập lịch CPU**

- First-Come-First-Serve (FCFS):

- Tiến trình được thực thi theo thứ tự đến.

- Đơn giản nhưng dẫn đến tình trạng "hiệu ứng đoàn tàu", khi tiến trình dài chờ tiến trình ngắn.

- Shortest Job Next (SJN):

- Chọn tiến trình có thời gian thực thi ngắn nhất.

- Hiệu quả trong việc giảm thời gian chờ trung bình.

- Nhược điểm: Cần biết trước thời gian thực thi, điều không thực tế trong nhiều trường hợp.

- Round Robin (RR):

- Cung cấp thời gian quantum cho mỗi tiến trình, thực thi theo vòng.

- Đảm bảo tính công bằng nhưng gia tăng chi phí chuyển đổi ngữ cảnh.

2**. Các chỉ số hiệu năng**

- Thời gian chờ trung bình (Average Waiting Time): Thời gian mà mỗi tiến trình chờ trong hàng đợi trước khi thực thi.

- Thời gian hoàn thành (Turnaround Time):Thời gian tổng từ khi tiến trình vào hàng đợi đến khi hoàn thành.

- Tỷ lệ sử dụng CPU:Tỷ lệ thời gian CPU bận thực thi so với tổng thời gian.

3. **Biểu đồ Gantt**

Biểu đồ Gantt cung cấp hình ảnh trực quan về thứ tự thực thi của các tiến trình trong một quá trình lập lịch.

**IV. Thiết kế hệ thống**

1**. Ngôn ngữ và công cụ**

- Ngôn ngữ lập trình:Java.

- Công cụ giao diện:JavaFX hoặc Swing.

- Quản lý mã nguồn: GitHub.

2. **Các chức năng chính**

- Menu chính:

- Lựa chọn thuật toán lập lịch.

- Hướng dẫn sử dụng (Help).

- Thoát chương trình (Quit).

- Mô phỏng thuật toán:

- Hiển thị danh sách tiến trình, bao gồm thời gian đến, thời gian thực thi, và độ ưu tiên.

- Biểu đồ Gantt cập nhật theo thời gian thực.

- Tính toán các chỉ số hiệu năng.

- Menu hướng dẫn: Mô tả chi tiết nguyên lý hoạt động của mỗi thuật toán.

3**. Biểu đồ Use Case**

- Actor: Người dùng.

- Use Cases:

- Lựa chọn thuật toán.

- Xem hướng dẫn.

- Thực thi mô phỏng.

- Thoát chương trình.

4. **Biểu đồ lớp**

- Các lớp chính:

- Scheduler: Lớp cha chứa logic cơ bản của thuật toán.

- FCFS, SJN, RR: Lớp con thực thi các thuật toán.

- Process: Chứa thông tin về các tiến trình.

- GanttChart: Hiển thị trực quan quá trình thực thi.

**V. Triển khai và kiểm thử**

1. **Quy trình triển khai**

1- Phân tích yêu cầu: Hiểu nguyên lý hoạt động của các thuật toán.

2- Thiết kế:Tạo biểu đồ Use Case và Class Diagram.

3- Viết mã nguồn: Lập trình theo nguyên tắc OOP.

4- Kiểm thử: Thực thi và so sánh kết quả.

2. **Công cụ sử dụng**

- IDE: IntelliJ IDEA/Eclipse.

- Công cụ quản lý mã nguồn:Git.

- Thời gian thực thi:Tối ưu cho Windows/Linux.

3**. Kết quả**

- Giao diện: Đơn giản, trực quan.

- Hiệu năng: Các thuật toán hoạt động đúng và cho kết quả tính toán chính xác.

- So sánh:

- FCFS: Dễ triển khai nhưng thời gian chờ lâu.

- SJN: Hiệu quả cao nhất về thời gian chờ trung bình.

- RR: Tính công bằng được bảo đảm, nhưng có chi phí chuyển đổi ngữ cảnh.

**VI. Kết luận**

Dự án mô phỏng các thuật toán lập lịch CPU đã giúp chúng tôi nắm bắt rõ hơn về cách hoạt động và hiệu quả của từng thuật toán trong thực tế. Qua việc triển khai và đánh giá, chúng tôi đã có những phát hiện quan trọng:

1. **Hiệu suất thuật toán:**
   * **FCFS:** Phù hợp cho các hệ thống đơn giản nhưng không hiệu quả khi xử lý các tiến trình có thời gian thực thi dài.
   * **SJN:** Tối ưu hóa thời gian chờ trung bình, tuy nhiên yêu cầu thông tin về thời gian thực thi trước, điều này không khả thi trong một số trường hợp thực tế.
   * **RR:** Đảm bảo tính công bằng giữa các tiến trình, phù hợp với các hệ thống đa nhiệm, nhưng lại gây tăng chi phí do chuyển đổi ngữ cảnh.
2. **Tầm quan trọng của trực quan hóa:** Việc sử dụng biểu đồ Gantt giúp người dùng dễ dàng hiểu cách hoạt động của thuật toán, từ đó hỗ trợ trong việc phân tích và đánh giá hiệu quả lập lịch.
3. **Ứng dụng OOP:**
   * Dự án đã áp dụng thành công các nguyên tắc của lập trình hướng đối tượng như kế thừa, đa hình và đóng gói, giúp mã nguồn dễ bảo trì và mở rộng.
   * Việc chia nhỏ các lớp và chức năng theo mô-đun đã giảm thiểu rủi ro lỗi và tăng hiệu quả triển khai.
4. **Hạn chế và hướng cải thiện:**
   * Giao diện còn đơn giản, cần cải tiến để nâng cao trải nghiệm người dùng.
   * Hiệu năng của chương trình có thể giảm khi số lượng tiến trình lớn, do đó cần tối ưu hóa thuật toán và quản lý bộ nhớ.

**Tổng kết:** Dự án không chỉ giúp nhóm nắm vững kiến thức về các thuật toán lập lịch CPU mà còn cải thiện kỹ năng làm việc nhóm, tư duy giải quyết vấn đề và lập trình hướng đối tượng. Đây là bước đệm quan trọng để chúng tôi phát triển các ứng dụng phức tạp hơn trong tương lai.