Hướng dẫn cách làm Assignment A1

P1 Create a design specification for data structures explaining the valid operations that can be carried out on the structures.  
Đặc tả cấu trúc dữ liệu QUEUE.

Giới thiệu Stack, queue

Các phép toán thông dụng trên stack: Pop, push, empty,…queue, dequeue,…

P2 Determine the operations of a memory stack and how it is used to implement function calls in a computer.

Xác định các thao tác của 1 stack bộ nhớ. Trình bày cách thức hoạt động của các thao tác này trong việc ứng dụng gọi thực hiện các chức năng trong máy tính.

P3 Using an imperative definition, specify the abstract data type for a software stack.

Sử dụng định nghĩa để dặc tả cấu trúc dữ liệu stack

M1 Illustrate, with an example, a concrete data structure for a First In First out (FIFO) queue.

Sử dụng ví dụ cụ thể trong cuộc sống để mô tả cấu trúc dữ liệu queue.

M2 Compare the performance of two sorting algorithms.

Chọn 2 thuật toán sắp xếp bất kỳ và so sánh hiệu quả của chúng:

* Độ phức tạp
* Thời gian,
* Tài nguyên,…

M3 Examine the advantages of encapsulation and information hiding when using an ADT.

Trình bày các ưu điểm của đóng gói và che dấu thông tin khi sử dụng ADT

D1 Analyse the operation, using illustrations, of two network shortest path algorithms, providing an example of each.

Phân tích, trình bày các thao tác của 2 thuật toán tìm đường đi ngắn nhất (Dijsktra và Bellman ford)

D2 Discuss the view that imperative ADTs are a basis for object orientation and, with justification, state whether you agree.

Trình bày quan điểm của bạn đối với phát biểu “ADT là nền tảng cyar hướng đối tượng”

Hướng dẫn cách làm Assignment 2

Sử dụng lại A1 để phát triển tiếp thành A2.

Yêu cầu của A2: Xây dựng 1 chương trình cho phép truyền message là 1 chuỗi 250 ký tự từ nguồn sang đích. Phía nguồn sử dựng queue để làm buffer và phía đích sử dụng stack để xử lý chuỗi.

* Sử dụng các ADT (STACK/Queue) để làm bộ đệm trong quá trình truyền và nhận chuỗi dữ liệu
* Xử lý lỗi: sử dụng trình bắt ngoài lệ Exception – khối “try … catch” để bắt và xử lý lỗi nếu có phát sinh trong quá trình thực thi chương trình
* Test chương trình và báo cáo kết quả kiểm thử (Test case, test log)

**P4** Implement a complex ADT and algorithm in an executable programming language to solve a well defined problem.  
Sử dụng Java hoặc C#, C++,… để viết chương trình theo yêu cầu trên

P5 Implement error handling and report test results.

Sử dụng trình bắt lỗi Exception:

try{

….

}

catch{

…

}

finally{}

* Test chương trình và báo cáo kết quả kiểm thử (Test case, test log)

P6 Discuss how asymptotic analysis can be used to assess the effectiveness of an algorithm

Thảo luận việc sử dụng phân tích tiệm cận để đánh giá hiệu quả của một toán toán (đánh giá bằng lý thuyết)

P7 Determine two ways in which the efficiency of an algorithm can be measured, illustrating your answer with an example.

Trình bày 2 yếu tố được dùng để đo lường tính hiệu quả của thuật toán. (Time/ space)

Sử dụng ví dụ cụ thể để minh họa cho trả lời

* M4 Demonstrate how the implementation of an ADT/algorithm solves a well-defined problem

Trình bày/ giải thích việc sử dụng ADT/thuật toán có thể giải quyết bài toán ở P4

* M5 Interpret what a trade-off is when specifying an ADT using  
  an example to support your answer

Giải thích điểm cân bằng khi đặc tả thuật toán. (Một thuật toán nếu thực hiện nhanh sẽ tốn nhiều bộ nhớ. Ngược lại muốn tiết kiệm bộ nhớ thì thuật toán sẽ chạy chậm. Trade-off là điểm cân bằng giữa thời gian và bộ nhớ mà hiệu suất có thể cháp nhận được.

Sử dụng ví dụ để hỗ trợ

* D3 Critically evaluate the complexity of an implemented ADT/algorithm

Đánh giá độ phức tạp của 1 thuật toán

* D4 Evaluate three benefits of using implementation independent data structures

Đánh giá 3 ưu điểm của việc sử cài đặt độc lập với cấu trúc dữ liệu