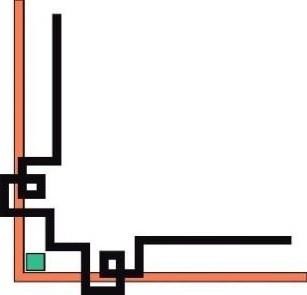
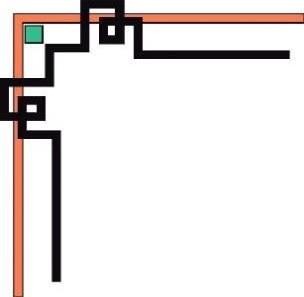
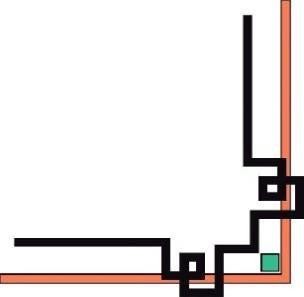
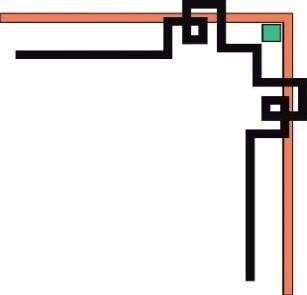
BỘ GIÁO DỤC VÀ ĐÀO TẠO



TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM KHOA: CÔNG NGHỆ THÔNG TIN

BÁO CÁO CUỐI KỲ

MÔN HỌC: CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

ĐỀ TÀI: MÔ PHỎNG DẠNG HÌNH ẢNH CHO CÁC CẤU TRÚC DỮ LIỆU: STACK, QUEUE, DEQUE, LINK LIST, CIRCLE LIST

GVHD: Huỳnh Xuân Phụng

**Nhóm sinh viên thực hiện:**

**Nhóm 20**

|  |  |  |
| --- | --- | --- |
| STT | Họ và tên | MSSV |
| 1 | Lê Đình Đạt | 21110416 |
| 2 | Lê Dưỡng | 21110412 |
| 3 | Trần Văn Đại | 21110875 |

Thành phố Hồ Chí Minh, tháng 12 năm 2022

**DANH SÁCH NHÓM**

**HỌC KÌ I, NĂM HỌC 2022 – 2023**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Họ và tên | Mssv | Tỉ lệ hoàn thành |
| 1 | Lê Đình Đạt | 21110416 | 100% |
| 2 | Trần Văn Đại | 21110875 | 100% |
| 3 | Lê Dưỡng | 21110412 | 100% |

***Nhận xét của giáo viên***

*Ngày 27 tháng 12 năm 2022*

**Giáo viên chấm điểm**

**GV.TS. Huỳnh Xuân Phụng**

**Đồ án cuối kì môn cấu trúc dữ liệu và giải thuật**

# Đề tài

Mô Phỏng Dạng hình ảnh với các thuật toán Insert,Delete cho các cấu trúc dữ liệu:

* Stack
* Queue
* Deque
* Link list
* Circle list

Viết Thuật toán và chương trình minh họa hình ảnh từng bước thực thi của thuật toán Topological sort.

# Chi tiết các nội dung đã làm

## Chương trình mô phỏng stack, queue, deque, link list, circle list

Các chức năng chính:

Mô phỏng hình ảnh các cấu trúc dữ liệu.

Mô phỏng hình ảnh quá trình thêm một Node vào các cấu trúc dữ liệu

Mô phỏng hình ảnh quá trình xóa một Node ra khỏi cấu trúc.

-Xây dựng Struct nodeStack( NodeQueue, nodeDequeue, nodeList, nodeList) gồm các trường :

+ value(string): để biểu diễn giá trị của node

+ x,y(int): để lưu lại vị trí của node trên màn hình console.

+ hide(bool) : để biểu diễn trạng thái ẩn/hiện của node trên màn hình console.

1. Các hàm cơ bản

* Hàm createNode : dùng để vẽ ra các node cùng với giá trị của nó ( sử dụng hàm rectangle và drawText ở file graphics.h)
* Hàm showNode : dùng để ẩn/hiện node trên màn hình console thông qua trường hide của node đó
* Hàm createStack( createQueue, createDequeue…) dùng để vẽ ra bộ khung của cấu trúc dữ liệu tương ứng.
* Hàm setup dùng để vẽ lại tất cả các node với trạng thái hiện tại (sử dụng hàm showNode và create để vẽ lại)
* Hàm menu : sử dụng cấu trúc rẽ nhánh switch case để tạo ra menu lựa chọn các chức năm.

1. Chi tiết các hàm của từng cấu trúc

* Stack
* Các biến toàn cục :

nCur : biểu diễn số node chưa đưa vào stack.

ESP : biểu diễn tọa độ y của node tiếp theo trong stack.

nStack : biểu diễn số phần tử có trong stack

* Các hằng:

Stopxdelete,stopydelete: lưu tọa độ của node. Khi node bị xóa khỏi stack

Hspace,wspace: lưu khoảng cách giữa các node.

* Hàm insertrun : Sử dụng biến nCur để hiển thị số node chưa cho vào Stack

Dùng vòng lặp while kết hợp với hàm logicinsert để thay đổi dần vị trí của các node từ đó tạo ra chuyển động của node trên màn hình console. Cụ thể :

So sánh giá trị x,y của node với giá trị xstop và ESP, thay đổi x,y cho đến khi x=xstop và y=ESP. Sau mỗi vòng lặp gọi hàm setup để vẽ lại vị trí của node bị thay đổi.

Sau khi di chuyển xong cho node thứ nCur vào stack đồng thời giảm nCur 1 đơn vị (giảm số node chưa cho vào stack).

Trường hợp nCur<0 (không còn node nào chưa cho vào stack) thì hiện thông báo không thể insert.

* Hàm deleterun : tương tự hàm insertrun, sử dụng cấu trúc while so sánh x,y với các giá trị xdelete,ydelete để tạo chuyển động node đi ra khỏi stack. Đồng thời xóa Node nstack (top) ra khỏi stack. Và gán trường hide của node đó bằng true để xóa node khỏi màn hình console

Điều kiện để xóa node khỏi stack là trong stack phải có ít nhất một node nên nếu nstack<0 thì không thực hiện và in thông báo không xóa được.

* Queue
* Hàm insertNode:

Sử dụng vòng lặp for để tăng dần các giá trị vị trí (x,y), sau mỗi vòng lặp gọi lệnh setupQueue để cập nhật lại vị trí của node trên console. Đồng thời đưa giá trị node đó vào queue và cập nhật lại giá trị rear của queue. Cập nhật lại biến ESP.

Sử dụng biến nCur để kiểm tra xem còn node nào chưa đưa vào không. Nếu nCur=0 thì thông báo không thêm được.

* Hàm deleteNode:

Điều kiện là queue hiện đang có ít nhất 1 node (Front<rear).

Sử dụng vòng lặp while cùng với hàm logicdelete() để thay đổi vị trí của node cho đến khi no được đưa đến vị trí mong muốn. sau mỗi vòng lặp gọi hàm setupqueue để cập nhật vị trí node. Gán trạng thái ẩn của node đó bằng true để xóa khỏi console, cập nhật lại giá trị ESP và front của queue.

* Deque
* Hàm insertNodeFront(), insertNodeRear():

Sử dụng biến nCur để quản lí số node chưa đưa vào dequeue. Nếu nCur=0 thì thông báo không thể thêm.

Sử dụng vòng lặp for để thay đổi giá trị vị trí của các node đến các vị trị cần đến, sau mỗi vòng lặp gọi hàm setupDequeue() để cập nhật vị trí của node trên màn hình.

Đưa giá trị Node vào Dequeue đồng thời cập nhật các giá trị rear, front.

* Hàm deleteNodeFront(), deleteNodeNear():

Tương tự hàm delete của queue, sử dụng while và hàm logicDelete( logicDeleteFront(), logicDeleteRear()) để thay đổi vị trí của node cần xóa. Đồng thời cập nhật lại các giá trị front, rear, ESP.

* Link List:
* Hàm creatNulll(): để thêm node NULL vào cuối danh sách.
* Hàm insertNode(): Đưa node vào list ở vị trí cần chèn. Khi đó các node trong list sẽ có sự thay đổi về vị trí. Gọi hàm logicIns để đẩy các node về bên phải và hàm showNode để cập nhật vị trí các node.

\*ở bước chèn node vào list nếu vị trí cần chèn nếu vị trí cần chèn lớn hơn số node trong list thì sẽ gán vị trí đó vào cuối list.

* Hàm deleteNode(): sử dụng hàm logicDelete để dịch tất cả cac node phía sau node cần xóa về bên phải. chồng vào vị trí node cần xóa.
* Xóa node k khỏi list

\* Cần kiểm tra điều kiện vị trí cần xóa có phù hợp không (>0 và <nList)

* Circle List:
* Hàm arowUp: vẽ lại mũi tên liên kết giữa node đầu với node cuối.
* Hàm insert và delete tương tự linklist, chỉ thêm hàm vẽ mũi tên liên kết giữa node đầu và cuối.

## Thuật toán và chương trình mô phỏng sắp xếp topo

## Lí thuyết:

* Thuật toán sắp xếp Topo là phương pháp sắp xếp các đối tượng có mối liên hệ phụ thuộc với nhau.

Vd: sắp xếp thứ tự các môn chính trị trong chương trình học, sắp xếp thứ tự lắp ráp các linh kiện của một máy (phải lắp linh kiện này trước rồi mới được lắp linh kiện kia).

* Điều kiện để tồn tại thứ tự Topo:

Nếu biểu diễn mối liên hệ giữa các đối tượng thành một đồ thị có hướng thì điều kiện để có thứ tự Topo là đồ thị đó không có chu trình.

Vd: 1 đồ thị có chu trình như hình trên. Ta thấy đối tượng 1 cần hoàn thành trước đối 3.

Đối tượng 3 phải hoàn thành trước đối tượng 2 nhưng đối tương 2 lại phải hoàn thành trước đối tượng 1. Như vậy không có cách sắp xếp nào hợp lí cả.

* Input: một danh sách n đối tượng và mối liên hệ giữa các đối tượng.
* Output: một danh sách n đối tượng đã được sắp xếp theo thứ tự topo.

1. **Chi tiết code**

File Topo.cpp (thuật toán):

Xây dựng struct Node gồm các trường:

* int data: chứa giá trị của đối tượng (đại diện cho đối tượng)
* int suc: đếm số đối tượng cần hoàn thành trước đối tượng hiện tại
* Int count: đếm số đối tượng cần hoàn thành trước đối tượng hiện tại mà chưa được xóa khỏi đồ thị
* Int top[]: chứa danh sách các đối tượng cần thực hiện trước đối tượng hiện tại
* Bool visit: đánh dấu đối tượng đã được đưa ra khỏi đồ thị hay chưa.

Các hàm trong chương trình:

* Hàm Input(): gọi để thực hiện nhập dữ kiện đầu vào. Trong đó cần thực hiện hàm NhapDanhSach để thực hiện nhập các đối tượng cần thực hiện trước đối tượng i.
* Hàm XoaDoiTuong: dùng để xóa 1 đối tượng ra khỏi đồ thị đồng thời xóa các cung chỉ tới đối tượng đó. Hàm được sử dụng khi 1 đối tượng có Count=0 và được int ra danh sách topo.

Sử dụng vòng lặp for duyệt qua tất các các đối tượng để kiểm tra đối tượng vừa xóa có xuất hiện trong danh sách top[] không. Nếu có thì giảm count của đối tượng đó.

Cập nhật trường Visit=True cho đối tượng vừa xóa.

* Hàm Toposort: sử dụng vòng lặp while và for lòng nhau để duyệt qua các đối tượng, cho đến khi tất cả đối tượng được đưa vào mảng sort[].

Đưa các đối tượng có count=0 vào mảng sort[] đồng thời gọi hàm XoaDoiTuong để xóa ra khỏi đồ thị và cập nhật cout của các đối tượng.