Mục tiêu

- ✓ Giải thuật đệ quy.

Nội dung

Giải thuật đệ quy

Khái niệm đệ quy

Dùng chính khái niệm đang định nghĩa để định nghĩa chính nó. Ví dụ: "Thư mục là nơi chứa các thư mục con và tập tin".

Giải thuật đệ quy

- ✓ Yêu cầu quan trọng là phải có điều kiện biên và điều kiện suy biến. Ví dụ với bài toán tính tổng Sn=1+2+…+n, thì điều kiện biên là n=1, điều kiện suy biến là: $S_n=n+S_{n-1}$. Như vậy khi đi tìm lời giải cho bài toán đệ quy thì chúng ta trước hết phải tìm điều kiện biên (điều kiện chặn). Sau đó tiến hành tìm điều kiện suy biến (truy hồi) tổng quát về điều kiện biên.
- ✓ Cách viết hàm đệ quy đơn giản: định nghĩa tác vụ đệ quy theo ngôn ngữ tự nhiên thế nào thì viết hàm đệ quy tương tự. Ví dụ: tính n!=1*2*3*…*n được định nghĩa như sau:
  - n!=1 nếu n<=1.
  - n!=n*(n-1)!.
  - Viết hàm:

```
int TinhGiaiThua(const int& n)
{
    if(n<=1)
            return 1;
    return n*TinhGiaiThua(n-1);
}
```

Khử đệ quy bằng phương pháp dùng ngăn xếp

Vấn đề

Giải thuật đệ quy ở nhiều tình huống sẽ không hiệu quả khi phải đệ quy quá sâu (quá nhiều lần). Bởi vì bản chất của đệ quy là chương trình phải gọi hàm liên tục (cấp phát và lưu con trỏ chương trình hàm). Đệ quy quá sâu sẽ gây chậm và dễ dẫn đến lỗi over stack vì các con trỏ chương trình hàm được lưu trên vùng nhớ stack (thường rất nhỏ so với chương trình).

Giải pháp khử đệ quy

- ✓ Chưa có phương pháp tổng quát.
- ✓ Sử dụng chính tư tưởng đệ quy để khử đệ quy bằng cách lưu lại các giá trị cho lần tính toán tiếp theo.
- ✓ Sử dụng stack như một hình ảnh chạy đệ quy.
  - Khởi tạo stack với các phần tử thích hợp.
  - Đưa bộ tham số đầu vào stack.
  - Thực hiện trong khi mà stack chưa rỗng thì lấy bộ tham số từ stack ra, xử lý các tác vụ đối với bộ tham số đó, nếu gặp tác vụ đệ quy thì lại đưa bộ tham số cho tác vụ đệ quy vào stack.
- ✓ Code minh họa:

```
//Hàm tính tổng S=1+2+...+n đệ quy
int TinhTong(const int& n)
{
	if(n==1)
		return 1;
	return n + TinhTong(n-1);
}

//Khử đệ quy bằng Stack
int TinhTongKhuDeQuy(const int& n)
{
	Stack s;
	InitStack(s);
	Data d;
	int kq = 0;
	d.x = n;
	Stack_Push(s, d);
	while(!Stack_IsEmpty(s))
	{
		d = Stack_Pop(s);
		kq = kq + d.x;
		d.x = d.x-1;
		if(d.x==0)
			break;
		Stack_Push(s, d);
	}
	return kq;
}

//hàm xuất nhị phân của 1 số nguyên dương
void XuatNhiPhan(const int& n)
{
	if(n>=0)
	{
		if(n/2>0)
			XuatNhiPhan(n/2);
		cout<<(n%2);
	}
}
//khử đệ quy bằng stack
void XuatNhiPhanKhuDeQuy(const int& n)
{
	Stack s;
	InitStack(s);
	Data d, dtemp;
	d.x = n;
	Stack_Push(s, d);
	while(!Stack_IsEmpty(s))
```

```
        {
                while(d.x/2>0)
                {
                        d.x = d.x/2;
                        Stack_Push(s, d);
                }
                dtemp = Stack_Pop(s);
                cout << (dtemp.x % 2);
        }
}
```

Bài tập

Bài 1

Cài đặt tính toán với đệ quy.

1. S(n) = 1 + 2 + 3 + … + n
2. S(n) = 1 + 1/2 + 1/3 + … + 1/n
3. T(n) = n!
4. T(n) = x^n
5. S(n) = x^2 + x^4 + … + x^2n
6. Fibonacci:
    a. f(0) = f(1) = 1
    b. f(n) = f(n-1) + f(n-2), n>1
7. Tính x(n), y(n):
    a. x(0) = 1
    b. y(0) = 0
    c. x(n) = x(n-1) + y(n-1)
    d. y(n) = 3*x(n-1) + 2*y(n-1)
8. C(n, k) = n, nếu k=1
   C(n,k) = 1, nếu n=k
   C(n, k) = C(n, k – 1) + C(n – 1, k – 1), nếu 1 < k < n.

Bài 2

Cho khai báo hàm như sau:

```
bool isPalindrome(int l, int r, char* s)
```

Hãy cài đặt hàm **isPalindrome** để kiểm tra xem chuỗi s có phải là chuỗi đối xứng hay ko.

- Chuỗi đối xứng là chuỗi nếu đọc từ trái sang phải hay từ phải sang trái đều như nhau.
- Chẳng hạn: “123321” là chuỗi đối xứng
  “apqfwfa” ko là chuỗi đối xứng

"quanggnauq" là chuỗi đối xứng

Bài 3

Thực hiện khử đệ quy Bài 2.

Bài 4

Viết các hàm đệ qui sau cho số nguyên dương n:

1. Đếm số lượng chữ số của số n
2. Tính tổng các chữ số của số n
3. Tính tích các chữ số lẻ của số n
4. Kiểm tra số n có phải là số toàn chẵn hay ko
   Chẳng hạn, số 2468 chứa toàn chữ số chẵn 2, 4, 6, 8) nên nó là số toàn chẵn.
   Trong khi số 12468 có chứa 1 chữ số lẻ (1) nên nó ko phải là số toàn chẵn
5. Kiểm tra số n có phải là số toàn lẻ hay ko

Bài 5

Tìm hiểu và cài đặt bài toán "Tháp Hà Nội" bằng phương pháp đệ quy.

Bài 6

Khử đệ quy của bài 5.