ASSIGNMENT #3

CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

Đọc kỹ hướng dẫn trước khi bắt đầu:

Sinh viên CHỈ được phép sử dụng:

- Tài liệu trên máy tính (bao gồm JDK, NetBeans).
- Đối với việc học từ xa: Google Meet, Hangout (cho mục đích giám sát kỳ thi).

Quy trình làm bài:

1. Tạo thư mục để lưu dự án, ví dụ: CSD_given (1). Tải các tài liệu đã cung cấp vào (1).
2. Hoàn thành các yêu cầu của bài kiểm tra.
3. Trước khi nộp bài:
    - Clean and Build Project (Shift+F11)
    - Sau đó đổi tên thư mục dist thành run. (Nếu thư mục run đã tồn tại, xóa nó trước khi đổi tên).
4. Nộp bài:
    - Để nộp dự án Q1, đầu tiên phải chọn Câu hỏi số: 1
    - Duyệt và chọn thư mục dự án (ví dụ: Q1, hoặc Q1A, hoặc Q2, ...)
    - Sau đó nhấn nút Submit.
5. Không sử dụng tiếng Việt có dấu khi viết các comment trong chương trình.
6. Không thêm câu lệnh import mới cho các file đã cung cấp.
7. Công cụ phần mềm phải sử dụng: NetBeans IDE 8.x và Java JDK 1.8.

Nếu không tuân thủ ít nhất một trong các yêu cầu trên, bài thi sẽ bị điểm 0.

Lưu ý:

- Đầu vào và đầu ra dự kiến bên dưới chỉ được sử dụng để kiểm tra mã của bạn.
- Đầu vào và đầu ra dự kiến trong các trường hợp kiểm tra thực tế (để chấm điểm) khác với các ví dụ bên dưới.
- Không được hardcode với các kết quả dự kiến đã cho.

Xử lý sự cố:

Nếu dự án đã cho (ví dụ: Q1) chạy với lỗi, bạn cần chạy "Clean and Build Project" (Shift+F11). Nếu vẫn còn lỗi, hãy thử đổi tên hoặc sao chép dự án sang dự án khác, ví dụ: từ Q1 sang Q1X hoặc Q1Y.

Mô tả bài toán

Các file đã cung cấp chứa các câu lệnh để triển khai một chương trình quản lý lịch trình xử lý nhiệm vụ. Cấu trúc của các lớp chính như sau:

- **Lớp Task**: chứa thông tin về một nhiệm vụ, bao gồm id, tên nhiệm vụ, độ ưu tiên và thời gian ước tính để hoàn thành (tính bằng phút).
- **Lớp Node**: bao gồm một đối tượng Task và một con trỏ next để liên kết.
- **Lớp WaitingHeap**: là một heap tối thiểu được triển khai dưới dạng mảng, trong đó các Task được sắp xếp theo độ ưu tiên (số càng nhỏ càng ưu tiên).
- **Lớp ExecutionStack**: là một cấu trúc stack (được triển khai dưới dạng danh sách liên kết), nơi các Task đang được xử lý.
- **Lớp TaskScheduler**: là lớp chính của chương trình, chứa một WaitingHeap để lưu trữ các nhiệm vụ đang chờ và một ExecutionStack cho các nhiệm vụ đang xử lý.

Sinh viên cần đọc kỹ các phần mã đã cung cấp để hiểu đầy đủ mối quan hệ giữa các lớp và các chức năng trong mỗi lớp. Các yêu cầu cụ thể của bài kiểm tra là thực hiện các chức năng sau:

a. f1() (2.5 điểm):

Để hoàn thành yêu cầu f1, sinh viên cần thực hiện hai nhiệm vụ cụ thể: triển khai hàm insert() trong cấu trúc WaitingHeap và hàm push() trong cấu trúc ExecutionStack. Đầu ra dự kiến để kiểm tra mã của bạn như sau:

Waiting Heap: (T003,Report Generation,4,45) (T001,Database Backup,2,60) (T005,User Authentication,5,30) (T007,Email Notification,8,15) (T002,Log Analysis,3,75) (T009,System Monitoring,10,10)

Execution Stack: (T004,Data Synchronization,1,90) (T006,Error Handling,6,20) (T008,Cache Cleaning,7,25)

b. f2() (2.5 điểm):

Để hoàn thành yêu cầu f2, sinh viên cần thực hiện hai nhiệm vụ cụ thể: triển khai hàm pop() của cấu trúc ExecutionStack và hàm removeMin() trong lớp WaitingHeap, sau đó sử dụng kết quả trả về từ các hàm này để thực hiện hành động cho hàm f2. Hàm removeMin() phải:

- Lấy ra nhiệm vụ có độ ưu tiên cao nhất (số nhỏ nhất) từ WaitingHeap.
- Điều chỉnh lại heap để duy trì tính chất heap tối thiểu. Đầu ra dự kiến để kiểm tra mã của bạn như sau:

Waiting Heap: (T003,Report Generation,4,45) (T001,Database Backup,2,60) (T005,User Authentication,5,30) (T007,Email Notification,8,15) (T002,Log Analysis,3,75) (T009,System Monitoring,10,10)

Execution Stack: (T004,Data Synchronization,1,90) (T006,Error Handling,6,20) (T008,Cache Cleaning,7,25)

Task completed from Execution Stack: (T008,Cache Cleaning,7,25)

Next task from Waiting Heap to Execution: (T001,Database Backup,2,60)

Waiting Heap: (T002,Log Analysis,3,75) (T003,Report Generation,4,45) (T005,User Authentication,5,30) (T007,Email Notification,8,15) (T009,System Monitoring,10,10)

Execution Stack: (T001,Database Backup,2,60) (T004,Data Synchronization,1,90) (T006,Error Handling,6,20)

Giải thích: Ban đầu, ExecutionStack chứa 3 phần tử và WaitingHeap chứa 6 phần tử. Sau khi thực hiện pop() để loại bỏ phần tử trên cùng T008 khỏi stack và removeMin() để lấy phần tử có độ ưu tiên cao nhất T001 từ heap, WaitingHeap còn 5 phần tử và ExecutionStack vẫn có 3 phần tử (vì chúng ta đã loại bỏ một phần tử và thêm một phần tử mới).

c. f3() (2.5 điểm):

Triển khai một thuật toán để tính tổng thời gian ước tính của tất cả các nhiệm vụ trong WaitingHeap và ExecutionStack. Đầu ra dự kiến để kiểm tra mã của bạn như sau:

Waiting Heap: (T003,Report Generation,4,45) (T001,Database Backup,2,60) (T005,User Authentication,5,30) (T007,Email Notification,8,15) (T002,Log Analysis,3,75) (T009,System Monitoring,10,10)

Execution Stack: (T004,Data Synchronization,1,90) (T006,Error Handling,6,20) (T008,Cache Cleaning,7,25)

Total Estimated Time: 370 minutes

Giải thích: Tổng thời gian ước tính được tính bằng tổng thời gian của tất cả các nhiệm vụ trong cả WaitingHeap và ExecutionStack. Tổng = 45 + 60 + 30 + 15 + 75 + 10 + 90 + 20 + 25 = 370 phút.

d. f4() (2.5 điểm):

Triển khai một thuật toán để chuyển các nhiệm vụ có độ ưu tiên cao hơn một ngưỡng cho trước (ví dụ: độ ưu tiên <= 5) từ WaitingHeap sang ExecutionStack. Đầu ra dự kiến để kiểm tra mã của bạn như sau:

Waiting Heap: (T003,Report Generation,4,45) (T001,Database Backup,2,60) (T005,User Authentication,5,30) (T007,Email Notification,8,15) (T002,Log Analysis,3,75) (T009,System Monitoring,10,10)

Execution Stack: (T004,Data Synchronization,1,90) (T006,Error Handling,6,20) (T008,Cache Cleaning,7,25)

Threshold Priority: 5

Tasks moved: 4

Waiting Heap: (T007,Email Notification,8,15) (T009,System Monitoring,10,10)

Execution Stack: (T005,User Authentication,5,30) (T002,Log Analysis,3,75) (T001,Database Backup,2,60) (T003,Report Generation,4,45) (T004,Data Synchronization,1,90) (T006,Error Handling,6,20) (T008,Cache Cleaning,7,25)

Giải thích: Ngưỡng độ ưu tiên được đặt là 5, vì vậy tất cả các nhiệm vụ có độ ưu tiên <= 5 sẽ được chuyển từ WaitingHeap sang ExecutionStack. Có 4 nhiệm vụ thỏa mãn điều kiện này (T001, T002, T003, T005), vì vậy chúng được chuyển sang ExecutionStack và WaitingHeap chỉ còn lại 2 nhiệm vụ.