Bài 1

* CTDL là cấu trúc (sự tổ chức) của dữ liệu/thông tin lên trên máy tính, mà ở đó với cấu trúc này máy tính có thể xử lý được.
* Cấu trúc này phải rõ ràng,xác định, các thành phần bên trong cấu trúc phải rõ ràng, và xcs định.
* Ví Dụ:
  1. Cấu trúc dữ liệu cơ bản của một sinh viên (mã số sv, họ và tên, giới tính, ngày sinh, địa chỉ).
     + **Trong đó**
     + Mã số sv, họ và tên, đại chỉ có kiểu dữ liệu là  **kiểu chuỗi**
     + Ngày sinh của sinh viên có kiểu Date

Bài 2

* Giải thuật là một tập hợp hữu hạn của các bước (chỉ thị hay hành động) theo một trình tự, được xác định rõ ràngnhằm mục đích để giải quyết một bài toán nào đó (dựa vào những giá trị đầu vào là “**input**” và cho ra kết quả đầu ra gọi là “**output**”)
* Ví Dụ:

Trong kiến thức **Toán trung học cơ sở**, ta có bài toán: Tìm nghiệm phương trình bậc hai một ẩn có dạng a*x*2 + b*x* + c = 0 (với: a, b, c ∈ ; a ≠ 0).

\*\*\* Ta có **giải thuật (T)** để giải bài toán tìm nghiệm cho phương trình a*x*2 + b*x* + c = 0 như sau:

**Giải Thuật (T):**

Đầu vào (**input**): a, b, c (a, b, c, ∈ )

Đầu ra (**output**): kết luận nghiệm

Bước 1: tính delta = b2 – 4ac

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 =

2.3 Nếu delta > 0 thì

phương trình có hai nghiệm phân biệt:

x1=

x2 =

bài 3

**Cấu trúc dữ liệu + Giải thuật = Chương trình**

Với một cấu trúc dữ liệu đã chọn, sẽ có những giải thuật tương ứng, phù hợp. Khi cấu trúc dữ liệu thay đổi thường giải thuật cũng phải thay đổi theo để tránh việc xử lý gượng ép, thiếu tự nhiên trên một cấu trúc không phù hợp. Hơn nữa, một cấu trúc dữ liệu tốt sẽ 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 hiễu và đơn giản hơn.

**Ví dụ 1:** Một chương trình quản lý điểm thi của sinh viên cần lưu trữ các điểm số của 3 sinh viên. Do mỗi sinh viên có 4 điểm số ứng với 4 môn học khác nhau nên dữ liệu có dạng bảng như sau:

Chỉ xét thao tác xử lý là xuất điểm số các môn của từng sinh viên.

Giả sử có các phương án tổ chức lưu trữ sau:

Phương án 1 : Sử dụng mảng một chiều

Có tất cả 3(SV)\*4(Môn) = 12 điểm số cần lưu trữ, do đó khai báo mảng *result* như sau :

int result [ 12 ] = {7, 9, 5, 2, 5, 0, 9, 4, 6, 3, 7, 4};

Khi đó trong mảng *result* các phần tử sẽ được lưu trữ như sau

Và truy xuất điểm số môn j của sinh viên i – là phần tử tại (dòng i, cột j) trong bảng – phải sử dụng một công thức xác định chỉ số tương ứng trong mảng result:

bảngđiểm(dòng i, cột j) Þ result[((i-1)\*số cột) + j]

Ngược lại, với một phần tử bất kỳ trong mảng, muốn biết đó là điểm số của sinh viên nào, môn gì, phải dùng công thức xác định sau

result[ i ] Þ bảngđiểm (dòng((i / số cột) +1), cột (i % số cột) )

Với phương án này, thao tác xử lý được cài đặt như sau :

*void XuatDiem() //Xuất điểm số của tất cả sinh viên  
{*

*const int so\_mon = 4;  
int sv,mon;  
for (int i=0; i<12; i+)  
{*

*sv = i/so\_mon;  
mon = i % so\_mon;  
printf(“Điểm môn %d của sv %d là: %d”, mon, sv,result[i]);*

*}*

*}*

Bài 4

Gọi **a[j-1] > x (\*)** là phép toán cơ bản

Với **i = 1** thì j chạy 1 lần => (\*)**3** lần so sánh

Với **i = 2** thì j chạy 1 lần => (\*) **5** lần so sánh

…

Với **i = n** thì j chạy 1 lần => (\*) **2n+1** lần so sánh

* Thuộc phâm lớp **O(n)**