ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH

Bài tập lớn 1

JJK RESTAURANT OPERATIONS
(Phần 1)

TP. HỒ CHÍ MINH, THÁNG 09/2023

Đặc Tả Bài Tập Lớn 1

Phiên bản 1.0

1 Chuẩn đầu ra

Sau khi hoàn thành bài tập lớn này, sinh viên sẽ có khả năng:

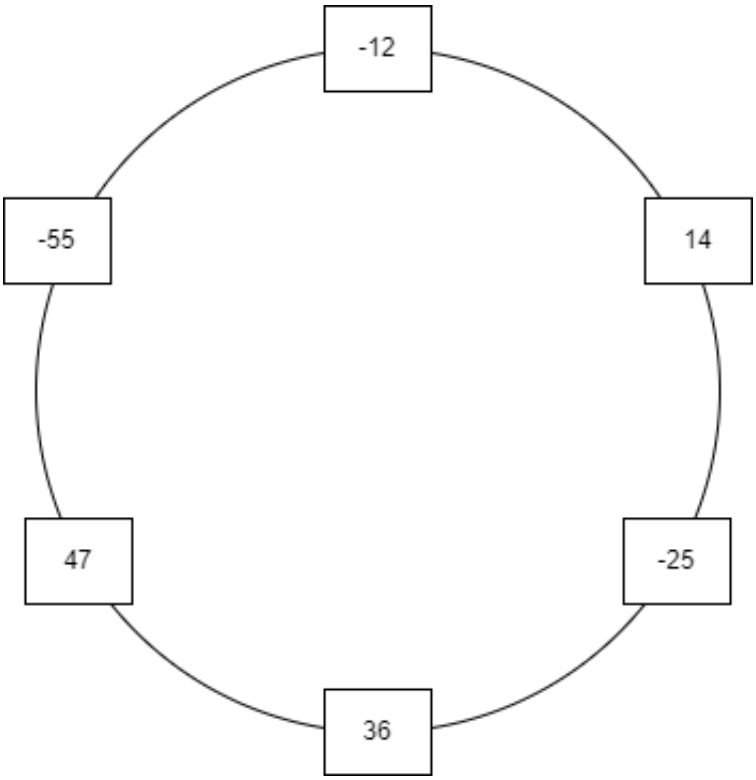
- Hiện thực cấu trúc dữ liệu danh sách liên kết.
- Chọn lựa và vận dụng các cấu trúc dữ liệu phù hợp để đạt được các kết quả mong muốn.
- Biết thêm về một bộ truyện tranh.

2 Giới thiệu

Trong bài tập lớn này, sinh viên sẽ mô phỏng việc xử lý yêu cầu đặt bàn và sắp xếp vị trí chỗ ngồi cho các chú thuật sư và các oán linh (gọi chung là khách hàng) trong một nhà hàng được đồng sở hữu bởi Gojo và Sukuna thông qua các lệnh được mô tả ở phần 2.1.

Hình 1: Hình ảnh tại nhà hàng.

Lưu ý: Nhà hàng sẽ không dùng số thứ tự ứng với từng bàn để xác định chỗ ngồi cho khách mà sẽ bố trí linh động dựa theo thông tin của khách. Và sau khi xử lý xong tất cả các lệnh trong tập tin đầu vào và xuất kết quả ra màn hình, chương trình phải đảm bảo huỷ tất cả các đối tượng dữ liệu được cấp phát động, không để lại rác trong bộ nhớ trước khi kết thúc chương trình.



Hình 2: Danh sách liên kết đôi vòng minh họa vị trí chỗ ngồi trong nhà hàng dựa vào ENERGY.

Ý nghĩa của từ viết tắt và kiểu dữ liệu của các thông số được mô tả như sau:

- **NAME**: một chuỗi ký tự trong bảng chữ cái Alphabet (bao gồm cả chữ viết thường và viết hoa) liên tục không có khoảng trắng, biểu thị tên của khách hàng.
- **ENERGY**: một số nguyên biểu thị năng lượng của các chú thuật sư (giá trị dương), và oán linh (giá trị âm).
- **NUM**: một số nguyên với ý nghĩa khác nhau ứng với từng lệnh xử lý khác nhau. Và ứng với mỗi lệnh thì giá trị NUM này sẽ có các khoảng giá trị khác nhau.
- **MAXSIZE**: số lượng bàn tối đa mà nhà hàng có thể phục vụ. Tuy nhiên giá trị này có thể tạm thời thay đổi trong quá trình vận hành nhà hàng.

2.1 Danh sách lệnh

RED <NAME> <ENERGY>:

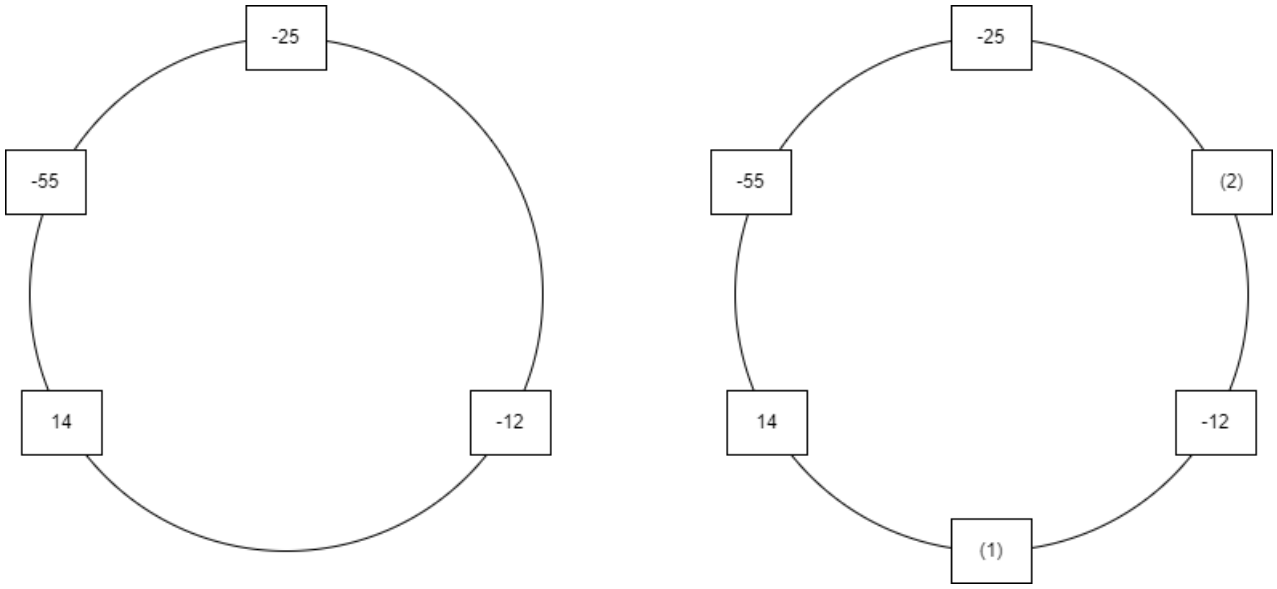
Hình 3: Thuật thức RED. [4]

Lệnh này được dùng để nhân viên ghi nhận thông tin khách hàng từ đó sắp xếp vị trí chỗ ngồi cho khách.

Khách hàng đến nhà hàng sẽ phải cung cấp thông tin cho nhân viên để nhân viên sắp xếp chỗ ngồi. Và trong một vài trường hợp, nhân viên sẽ tùy theo giá trị của ENERGY của khách hàng để bố trí chỗ ngồi. Tuy nhiên, nhà hàng cũng có một số quy định về việc sắp xếp vị trí chỗ ngồi như sau:

- Nhà hàng chỉ tiếp đón chú thuật sư (ENERGY dương) hoặc oán linh (ENERGY âm) và từ chối nhận khách hàng có ENERGY bằng 0.
- Khi nhà hàng vừa mở cửa thì khách hàng đầu tiên đến có thể ngồi tại bất cứ vị trí nào trong bàn tròn.

- Sau đó, việc bố trí chỗ ngồi cho khách mới sẽ **được tính từ vị trí gần nhất vừa có sự thay đổi** (ngồi vào bàn hoặc ra khỏi bàn). Giả sử gọi là vị trí X.
- Khi vị khách tiếp theo đến, nhân viên sẽ bố trí chỗ ngồi cho khách tại vị trí liền kề bên phía thuận chiều kim đồng hồ nếu ENERGY của khách lớn hơn hoặc bằng với ENERGY của khách tại vị trí X. Ngược lại, khách sẽ ngồi tại vị trí liền kề bên phía ngược chiều kim đồng hồ của khách tại vị trí X.
- Ví dụ: giả sử vị trí X là vị trí có giá trị ENERGY = -12 (hình bên trái) và đang được xem xét để thêm phần tử vào. Vị trí (1) được xem là vị trí liền kề phần tử X theo chiều kim đồng hồ, vị trí (2) được xem là vị trí liền kề phần tử X bên phía ngược chiều kim đồng hồ (hình bên phải).



Hình 4: Minh họa việc thêm phần tử liền kề tại vị trí X.

- Tuy nhiên, khi số lượng khách trong bàn lớn hơn hoặc bằng MAXSIZE/2. Nhân viên sẽ đổi chiến lược chọn chỗ ngồi cho khách. Vì biết những người có ENERGY gần bằng nhau thường không thích ngồi gần nhau. Do đó trước khi chọn vị trí, nhân viên sẽ tính giá trị chênh lệch lớn nhất giữa vị khách mới với tất cả vị khách trong nhà hàng thông qua việc lấy hiệu trị tuyệt đối của từng cặp ENERGY ứng với từng khách hàng (giả sử gọi là **RES**) để quyết định chỗ ngồi. Nếu:
  - Có nhiều vị trí sau khi tính đều trả về cùng giá trị RES thì sẽ chọn vị trí đầu tiên tìm được theo chiều kim đồng hồ. Chỉ có một giá trị RES lớn nhất tìm được thì chọn vị trí đó.
  - Sau đó, nhân viên sẽ bỏ trị tuyệt đối của RES. Trường hợp kết quả là số âm thì thêm vào liền kề bên phía ngược chiều kim đồng hồ tại vị trí tìm được. Ngược lại,

thêm vào liền kề bên phía thuận chiều kim đồng hồ.

- Và bởi vì "thiên thượng thiên hạ, duy ngã độc tôn", nên nhà hàng sẽ không đón tiếp các vị khách đến sau và không cho phép vào hàng chờ nếu có tên giống với tên của các vị khách đang dùng bữa trong nhà hàng (hoặc đã có tên trong hàng chờ). Ví dụ một chú thuật sư tên **ABC** đang dùng bữa thì những chú thuật sư hoặc oán linh đến sau có tên **ABC** sẽ bị mời về.
- Nếu số lượng khách ngồi vào bàn đã đạt đến MAXSIZE thì sẽ dừng việc nhận khách và đưa khách vào hàng chờ. Số lượng khách tối đa trong hàng chờ là một số nguyên và có giá trị bằng MAXSIZE. Nếu hàng chờ đã đạt đến số lượng tối đa thì nhà hàng sẽ ngưng nhận khách.

BLUE <NUM>:

Hình 5: Thuật thức BLUE. [4]

Lệnh này được dùng để nhân viên đuổi khách và dọn bàn vì đã dùng món quá lâu. Sau khi dọn xong, nếu trong hàng chờ có khách, nhân viên sẽ tiến hành chọn chỗ cho khách ngược lại không làm gì cả.

Khi nhận được lệnh, nhân viên sẽ đuổi NUM vị khách theo thứ tự vào nhà hàng từ sớm nhất đến gần đây nhất. Ví dụ: khách đến nhà hàng theo thứ tự $A \longrightarrow B \longrightarrow C \longrightarrow D$ thì sau khi thực thi lệnh **BLUE 2** thì trong nhà hàng chỉ còn hai vị khách là C và D. Trường hợp nếu NUM lớn hơn hoặc bằng số lượng khách trong bàn ăn hoặc lớn hơn MAXSIZE thì xem như chủ nhà hàng quyết định đuổi hết khách trong bàn ăn.

Lưu ý: việc chọn chỗ ngồi mới cho khách trong hàng chờ chỉ được thực hiện sau khi đã thực

hiện xong lệnh BLUE. Cơ chế chọn chỗ ngồi cho khách tương tự như lệnh RED và thứ tự của khách được xem xét ngồi vào bàn từ hàng chờ sẽ theo cơ chế First In First Out (FIFO). Ngoài ra, giá trị của NUM sẽ được đảm bảo luôn lớn hơn 0.

PURPLE:

Hình 6: Thuật thức Purple.[2]

Lệnh này dùng để sắp sếp lại hàng chờ dựa theo thứ tự giảm dần trị tuyệt đối của ENERGY (tính từ đầu hàng chờ) theo giải thuật Shell Sort và có thể dẫn đến việc thay đổi các vị trí của khác trong nhà hàng. Sinh viên đọc và hiện thực giải thuật Shell Sort theo đúng với tinh thần được mô tả trong tài liệu hướng dẫn [5] ở chương 7.3.

Khi nhận được lệnh này, nhân viên sẽ tiến hành sắp xếp như sau:

- Đầu tiên, tìm vị trí của khách hạng có giá trị trị tuyệt đối của ENERGY cao nhất trong hàng chờ. Nếu có nhiều chú thuật sư có cùng giá trị trị tuyệt đối của ENERGY cao nhất thì chọn vị khách nào được thêm vào hàng chờ gần đây nhất. viết tiếp trong hàm addwait
- Sau đó tiến hành sắp xếp lại hàng chờ **CHỈ** tính từ vị trí tìm được cho đến đầu hàng chờ. **Trường hợp khách hàng có cùng ENERGY với nhau thì khách hàng nào đến trước được xem là lớn hơn.**

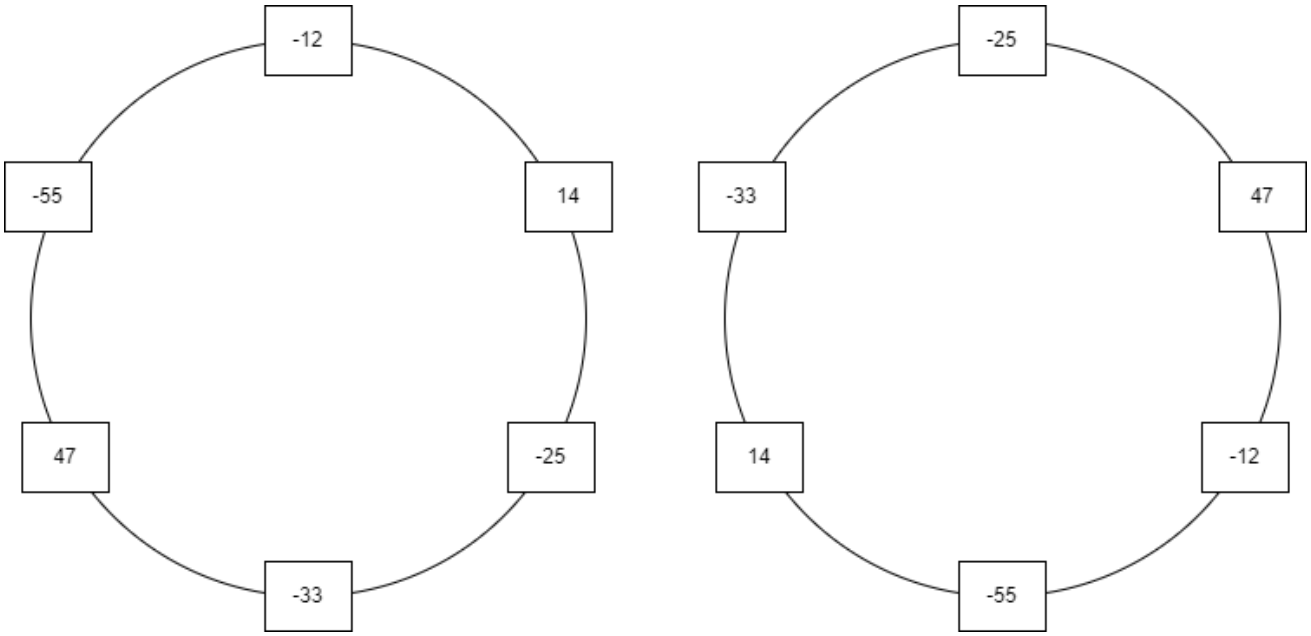
- Gọi số lần chuyển đổi vị trí trong hàng chờ khi thực hiện việc sắp xếp bằng giải thuật Shell Sort là N. Nhân viên sẽ tiếp tục thực hiện lệnh **BLUE <N mod MAXSIZE>**

REVERSAL:

Từ vị trí X và xét theo chiều ngược chiều kim đồng hồ, tiến hành đảo ngược vị trí của khách hàng trong nhà hàng.

Tuy nhiên, nhân viên chỉ có thể đảo ngược vị trí của những khách hàng cùng là oán linh hoặc cùng là chú thuật sư và không thể hoán đổi vị trí của oán linh và chú thuật sư cho nhau.

Ví dụ đối với vị trí của các khách hàng được mô tả ở hình bên trái. Giả sử vị trí X là vị trí có ENERGY = -12, sau khi thực hiện lệnh REVERSAL sẽ được kết quả như hình bên phải:



Hình 7: Kết quả sau khi thực hiện lệnh REVERSAL (từ trái sang phải).

UNLIMITED_VOID:

Bắt đầu từ vị trí X, xét theo chiều thuận chiều kim đồng hồ, tìm dãy con dài nhất (số phần từ liên tiếp nhau >= 4) có tổng giá trị ENERGY nhỏ nhất ứng với các vị khách trong bàn ăn. Sau đó in ra màn hình thông tin các vị khách trong dãy tìm được theo định dạng "NAME-ENERGY/n" theo thứ tự thuận theo chiều kim đồng hồ tính từ phần tử nhỏ nhất trong dãy con đó. Nếu tồn tại nhiều dãy con có cùng giá trị thì chọn dãy con cuối cùng tìm được. Trường hợp không tồn tại dãy con thỏa yêu cầu thì không cần làm gì cả.

Hình 8: Thuật thức vô lượng không xứ. [3]

DOMAIN_EXPANSION:

Vì các chú thuật sư và oán linh trong quá trình dùng bữa có lời qua tiếng lại dẫn đến mất không khí dùng bữa tại nhà hàng. Cho nên chủ nhà hàng quyết định sẽ đuổi tất cả chú thuật sư hoặc oán linh nếu:

- Tổng ENERGY của tất cả chú thuật sư lớn hơn hoặc bằng tổng trị tuyệt đối ENERGY của tất cả chú linh có mặt tại nhà hàng **(đang có mặt ở nhà hàng = tất cả khách hàng trong bàn ăn và trong hàng chờ)** thì nhân viên sẽ đuổi tất cả các oán linh đang có mặt ở nhà hàng. Ngược lại nhân viên sẽ đuổi tất cả các chú thuật sư đang có mặt ở nhà hàng.
- Sau đó nếu có vị trí trống, nhân viên sẽ tiếp tục bố trí khách hàng trong hàng chờ vào bàn ăn.
- Đồng thời, in ra thông tin các chú thuật sư hoặc oán linh bị đuổi về theo thứ tự thời gian thời gian của khách hàng đến nhà hàng gần đây nhất cho đến khách hàng đến nhà hàng sớm nhất theo định dạng: "NAME-ENERGY/n".

LIGHT <NUM>:

In thông tin của tất cả vị khách ngồi vào bàn ăn tính từ vị trí gần nhất vừa có sự thay đổi theo chiều kim đồng hồ nếu giá NUM dương, và in theo chiều ngược chiều kim đồng hồ nếu

giá trị NUM âm theo định dạng "NAME-ENERGY/n". Trường hợp NUM bằng 0 thì in thông tin của các vị khách trong hàng chờ theo thứ tự từ đầu hàng chờ đến cuối hàng chờ theo định dạng "NAME-ENERGY/n".

2.2 Kết bài

Bởi vì trong quá trình điều hành nhà hàng, giữa hai người đồng sở hữu nhà hàng là Gojo và Sukuna vì bảo vệ quyền lợi cho khách hàng của mình đã bất đồng quan điểm. Cho nên nhà hàng tạm thời đóng cửa để hai vị giải quyết. Kết quả của sau khi giải quyết sẽ được mô tả trong bài tập lớn tiếp theo. Hy vọng sẽ không có gì phải bị chia làm đôi.

Hình 9: Chủ nhà hàng lời qua tiếng lại với nhau để bênh vực khách của mình. [1]

2.3 Yêu cầu

Để hoàn thành bài tập lớn này, sinh viên phải:

- Tải xuống tập tin initial.zip và giải nén nó.
- Sau khi giải nén sẽ được 4 files: main.cpp, main.h, restaurant.cpp và test.txt. Sinh viên **KHÔNG ĐƯỢC** sửa đổi các file main.cpp, main.h.
- Sinh viên được quyền chỉnh sửa bất kỳ nội dung nào trong file restaurant.cpp để hiện thực bài toán.
- Đảm bảo rằng chỉ có một lệnh **include** trong file restaurant.cpp đó là **#include "main.h"**. Ngoài ra, không cho phép có một **include** nào khác trong file này.
- Bài tập lớn này dùng để kiểm tra việc sử dụng danh sách liên kết nên sinh viên không được phép dùng bất cứ các dòng lệnh nào liên quan đến static array hoặc dynamic array như: *(arr), arr[i], ... Ngoài ra, sinh viên **chỉ được dùng lệnh print() được đặc tả trong class customer** để in thông tin của khách hàng ra màn hình. Vì khi chấm bài, lệnh print() này có thể được thay đổi để kiểm tra một số vấn đề khác trong bài tập lớn.
- Môi trường dùng để chấm bài là g++ (MinGW-W64 i686-ucrt-posix-dwarf) 11.2.0.

3 Nộp bài

Sinh viên chỉ nộp file: **restaurant.cpp**, trước thời hạn được đưa ra trong đường dẫn "Assignment 1 Submission". Có một số testcase đơn giản được sử dụng để kiểm tra bài làm nhằm đảm bảo rằng kết quả có thể biên dịch và chạy được. Sinh viên có thể nộp bài bao nhiêu lần tùy ý nhưng chỉ có bài nộp cuối cùng được tính điểm. Vì hệ thống không thể chịu tải khi quá nhiều em nộp bài cùng một lúc, vì vậy sinh viên nên nộp bài càng sớm càng tốt. Sinh viên sẽ tự chịu rủi ro nếu nộp bài sát hạn chót. Khi quá thời hạn nộp bài, hệ thống sẽ đóng nên sẽ không thể nộp nữa. Bài nộp qua email không được chấp nhận.

4 Harmony

Trong đề thi có thể có các câu hỏi liên quan đến nội dung bài tập lớn (phần mô tả câu hỏi sẽ được nêu rõ là dùng để harmony cho bài tập lớn, nếu có) . Điểm của các câu hỏi harmony sẽ được scale về thang 10 và sẽ được dùng để tính lại điểm của các bài tập lớn. Cụ thể:

- Gọi x là điểm bài tập lớn.
- Gọi y là điểm của các câu hỏi harmony sau khi scale về thang 10.

Điểm cuối cùng của bài tập lớn sẽ được tính theo công thức trung bình điều hòa sau:

$$\mathbf{Assignment\_Score = 2 * x * y / (x + y)}$$

5 Xử lý gian lận

Bài tập lớn phải được sinh viên TỰ LÀM. Sinh viên sẽ bị coi là gian lận nếu:

- Có sự giống nhau bất thường giữa mã nguồn của các bài nộp. Trong trường hợp này, TẤT CẢ các bài nộp đều bị coi là gian lận. Do vậy sinh viên phải bảo vệ mã nguồn bài tập lớn của mình.
- Sinh viên không hiểu mã nguồn do chính mình viết, trừ những phần mã được cung cấp sẵn trong chương trình khởi tạo. Sinh viên có thể tham khảo từ bất kỳ nguồn tài liệu nào, tuy nhiên phải đảm bảo rằng mình hiểu rõ ý nghĩa của tất cả những dòng lệnh mà mình viết. Trong trường hợp không hiểu rõ mã nguồn của nơi mình tham khảo, sinh viên được đặc biệt cảnh báo là KHÔNG ĐƯỢC sử dụng mã nguồn này; thay vào đó nên sử dụng những gì đã được học để viết chương trình.
- Nộp nhầm bài của sinh viên khác trên tài khoản cá nhân của mình.
- Nếu vi phạm các ràng buộc được mô tả ở mục 2.3, sinh viên sẽ nhận điểm 0.

Trong trường hợp bị kết luận là gian lận, sinh viên sẽ bị xử lý theo kết luận của Hội đồng giảng viên giảng dạy môn học này.

KHÔNG CHẤP NHẬN BẤT KỲ GIẢI THÍCH NÀO VÀ KHÔNG CÓ BẤT KỲ NGOẠI LỆ NÀO!

Tài liệu tham khảo

[1] https://www.deviantart.com/

[2] https://www.pxfuel.com/

[3] https://www.wallpaperflare.com/

[4] https://wallpaperaccess.com/

[5] Data Structures & Algorithm Analysis by Clifford A. Shaffer - Edition 3.2 (C++ Version)

——————————HẾT——————————