ÔN TẬP CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

# PHẦN I: KIẾN THỨC (thi viết trên giấy) 1/ Thiết kế và phân tích giải thuật

* Trình bày khái niệm giải thuật và các tính chất của giải thuật.

(khái niệm)Giải thuật hay thuật toán(là 1 chuỗi hữu hạn các lệnh. Mỗi lệnh có 1 ngữ nghĩa rõ ràng và có thể đc thực hiện vs 1 lượng hữu hạn dài nguyên trong 1 khoảng hữu hạn thời gian

Tính chất của thuật toán

1. Xác định rõ đầu vào ,đầu ra
2. Hữu hạn
3. Chuẩn xác cao
4. Tổng quát

* Trình bày khái niệm cấu trúc dữ liệu và các tiêu chuẩn khi lựa chọn cấu trúc dữ liệu.

Khái niệm:

**Cấu trúc dữ liệu(Data Structure):** là cách lập trình để lưu trữ dữ liệu để dữ liệu có thể được sử dụng một cách hiệu quả. Hầu hết mọi ứng dụng doanh nghiệp đều sử dụng nhiều kiểu cấu trúc dữ liệu khác nhau theo cách này hay cách khác, vì nó mang lại nhiều lợi ích rất lớn không chỉ cho việc lưu trữ dữ liệu.

Các tiêu chuẩn khi lựa chọn cấu trúc dữ liệu

1. CTDLiệu phải biểu diễn đc đầy đủ các thông tin nhập và xuất của bài toán
2. CTDLiệu phải phù hợp với các thao tác của thuật toán mà ta lựa chọn để giải quyết bài toán
3. CTDLiệu phải cài đặt đc trên MTính vs ngôn ngữ lập trình đang sd.

* Trình bày khái niệm đệ quy, giải thuật đệ quy và các đặc điểm của giải thuật đệ quy.

Khái niệm đệ quy:

Một đối tượng được gọi là đệ quy nếu nó or 1 phần của nó đc định nghĩa thông qua khái niệm về nó

Khái niệm giải thuật đệ quy:

- Nếu lời giải của bài toán P được thực hiện = lời giải của bài toán P’ giống như P thì lời giải này được gọi là lời giải đệ quy

- Giải thuật tg ứng vs lời giải bài toán P đc gọi là giải thuật đệ quy

Đặc điểm:

* Trong hàm đệ quy có lời gọi đến chính hàm đó
* Sau mỗi lần có lời gọi đệ quy thì kích thước của bài toán được thu nhỏ hơn trước
* Có ít nhất một TH suy biến xảy ra. Khi đó bài toán sữ được giải quyết theo 1 cách khác, vc gọi đệ quy kết thúc

* Trình bày giải thuật đệ quy tính n!.

***#include <stdio.h>***

***long long fact(int n)***

***{***

***if(n == 0)***

***return 1;***

***return n \* fact(n-1);***

***}***

***int main()***

***{***

***int n;***

***printf("nhap n = ");***

***scanf("%d", &n);***

***printf("giai thua cua %d! = %lld\n", n, fact(n));***

***return 0;***

***}***

* Hàm đệ quy (viết bằng ngôn ngữ C) dưới đây cho kết quả là gì? Giải thích tại sao?

int Tinh(int n, int x)

{

if (n == 1) Tinh = x;

else Tinh = n\* Tinh(n-1,x); return Tinh;

}

***Cho kết quả trả về là giá trị của Tinh.***

***Nếu nhập n= 1 thì Tinh sẽ bằng x***

***Nếu n #1 thì Tinh sẽ bằng n\*Tinh(n-1,x)***

* Cho biết số Fibonacci F11 = 89 và F12 = 144

+ Hãy tính F16 = ?. Viết công thức dãy số Fibonacci;

F11=89;

F12=144;

F13=233;

F14= 377;

F15=610;

F16=987;(1 số bằng tổng của kết quả 2 số trc đó)

Công thức truy hồi của dãy fibonacci có dạng: f(n) = f(n-1) + f(n-2) .

+ Viết một thủ tục đệ quy để tính và in ra n số Fibonacci đầu tiên.

***# include<stdio.h>***

***# include<string.h>***

***int fibonacci(int val)***

***{***

***if(val<0)***

***{***

***return -1;***

***}***

***else if(val==0 ||val==1)***

***{***

***return val;***

***}***

***else***

***{***

***return fibonacci(val - 1)+fibonacci(val - 2);***

***}***

***}***

***int main()***

***{***

***float inputNum;***

***int i;***

***printf("\nNhap so phan tu trong day Fibo:");***

***scanf("%f",&inputNum);***

***printf("Day so Fibo dau tien cua ban la: \n");***

***for (i=0;i< inputNum;i++)***

***{***

***printf("%d ",fibonacci(i));***

***}***

***return 0;***

***}***

# 2/ Danh sách:

Trình bày khái niệm danh sách, và các cách cài đặt danh sách, cho biết ưu nhược điểm của từng dạng cài đặt.( https://v1study.com/giai-thuat-va-lap-trinh-c-i-kieu-du-lieu-danh-sach-list.html#gsc.tab=0)

Khái niệm:

***Danh sách là một cấu trúc dữ liệu gồm một hữu hạn các phần tử có kiểu dữ liệu xác định và giữa các phần tử có mỗi liên hệ với nhau;***

***Trên phương diện toán học, danh sách là tập hợp hữu hạn các phần tử, có thứ tự và giữa chúng có mối liên hệ tuyến tính.***

Cách cài đặt:

1. Cài đặt danh sách bằng mảng (danh sách đặc)

Ưu điểm:

* Tiết kiệm bộ nhớ hơn(chỉ đối vs mảng động)
* Truy xuất tới 1 phần tử sẽ nhanh hơn ( nhờ vào toán tử lấy chỉ số):vd:a[3]

Nhược điểm:

* Các phần tử phải liên tục nhau,cho nên giả sử cần 5 ô nhưng cần trúng đúng 5 ô nhưng 5 ô đó không liên tục nhau => ko dùng mảng đc
* Luôn phải yêu cầu cấp trc 1 slg nào đó( vd:mảng tỉnh là 100,mảng động là n)
* Thao tác thêm/xóa phải lùi lại hay tiến lên rất nhìu phần tử =>chương trình chạy chậm

1. Cài đặt danh sách bằng con trỏ ( danh sách liên kết)

Ưu điểm:

* Các phần tử ko cần liên tục;khắc phục tình trạng phân mảnh bộ nhớ
* Ko cần cấp phát trc slg như bên mảng, dữ liệu luôn được ở trạng thái động(khi nào cần thì cứ thêm vào như bình thường, ko cần phải cấp lại bộ nhớ như bên mảng )
* Các thao tác thêm/xóa ko cần phải lùi lại hay tiến lên các phần tử từ vị trí đó trở xuống như bên mảng mà ta chỉ cần bẻ lại mối lk giữa 2 node là đc=>chạy nhanh hơn bên mảng trog TH này

Nhược điểm:

* 1 node trong dslk sẽ lưu 2 thông tin đi kèm nó là data và 1 con trỏ pnext=>đòi hỏi bộ nhớ phải tốn nhìu hơn mảng
* Muốn truy xuất tới 1 node cụ thể thì ta phải đi từ đầu cho đến đó ỏ đi từ cuối cho đến đó
* Nếu như dslk mà vô tình tại 1 node nào đó bị hư là hư hết tất cả ds bởi vì nó truyền dl liên tục cho nhau

1. Cài đặt bằng con nháy

# 3/ Cây:

* Trình bày khái niệm cây và các tính chất của cây;

Khái niệm:

**Cây là 1 tập hợp hữu hạn của các phần tử(mỗi phần tử được gọi là 1 NODE) hay còn gọi là 1 đỉnh và 1 tập hợp hữu hạn các cạnh có hướng nối các nút lại vs nhau gọi là nhánh.**

Tính chất:

* **Mỗi nhánh có 2 nút ở đầu**
* **Với mỗi nút bất kỳ thì nút trên gọi là cha, nút dưới gọi là con**
* **Có 1 nút duy nhất ko có nút cha,gọi là nút gốc.Mọi nút khác có đúng 1 nút cha.**
* **Các nút nằm cùng cấp gọi là nút anh em**
* **Mọi nút khác đc nối vs gốc = 1 đường duy nhất bắt đầu từ gốc và kết thúc từ nút đó.**
* Thế nào là cây nhị phân, hãy vẽ cây nhị phân biểu diễn biểu thức toán học sau: 3\*4+(12+b)/6;

Khái niệm:

* **Là cây mà mỗi nút có tối đa 2 con: con trái và con phải**
* **Lchild và Rchild là các con trỏ chỉ đến nút con bên trái và nút con bên phải**
* **Lchild và Rchild là con trỏ rỗng nếu ko có nút con trái or nút con phải**
* **Do vị trí nút trên cây nên cây nhị phân đc xem là cây có thứ tự.**
* **Từ một cây tổng quát n phân ta có thể chuyển về cây nhị phân**
* **Có thể định nghĩa cây nhị phân như là 1 cấu trúc dữ liệu đệ qui**

Trình bày phương pháp duyệt cây nhị phân theo thứ tự trước và lấy ví dụ minh họa;

Phương pháp:

1. **Nếu cây rỗng thì ko làm gì.**
2. **Nếu cây ko rỗng thì:**

* **Thăm gốc**
* **Duyệt cây con trái Aleft theo thứ tự trc**
* **Duyệt cây con phải Aright theo thứ tự trước**

**A->B->D->E->C->F->G**

* Trình bày phương pháp duyệt cây nhị phân theo thứ tự giữa và lấy ví dụ minh họa.

Phương pháp:

1. **Nếu cây rỗng thì ko làm gì.**
2. **Nếu cây ko rỗng thì:**

* **Duyệt cây con trái Aleft theo thứ tự giữa**
* **Thăm gốc**
* **Duyệt cây con phải Aright theo thứ tự giữa**

**D->B->E->A->F->C->G**

* Trình bày phương pháp duyệt cây nhị phân theo thứ tự sau và lấy ví dụ minh họa.

Phương pháp:

1. **Nếu cây rỗng thì ko làm gì.**
2. **Nếu cây ko rỗng thì:**

* **Duyệt cây con trái Aleft theo thứ tự sau**
* **Duyệt cây con phải Aright theo thứ tự giữa**
* **Thăm gốc**

D->E->B->F->G->C->A

# PHẦN II: THỰC HÀNH (thực hành trên máy)

1/ Viết chương trình tìm 10 số Fibonacci đầu tiên sử dụng phương pháp đệ quy.

***# include<stdio.h>***

***# include<string.h>***

***int fibonacci(int val)***

***{***

***if(val<0)***

***{***

***return -1;***

***}***

***else if(val==0 ||val==1)***

***{***

***return val;***

***}***

***else***

***{***

***return fibonacci(val - 1)+fibonacci(val - 2);***

***}***

***}***

***int main()***

***{***

***float inputNum;***

***int i;***

***printf("\nNhap so phan tu trong day Fibo:");***

***scanf("%f",&inputNum);***

***printf("Day so Fibo cua ban la: \n");***

***for (i=0;i< inputNum;i++)***

***{***

***printf("%d ",fibonacci(i));***

***}***

***return 0;***

***}***

2/Viết chương trình nhập vào một dãy số nguyên gồm n phần tử. Hãy sắp xếp dãy số đó theo chiều tăng dần sử dụng thuật toán:

* Sắp xếp chọn (Selection Sort).
* Sắp xếp chèn (Insertion Sort).
* Sắp xếp nổi bọt (Bubble Sort) Xuất ra dãy số nguyên đã được sắp xếp.

***#include<stdio.h>***

***void Nhapmang(int a[],int n)***

***{***

***for(int i = 0;i< n; ++i)***

***{***

***printf("Nhap phan tu a[%d] =",i);***

***scanf("%d",&a[i]);***

***}***

***}***

***void selection\_sort(int a[],int n)***

***{***

***int i,j,min,temp;***

***for (i=0;i<n-1;i++)***

***{***

***min=i;***

***for(j=i+1;j<n;j++)***

***{***

***if(a[j]<a[min])***

***min=j;***

***}***

***temp=a[i];***

***a[i]=a[min];***

***a[min]=temp;***

***}***

***}***

***void insertion\_sort(int a[],int n)***

***{***

***int i,j,tmp;***

***for(i=1;i<n;i++)***

***{***

***j=i-1;***

***tmp=a[i];***

***while(tmp<a[j]&&j>=0)***

***{***

***a[j+1]=a[j];***

***j--;***

***}***

***a[j+1]=tmp;***

***}***

***}***

***void bubble\_sort(int a[],int n)***

***{***

***int i,j,tmp;***

***for (i=0;i<n-1;i++)***

***{***

***for (j=n-1;j>i;j--)***

***{***

***if(a[j]<a[j-1])***

***{***

***tmp=a[j-1];***

***a[j-1]=a[j];***

***a[j]=tmp;***

***}***

***}***

***}***

***}***

***void Xuatmang( int a[],int n)***

***{***

***for (int i =0;i<n;++i)***

***{***

***printf("%d\t",a[i]);***

***}***

***}***

***int main()***

***{***

***// int n;***

***// int a[n];***

***// printf("nhap so luong phan tu trong mang=");***

***// scanf("%d",&n);***

***// for(int i=0;i<n;i++)***

***// {***

***// printf("nhap phan tu thu a[%d]=",i);***

***// scanf("%d",&a[i]);***

***// }***

***//***

***// printf("mang tang dan:\t");***

***// selection\_sort(a,n);***

***// Xuatmang(a,n);***

***int a[50];***

***int n;***

***do***

***{***

***printf("Nhap so luong phan tu ");***

***scanf("%d",&n);***

***}***

***while (n<=0 || n>=50);***

***Nhapmang(a,n);***

***printf("sap xep chon la:\t");***

***selection\_sort(a,n);***

***Xuatmang(a,n);***

***printf("\n\n");***

***printf("sap xep chen la:\t");***

***insertion\_sort(a,n);***

***Xuatmang(a,n);***

***printf("\n\n");***

***printf("sap xep noi bot la:\t");***

***bubble\_sort(a,n);***

***Xuatmang(a,n);***

***}***

3/ Viết chương trình nhập vào một mảng các số nguyên, in ra màn hình phần tử có giá trị nhỏ nhất và các giá trị lớn nhất.

***#include <stdio.h>***

***void nhap(int a[], int n)***

***{***

***for (int i = 0; i < n; i++)***

***{***

***printf("Nhap vao phan tu a[%d]: ", i);***

***scanf("%d", &a[i]);***

***}***

***}***

***int max(int a[], int n)***

***{***

***int max = a[0];***

***for (int i = 1; i < n; i++)***

***if (max < a[i])***

***max = a[i];***

***return max;***

***}***

***int min(int a[], int n)***

***{***

***int min = a[0];***

***for (int i = 1; i < n; i++)***

***if (min > a[i])***

***min = a[i];***

***return min;***

***}***

***void Xuatmang( int a[],int n)***

***{***

***for (int i =0;i<n;++i)***

***{***

***printf("%d\t",a[i]);***

***}***

***}***

***int main()***

***{***

***int a[1000];***

***int n;***

***printf("\nNhap so luong phan tu trong mang = ");***

***scanf("%d", &n);***

***nhap(a, n);***

***printf("mang vua nhap la:");***

***Xuatmang(a,n);***

***printf("\nMax = %d", max(a, n));***

***printf("\nMin = %d", min(a, n));***

***return 0;***

***}***

4/ Cho khai báo cấu trúc danh sách liên kết đơn số nguyên và các hàm được cài đặt sẵn bằng ngôn ngữ C như sau:

struct Node

{

int data;

Node \*pNext;

};

struct List

{

Node \*pHead;

Node \*pTail;

};

* Xây dựng hàm khởi tạo một danh sách rỗng.

***void Init(LIST&l){***

***l.pHead=l.pTail=NULL;***

***};***

* Xây dựng hàm thêm một node có tên là pNew vào đầu danh sách List.

***void AddHead(LIST &l,NODE \*p)***

***{***

***if(l.pHead==NULL)***

***l.pHead=l.pTail=p;***

***else***

***{***

***p->pNext=l.pHead;***

***l.pHead=p;***

***}***

***}***

* Xây dựng hàm thêm một node có tên là pNew vào cuối danh sách List.

***void AddTail(LIST &l,NODE \*p)***

***{***

***if(l.pHead==NULL)***

***l.pHead=l.pTail=p;***

***else***

***{***

***l.pTail->pNext=p;***

***l.pTail=p;***

***}***

***}***

5/ Cho danh sách liên kết đơn gồm các phần tử là số nguyên, viết chương trình thực hiện các yêu cầu sau:

* Nhập và xuất danh sách ra màn hình;
* Liệt kê các phần tử mang giá trị chẵn;
* Liệt kê các phần tử mang giá trị lẻ.

***#include<stdio.h>***

***/\*dinh nghia 1 cau truc NODE:khai bao\*/***

***struct NODE{***

***int data;***

***struct NODE \*pNext;***

***};***

***//typedef struct NODE\*node;***

***/\*dinh nghia 1 cau truc LIST: kha b?o \*/***

***struct LIST{***

***NODE\*pHead;***

***NODE\*pTail;***

***};***

***/\*khoi tao 1 ds rong\*/***

***void Init(LIST&l){***

***l.pHead=l.pTail=NULL;***

***};***

***/\* tao 1 node bat ki\*/***

***NODE\* CreateNode()***

***{***

***int x;***

***printf("nhap gia tri cho node: ");***

***scanf("%d",&x);***

***NODE \*p=new NODE;***

***p->data=x;***

***p->pNext=NULL;***

***return p;***

***}***

***/\*chèn node vao cuoi ds\*/***

***void AddTail(LIST &l,NODE \*p)***

***{***

***if(l.pHead==NULL)***

***l.pHead=l.pTail=p;***

***else***

***{***

***l.pTail->pNext=p;***

***l.pTail=p;***

***}***

***}***

***void xuatchan(LIST &l)***

***{***

***NODE\*p=l.pHead;***

***while(p)***

***{***

***if(p->data%2==0)***

***printf("%d->",p->data);***

***p=p->pNext;***

***}***

***}***

***void xuatle(LIST &l)***

***{***

***NODE\*p=l.pHead;***

***while(p)***

***{***

***if(p->data%2!=0)***

***printf("%d->",p->data);***

***p=p->pNext;***

***}***

***}***

***int main()***

***{***

***LIST l;***

***Init(l);***

***int n;***

***printf("nhap so luong phan tu cua danh sach:");***

***scanf("%d",&n);***

***int i=1;***

***while(i<=n)***

***{***

***NODE \*p=new NODE;***

***p=CreateNode();***

***AddTail(l, p);***

***i++;***

***}***

***printf("ds vua nhap la:");***

***for (NODE \*p=l.pHead;p!=NULL;p=p->pNext)***

***printf("%d\t",p->data);***

***printf("cac so chan");***

***xuatchan(l);***

***printf("\tcac so le");***

***xuatle(l);***

***}***