**Tên:** Đinh Phạm Thảo Trân

**Lớp:** IT81

**Câu hỏi chương 1:**

Cấu trúc dữ liệu và giải thuật

*Câu 1:*

* Cấu trúc dữ liệu là cấu trúc (sự tổ chức) của dữ liệu/thông tin trên máy tính, mà ở đó với cấu trúc này máy tính có thể sử lý được.
* Vd: Cấu trúc dữ liệu cơ bản của một sinh viên (mã số, họ và tên, giới tính, ngày sinh, địa chỉ)

**Trong đó:**

Mã số sinh viên, họ và tên, địa chỉ có kiểu dữ liệu là kiểu chuỗi.

Ngày sinh của sinh viên có kiểu dữ liệu là kiểu date.

*Câu 2:*

* Giải thuật là tập hữu hạn các bước (chỉ thị hay hành động) theo một trình tự, được xác định rõ ràng nhằm mục đích để giải quyết một bài toán nào đó (dựa vào những giá trị đầu vào gọi là input và cho ra kết quả đầu ra gọi là output)
* Vd: Ta có bài toán: Tìm nghiệm của phương trình bậc hai một ẩn có dạng:

Ta có giải thuật (T) để giải bài taosn tìm nghiệm cho phương trình: như sau:

**Giải thuật (T):**

Đầu vào (input): a, b,c (a,b, c R)

Đầu ra (output): kết luận nghiệm.

Bước 1: tính

Bước 2: thực hiện kiểm tra delta:

2.1 nếu delta < 0 thì

Phương trình vô nghiệm

2.2 nếu delta = 0 thì:

Phương trình có nghiệm kép x1=x2= -b/2a

2.3 nếu delta > 0 thì

Phương trình có hai nghiệm phân biệt

*Câu 3:*

* Trong một đề án tin học, giải thuật và cấu trúc dữ liệu có mốt liên hệ chặc chẽ với nhau, được thể hiện qua công thức:

Cấu trúc dữ liệu + giải thuật = Chương trình

Với một cấu trúc dữ liệu đã chọn, sẽ có những thuật giải tương ứng, phù hợp. Khi cấu trúc dữ liệu thay đổi thường giải thuật cũng phải thay đổi theo để tránh gượng ép, thiếu tự nhiên trên một cấu trúc không phù hợp. Hơn nữa, một cấu trúc dữ liệu tốt có thể giúp giải thuật xử lý trên đó có thể phát huy tác dụng tốt hơn, vừa đáp ứng nhanh vừa tiết kiệm vật tư, giải thuật cũng dễ hiểu và đơn giản hơn.

* Vd: cho một chương trình quản lý điểm thi của 3 sinh viên. Mỗi sinh viên có 3 điểm tương ứng với 3 môn học khác nhau như sau:

|  |  |  |  |
| --- | --- | --- | --- |
| Sinh viên | Môn 1 | Môn 2 | Môn 3 |
| SV1 | 7 | 9 | 5 |
| SV2 | 5 | 0 | 9 |
| SV3 | 6 | 3 | 7 |

Xét về thao tác xuất điểm số các môn của từng sinh viên:

* Phương án 1: sử dụng mảng một chiều

*Cấu trúc dữ liệu:* Có tất cả 3(SV)\*3(môn) = 9 điểm số cần lưu trữ, khi đó khai báo result như sau:

Int result [9] = {7, 9, 5, 5, 0, 9, 6, 3, 7}

*Giải thuật:* (giả sử vị trí hàng và cột của người dùng như của người lập trinh)

Để truy xuất điểm số môn j của sinh viên i – phần tử tai dòng I cột j trong bảng – phải sử dụng một công thức xác định chỉ số tương ứng trong mảng result:

Bảng điểm (dòng I, cột j) => result {[(i – 1)\*số cột] + j}

Ngược lại nếu muốn biết một điểm số bất kỳ là của ai, môn gì, ta có công thức sau:

Result [i] => bảng điểm (dòng (i/số cột + 1), cột (I % số cột))

* Phương án 2: Sử dụng mảng hai chiều:

*Cấu trúc dữ liệu:* mảng hai chiều có kích thước 3 dòng \* 3 cột.

Int result [3][3] = {{7, 9, 5},{5, 0, 9},{6, 3, 7}};

*Giải thuật:*

Truy xuất điểm số môn j của sinh viên thứ I cũng chính là phần tử nằm ở dòng I cột j trong bảng (giả sử i, j của người dùng như lập trình)

Bảng điểm (dòng i, cột j) => result [i][j]

Kết luận: cả hai phương án đều có thể giải quyết được vấn đề của bài toán. Nhưng đối với phương án một thì việc giải quyết bài toán trở nên phức tạp hơn, còn phương án hai thì đơn giản và tự nhiên hơn.

*Câu 4:*

While (j > 0 && a[j – 1] > x)

…

Với i = 1 có 3 phép ss.

Với i = 2 có 5 phép ss.

Với i = 3 có 7 phép ss.

…

Với i = n có 2n + 1 phép ss.

For (i = 1; i <; i++)

Với i = 1 có 1 phép ss

Với i = 2 có 1 phép ss

…

Với i = n có n phép ss.

Nên với giải thuật trên có n \* (2n + 1) **số phép ss**.

**Bài tập chương 1:**

Cấu trúc dữ liệu và giải thuật

*Bài 1:*

* Số phép gán:

Trong đoạn code

For (i = 0; i < n; i++)

Có (n + 1) + n phép gán

For (j = 0; j < m; j++)

Có (m + 1) + m phép gán

If (a[i][j] == x) return 1;

Có 1 phép gán.

Return -1;

* Đoạn code trên có ((n + 1) + n) \* ((m + 1) + m )\*1 = (2n + 1) \* (2m + 1) số phép gán
* Giả sử m = n ->> ((n + 1) + n) \*((n + 1) + n) = (2n + 1)\*(2n + 1) = số phép gán
* Số phép ss:

Trong đoạn code

For (i = 0; i < n; i++)

Có n phép ss

For (j = 0; j < m; j++)

Có m phép ss

If (a[i][j] == x) return 1;

Return -1;

* Đoạn code trên có n \* m phép ss
* Giả sử m = n thì ta có n \* n = phép ss

Độ phức tạp trong đoạn code trên là: O(n2)

*Bài 2:*

* Số phép gán:

Sum = 0;

Có 1 phép gán

For (i = 0; i < n; i++)

Có 2n + 1 phép gán

For (j = 0; j < i; j++)

Với i = 1 có 3 phép gán

Với i = 2 có 5 phép gán

Với i =3 có 7 phép gán

…

Với i = k có 2k + 1 phép gán

Với i = n có 2n + 1 phép gán

Sum++;

Có 1 phép gán

* Có 1 + (2n + 1)\*((2n + 1)\*1) = phép gán
* Số phép ss:

Sum = 0;

For (i = 0; i < n; i++)

Có n + 1 phép ss

For (j = 0; j < i; j++)

Có n + 1 phép ss

Sum++;

* Đoạn code trên có n \* n = n2 phép ss.

Độ phức tạp của đoạn code trên là: O(n2)

*Bài 3:*

For (i = 0; i < n; i++)

Sum1 += i;

For (i = 0; i < n; i++)

Sum2 += i;

Đoạn code trên có 2 \* (n + 1) = 2n + 2 phép ss

Có 2 \* (2n + 1) = 4n + 2 phép gán

*Độ phức tạp O(n)*

*Bài 4:*

Int GT (int n)

{

If (n == 1)

Return 1;

Return n \* GT(n – 1);

}

Đoạn code trên có n phép ss

Độ phức tạp O(n)

*Bài 5:*

Int Fibo (int n)

{

If (n <= 1)

Return n;

Return Fibo(n – 1) + (n – 2);

}

Đoạn code trên có n phép ss

Độ phức tạp O(n)