Chương 1
THUẬT TOÁN QUICK SORT

ThS. Nguyễn Tấn
Trần Minh Khang

THUẬT TOÁN QUICK SORT

Hãy định nghĩa hàm sắp xếp danh sách liên kết đơn các số thực tăng dần bằng thuật toán Quick Sort

THUẬT TOÁN QUICK SORT

- Thuật toán sắp xếp quick sort chia không gian cần sắp xếp thành 2 không gian con và gọi đệ qui để sắp xếp hai không gian con này.

THUẬT TOÁN QUICK SORT

- Cấu trúc dữ liệu

```
11. struct node
12. {
13.     float info;
14.     struct node *pNext;
15. };
16. typedef struct node NODE;
17. struct list
18. {
19.     NODE *pHead;
20.     NODE *pTail;
21. };
22. typedef struct list LIST;
```

THUẬT TOÁN QUICK SORT

- Hàm init

```
11. void Init(LIST &ℓ)
12. {
13.     ℓ.pHead = NULL;
14.     ℓ.pTail = NULL;
15. }
```

- Hàm thêm vào cuối

```
11. void AddTail(LIST &ℓ,NODE*p)
12. {
13.     if(ℓ.pHead==NULL)
14.         ℓ.pHead = ℓ.pTail = p;
15.     else
16.     {
17.         ℓ.pTail->pNext = p;
18.         ℓ.pTail = p;
19.     }
20. }
```

THUẬT TOÁN QUICK SORT

- Hàm tách node đầu dslk

```
11. NODE* GetHead(LIST &ℓ)
12. {
13.     if (ℓ.pHead==NULL)
14.         return NULL;
15.     if (ℓ.pHead ==ℓ.pTail)
16.     {
17.         NODE*p = ℓ.pHead;
18.         ℓ.pHead = ℓ.pTail = NULL;
19.         return p;
20.     }
21.     NODE*p = ℓ.pHead;
22.     ℓ.pHead = ℓ.pHead->pNext;
23.     p->pNext = NULL;
24.     return p;
25. }
```

- Định nghĩa hàm nối hai dslk đơn

```
LIST AddList(LIST &ℓ1,LIST &ℓ2)
{
    if(ℓ1.pHead==NULL)
    {
        LIST ℓ=ℓ2;
        Init(ℓ2);
        return ℓ;
    }
    if(ℓ2.pHead==NULL)
    {
        LIST ℓ=ℓ1;
        Init(ℓ1);
        return ℓ;
    }
    LIST ℓ;
    ℓ.pHead = ℓ1.pHead;
    ℓ1.pTail->pNext = ℓ2.pHead;
    ℓ.pTail = ℓ2.pTail;
    Init(ℓ1);
    Init(ℓ2);
    return ℓ;
}
```

- Định nghĩa hàm quicksort

```
11. void QuickSort(LIST &ℓ)
12. {
13.     if(ℓ.pHead==NULL)
14.         return;
15.     if(ℓ.pHead==ℓ.pTail)
16.         return;
17.     NODE*tt = GetHead(ℓ);
18.     LIST ℓ1, ℓ2;
19.     Init(ℓ1);
20.     Init(ℓ2);
21.     while(ℓ.pHead!=NULL)
22.     {
23.         NODE*p = GetHead(ℓ);
24.         if(p->info<=tt->info)
25.             AddTail(ℓ1,p);
26.         else
27.             AddTail(ℓ2,p);
28.     }
29.     QuickSort(ℓ1);
30.     QuickSort(ℓ2);
31.     AddTail(ℓ1,tt);
32.     ℓ = AddList(ℓ1, ℓ2);
33. }
```