BÀI 1

TỔNG QUAN VỀ CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

Nội dung

1. Giải thuật và cấu trúc dữ liệu
2. Phân tích và đánh giá giải thuật
3. Ngôn ngôn ngữ diễn đạt giải thuật
4. Các cấu trúc dữ liệu cơ sở

1.1. Giải thuật và cấu trúc dữ liệu

- Thông tin và dữ liệu
- Khái niệm cấu trúc dữ liệu
- Khái niệm giải thuật
- Các đặc trưng của giải thuật
- Mối quan hệ giữa CTDL và giải thuật

1.1.1. Thông tin và dữ liệu

- **Thông tin là gì?**
  - **Là những tín hiệu, ký hiệu, hình ảnh tác động vào các giác quan đem lại sự hiểu biết cho con người.**
  - **Thông tin là nguồn gốc của nhận thức.**
- **Dữ liệu là gì?**
  - **Là những thông tin được lưu trữ trên các vật mang tin – Bộ nhớ máy tính.**

1.1.2. Khái niệm cấu trúc dữ liệu

- Dữ liệu được lưu trong bộ nhớ máy tính và được xử lý nên nó phải có cấu trúc
- Dữ liệu lớn được xây dựng từ các dữ liệu nguyên tử
- Cấu trúc dữ liệu là mô hình của dữ liệu được lưu trong bộ nhớ
- Trong các ngôn ngữ lập trình cấu trúc dữ liệu chính là các kiểu dữ liệu

1.1.3. Khái niệm giải thuật

- Các bước thực hiện khi một người muốn đi đến quán ăn tự phục vụ từ phòng học.

- Phòng học

- Rời phòng học

Đến cầu thang

Xuống tầng hầm

Đi đến quán ăn tự phục vụ

Cafeteria

Khái niệm giải thuật (tt)

- Giải thuật là dãy các bước có thứ tự chính xác để giải quyết được một bài toán cụ thể, theo đó với mỗi bộ dữ liệu vào giải thuật cho một kết quả
- Ví dụ: Giải phương trình bậc 2
  - Bước 1: Tính delta
  - Bước 2 so sánh delta với 0
    - > 0: tính 2 nghiệm x1 = …, x2 = … và thông báo nghiệm
    - = 0: tính nghiệm kép và thông báo
    - < 0: thông báo vô nghiệm

1.1.4. Các đặc trưng của giải thuật

- **Bộ dữ liệu vào:** Các dữ liệu mà giải thuật xử lý.
- **Bộ dữ liệu ra:** Là kết quả của việc thực hiện giải thuật, dữ liệu ra có quan hệ xác định với dữ liệu vào.
- **Tính tất định:** mỗi bước của giải thuật chỉ cho một kết quả duy nhất.
- **Tính dừng:** Sau hữu hạn bước giải thuật dừng lại và cho kết quả.
- **Tính đúng đắn:** Giải thuật thực sự giải quyết được yêu cầu của bài toán.
- **Tính phổ dụng:** Giải thuật giải quyết được một lớp bài toán.

1.1.5. Mối quan hệ giữa CTDL và GT

- Cấu trúc dữ liệu và giải thuật là hai phần của một bài toán.

Cấu trúc dữ liệu + Giải thuật = Chương trình

- Giải thuật là mã lệnh xử lý dữ liệu có cấu trúc định sẵn trong bộ nhớ và tạo ra dữ liệu mới.
- Giải thuật qui định cấu trúc dữ liệu và ngược lại.

Mối quan hệ giữa CTDL và GT (tt)

- **Ví dụ: Bài toán tìm max của 4 số nguyên**
  - **Dữ liệu vào: 4 số nguyên**
  - **Dữ liệu ra: 1 số nguyên**
  - **Thuật toán: Giải thuật Max:**
    - **Bước 1: Gán một số (đầu tiên) cho biến max.**
    - **Bước 2: So sánh max với các số còn lại, nếu max nhỏ hơn số nào thì gán max bằng số đó.**
    - **Bước 3: Kết thúc, trả về max**

Mối quan hệ giữa CTDL và GT (tt)

- **Ví dụ: Cài đặt giải thuật max**

Cách 1:

Dữ liệu vào được lưu trữ bởi 4 biến độc lập: a, b, c, d.

Khi đó giải thuật như sau:

```
max = a;
if (max<b) max = b;
if (max<c) max = c;
if (max<d) max = d;
return max;
```

Cách 2:

Dữ liệu vào được lưu trữ bởi mảng A[4] có kích thước 4 phần tử.

Khi đó giải thuật như sau:

```
max = A[0];
for (i=1;i<4;i++)
    if (max<A[i])
        max = A[i];
return max;
```

1.2. Đánh giá giải thuật

- Đánh giá về bộ nhớ để lưu trữ bộ dữ liệu mà giải thuật sẽ xử lý.
- Đánh giá về giải thuật.
  - Tính khả thi của giải thuật.
  - Thời gian mà giải thuật thực hiện xử lý dữ liệu.

1.2.1. Đánh giá bộ nhớ

- **Có 2 quan niệm:**
  - **Quan niệm 1: Tổng dung lượng nhớ để lưu trữ tất cả các dữ liệu mà giải thuật xử lý (tính bằng đơn vị nhớ - bit, byte, KB…).**
  - **Quan niệm 2: Tổng số chỗ nhớ để lưu tất cả các dữ liệu.**
- **Tổng số chỗ nhớ gồm chỗ nhớ chứa dữ liệu vào, dữ liệu ra, và chỗ nhớ chứa các biến phụ.**

1.2.2. Đánh giá thời gian thực hiện giải thuật

- **Có 2 quan niệm**
  - **Quan niệm 1: Là tổng thời gian mà giải thuật thực hiện xử lý dữ liệu (tính bằng đơn vị thời gian).**
  - **Quan niệm 2: Là tổng số phép toán cơ bản mà giải thuật phải thực hiện để xử lý dữ liệu (các phép toán cơ bản: cộng, trừ, nhân, chia, gán, các phép toán logic, …).**

1.2.3. Độ phức tạp của giải thuật

- **Có 3 kiểu đánh giá**
  - **Độ phức tạp trong trường hợp tốt nhất: Số phép toán ít nhất mà giải thuật phải thực hiện để xử lý một bộ dữ liệu vào.**
  - **Độ phức tạp trung bình: Số phép toán trung bình mà giải thuật phải thực hiện để xử lý một bộ dữ liệu vào.**
  - **Độ phức tạp trong trường hợp xấu nhất: Số phép toán nhiều nhất mà giải thuật phải thực hiện để xử lý một bộ dữ liệu vào.**

1.2.4. Đánh giá giải thuật

- **Ví dụ 1: Bài toán max**
  - **Vào:** Dãy x[] có n số: x[0], x[1], x[2], …, x[n-1], số n.
  - **Ra:** Số max
  - **Giải thuật:**

```
max(x[],n){
   m = x[0];
   for (i=1; i<n; i++)
      if (m < x[i])
         m = x[i];
   return m;
}
```

Đánh giá giải thuật (tt)

- **Bài toán Max: Trường hợp ngẫu nhiên**
  - **Xét dãy `x[] = {34, 32, 45, 65, 23, 54}, n = 6`**
  - **Độ phức tạp bộ nhớ: 9 chỗ nhớ**
    - **Dữ liệu vào: 7 chỗ nhớ (x[0], …, x[5], n)**
    - **Biến phụ: 1 (i)**
    - **Dữ liệu ra: 1 (max)**
  - **Độ phức tạp thời gian: 8 phép toán**

Đánh giá giải thuật (tt)

- Cách đánh giá

```
x[] = {34 32 45 65 23 54}
n = 6
```

```
                                          Số phép toán
max = 34;                                      1
i = 1 -> max < x[1]:false -> không gán         1
i = 2 -> max < x[2]:true  -> max = 45          2
i = 3 -> max < x[3]:true  -> max = 65          2
i = 4 -> max < x[4]:false -> không gán         1
i = 5 -> max < x[5]:false -> không gán         1
```

Đánh giá giải thuật (tt)

- **Bài toán Max: Trường hợp tốt nhất**
  - **Xét dãy x[] = {65, 32, 45, 34, 23, 54}, n = 6**
  - **Độ phức tạp bộ nhớ: 9**
    - **Dữ liệu vào: 7**
    - **Biến phụ: 1**
    - **Dữ liệu ra: 1**
  - **Độ phức tạp thời gian: 6 phép toán**

Đánh giá giải thuật (tt)

- **Bài toán Max: Trường hợp xấu nhất**
  - **Xét dãy x[] = {23 32 34 45 54 65}, n=6**
  - **Độ phức tạp bộ nhớ: 9**
    - **Dữ liệu vào: 7**
    - **Biến phụ: 1**
    - **Dữ liệu ra: 1**
  - **Độ phức tạp thời gian: 11 phép toán**

Đánh giá giải thuật (tt)

- **Bài toán max – Đánh giá tổng quát**
  - **Vào: Dãy x[] có n số: x[0], x[1], …, x[n-1]**
  - **Ra: max**
  - **Giải thuật:**

```
max = x[0];
for (i=1; i<n;i++)
   if (max < x[i])
      max = x[i];
```

Phép toán tích cực

Đánh giá giải thuật (tt)

- **Bài toán Max: Đánh giá tổng quát**
  - **Độ phức tạp bộ nhớ: n+3**
    - **Dữ liệu vào: n+1**
    - **Biến phụ: 1**
    - **Dữ liệu ra: 1**
  - **Độ phức tạp thời gian trong trường hợp xấu nhất: 1+2*(n-1) = 2n-1 phép toán.**

Đánh giá giải thuật (tt)

- **Bài toán Sắp xếp nổi bọt**
  - Vào: Dãy x[] có n số: x[0], x[1], …, x[n-1], số n
  - Ra: Dãy x[] được sắp theo chiều tăng dần
  - Giải thuật:

```
for (i=1; i<n; i++)
    for (j=0; j<n-i; j++)
     if (x[j] > x[j+1]){
       tg = x[j];
       x[j] = x[j+1];
       x[j+1] = tg;
     }
```

Phép toán tích cực

Đánh giá giải thuật (tt)

- Sắp xếp nổi bọt: Đánh giá tổng quát
  - Độ phức tạp bộ nhớ: n+4
    - Dữ liệu vào: n+1
    - Biến phụ: 3 (i, j, tg)
    - Dữ liệu ra: 0
  - Độ phức tạp thời gian trong trường hợp xấu nhất: $4n(n-1)/2 = 2n^2 - 2n$ phép toán.

1.2.5. Ký hiệu O

- Đọc là ô lớn.
- Đây là ký hiệu chỉ độ phức tạp về thời gian của giải thuật.
- Giả sử độ phức tạp trong trường hợp xấu nhất của giải thuật max là 2n-1, khi đó ta gọi độ phức tạp của giải thuật là O(n).
- Độ phức tạp của giải thuật sắp xếp nổi bọt là $O(n^2)$.

1.3. Ngôn ngữ diễn đạt giải thuật

- **Ngôn ngữ tự nhiên:** Liệt kê các bước của giải thuật.
- **Lược đồ khối:** Sử dụng ký hiệu, hình vẽ để biểu diễn giải thuật.
- **Mã giả:** Biểu diễn giải thuật dựa theo cú pháp của ngôn ngữ lập trình (dự định cài đặt) chương trình.

1.4. Các cấu trúc dữ liệu cơ bản

- Là các kiểu dữ liệu có sẵn trong ngôn ngữ lập trình.
- Các kiểu dữ liệu cơ sở:
  - Số nguyên: int, char, long…
  - Số thực: float, double, long double
- Các kiểu dữ liệu có cấu trúc:
  - Mảng 1 chiều, đa chiều, chuỗi ký tự, cấu trúc, tệp tin
- Kiểu dữ liệu con trỏ.

1.4.1. Các kiểu dữ liệu cơ sở

Tên kiểu	Phạm vi	Kích thước	Giải thích
int	-32768 -> +32767	2 byte	Số nguyên có dấu
char	-128 -> +127	1 byte	
long	-2147483648 -> +2147483647	4 byte	
unsigned char	0 -> 255	1 byte	Số nguyên không dấu
unsigned int	0 -> 65535	2 byte	
unsigned long	0 -> 4294967295	4 byte	
float	$1.2*10^{-38}$ -> $3.4*10^{38}$	4 byte	Số dấu chấm động
double	$2.2*10^{-308}$ -> $1.8*10^{308}$	8 byte	
long double	$3.5*10^{-3942}$ - > $3.4*10^{4932}$	10 byte	
bool	true, false	1 bit	logic

1.4.2. Thiết kế cấu trúc dữ liệu và giải thuật

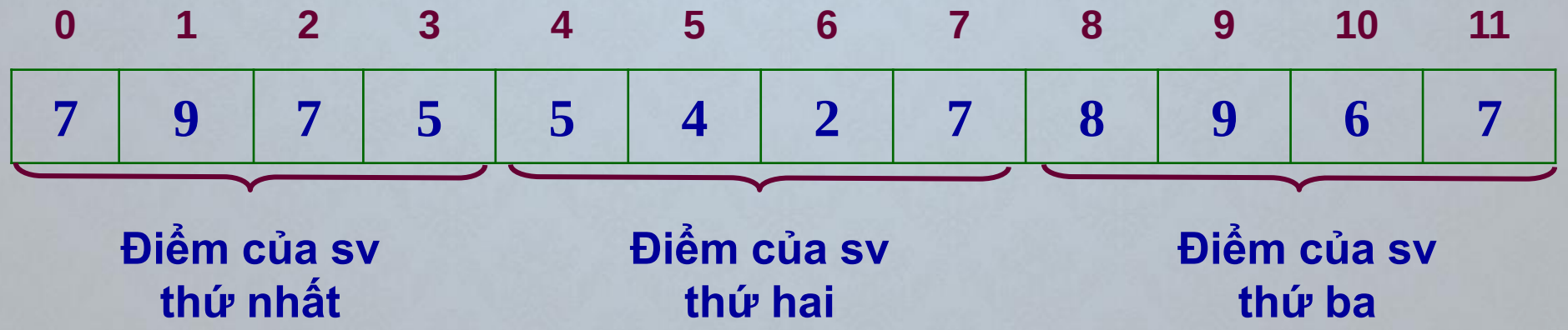
- Dữ liệu về điểm của sinh viên có dạng

Sinh viên	Môn 1	Môn 2	Môn 3	Môn 4
SV1	7	9	7	5
SV2	5	4	2	7
SV3	8	9	6	7

- Xét thao tác xử lý là hiển thị điểm số các môn của từng sinh viên ra màn hình.

Thiết kế cấu trúc dữ liệu và giải thuật (tt)

- Có tất cả 3(sinh viên) * 4(môn) = 12 (điểm số) cần lưu trữ trong bộ nhớ.
- Chọn cấu trúc dữ liệu là mảng một chiều A[12] có 12 phần tử.
- Việc lưu trữ điểm số trong mảng được bố trí như hình dưới đây.



Thiết kế cấu trúc dữ liệu và giải thuật (tt)

- Giải thuật xử lý như sau

```
void xuat(int a[]) {
   int i, mon, so_mon, sv;
      so_mon = 4;
   for (i = 0; i < 12; i++)
   {
      sv = i / so_mon;
      mon = i % so_mon;
      cout<<"\nĐiểm môn: "<<mon;
      cout<<" của sinh viên "<<sv;
      cout<<" là: "<< a[i]<<endl;
   }
}
```

Thiết kế cấu trúc dữ liệu và giải thuật (tt)

- Chọn cấu trúc dữ liệu là mảng 2 chiều 3 dòng, 4 cột A[3][4];

	Cột 1	Cột 2	Cột3	Cột 4
Dòng 1	A[0][0] = 7	A[0][1] = 9	A[0][2] = 7	A[0][3] = 5
Dòng 2	A[1][0] = 5	A[1][1] = 4	A[1][2] = 2	A[1][3] = 7
Dòng 3	A[2][0] = 8	A[2][1] = 9	A[2][2] = 6	A[2][3] = 7

Thiết kế cấu trúc dữ liệu và giải thuật (tt)

- **Giải thuật xử lý như sau**

```
void xuat(int a[4][3]) {
   int i, j, so_sv = 3, so_mon = 4;
   for (i = 0; i < so_sv; i++)
   {
      for (j = 0; j < so_mon; j++)
      {  cout<<"\nĐiểm môn "<<(j + 1);
         cout<<" của sinh viên "<<(i + 1);
         cout<<" là: "<< a[i][j]<<endl;
      }
   }
}
```

Thank you …!