****Câu 1:****Trong khoa học máy tính, **danh sách đặc** được hiểu như thế nào? Cho ví dụ.

- Là danh sách mà các phần tử trong danh sách có cùng kiểu dữ liệu, và được cấp phát liên tục trong bộ nhớ.

Ví dụ:

# define MAX 100

int a[MAX];

int n; // n là tổng số phần tử hiện có trong danh sách, 0 <=n<= MAX

Trong đó:‘MAX’ là độ dài tối đa của danh sách đặc

****Câu 2:**** Trong khoa học máy tính, **danh sách liên kết** được hiểu như thế nào? Có mấy loại? Cho ví dụ.

- là một danh sách mà các phần tử được cấp phát rời rạc nhau, và cố định trong bộ nhớ. Mỗi Phần tử trong danh sách gồm có 2 thành phần:

* Phần 1: vùng thông tin chưa giá trị cần quản lý
* Phần 2: vùng liên kết, chứa địa chỉ bộ nhớ của phần tử kế tiếp

****Câu 3:**** Tại sao nói **STACK** và **QUEUE** là danh sách hạn chế? Cho ví dụ?

- Do số lượng nhỏ các hoạt động có thể được thực hiện.

Ví dụ:

* Stack:

# define MAX 100

int a[MAX];

int sp; // đỉnh stack

* Queue:

void init(int a[], int &front, int &rear)

{

front = -1;

rear = -1;

}

****Câu 4:**** Thế nào là **LIFO**, **FIFO**? Cho ví dụ.

- LIFO(Last In First Out): Phần tử được thêm vào ngăn xếp sau, sẽ được lấy ra (xóa) khỏi ngăn xếp trước.

Ví dụ: bool isFull()

{

return top == size ? true : false;

}

- FIFO(First In First Out): Phần tử được thêm vào trước, sẽ được lấy ra (xóa) trước.

Ví dụ: bool isFull()

{

return rear == size - 1 ? true : false;

}

****Câu 5:**** Hãy so sánh **STACK** và **QUEUE**..

****- Stack**** và ****Queue**** là các cấu trúc dữ liệu được sử dụng để lưu trữ các yếu tố dữ liệu và nó dựa trên một số các ví dụ có thực trong cuộc sống hàng ngày của chúng ta.

|  |  |  |
| --- | --- | --- |
| **Cơ sở để so sánh** | **STACK** | **QUEUE** |
| Nguyên tắc làm việc | LIFO (Last in First out) | FIFO (First in First out) |
| Structure | Dùng một đầu để chèn và xóa các phần tử dữ liệu | Có 2 đầu để xử lý dữ liệu, một đầu chèn một đầu xóa |
| Số con trỏ được sử dụng | Một | Hai (Trong trường hợp đơn giản) |
| Hoạt động được thực hiện | Push và Pop | Enqueue và dequeue |
| Kiểm tra empty condition | Top == -1 | Front == -1 |
| Examination full condition | Top == Max - 1 | Rear == Max - 1 |
| Biến thể | Không có biến thể | Nó có các biến thể như hàng đợi tròn, hàng đợi ưu tiên, hàng đợi kết thúc gấp đôi. |
| Thực hiện | Đơn giản | Tương đối phức tạp |

****Câu 6:**** Theo bạn, danh sách danh sách liên kết có thể ứng dụng xử lý các vấn đề gì trong máy tính?

- Danh sách liên kết đơn chủ yếu được dùng để xây dựng các loại cấu trúc dữ liệu khác như ngăn xếp hoặc hàng đợi hoặc đồ thị …,In danh sách theo vị trí đảo ngược.Phát hiện điểm nối của danh sách liên kết,…

****Câu 7:**** Thế nào là cấu trúc dữ liệu động? Cho ví dụ.

- Danh sách liên kết:

* Cấp phát động lúc chạy chương trình
* Các phần tử nằm rải rác ở nhiều nơi trong bộ nhớ
* Kích thước danh sách chỉ bị giới hạn do RAM
* Thao tác thêm xoá đơn giản

- Cần xây dựng cấu trúc dữ liệu đáp ứng được các yêu cầu:

* Linh động hơn.
* Có thể thay đổi kích thước, cấu trúc trong suốt thời gian sống.

Cấu trúc dữ liệu động

Ví dụ:

int \*p1, \*p2;

// cấp phát vùng nhớ cho 1 biến động kiểu int

p1 = (int\*)malloc(sizeof(int));

\*p1 = 5; // đặt giá trị 5 cho biến động đang được p1 quản lý

// cấp phát biến động kiểu mảng gồm 10 phần tử kiểu int

p2 = (int\*)calloc(10, sizeof(int));

\*(p2+3) = 0; // đặt giá trị 0 cho phần tử thứ 4 của mảng p2

free(p1);

free(p2);