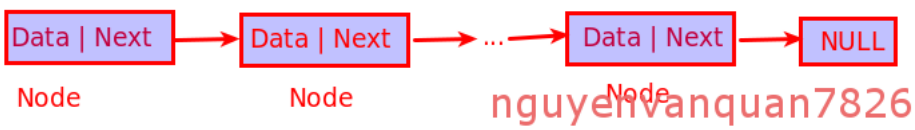
Lập trình C: Bài 13 – Danh sách liên kết đơn cài bằng con trỏ

Nội dung

Danh sách liên kết có thể được cài đặt bằng mảng hoặc bằng con trỏ. Trong bài viết này mình sẽ hướng dẫn các bạn sử dụng mảng :), Loại danh sách này gọi tắt là danh sách liên kết đơn

Trong các bài trước mình viết code tất cả đều là chuẩn C, nhưng từ bây giờ mình sẽ xen lẫn chút cấu trúc của C++ nên code trong bài này bạn để trong file *.cpp nhé.

Danh sách liên kết có thể được mô tả như sau:



Danh sách liên kết đơn

Trong đó Data là dữ liệu của mỗi Node, có thể là Sinh viên, công nhân,… (có kiểu item, và mình làm đơn giản với số nguyên), Next là con trỏ trỏ tới phần tử tiếp theo.

1. Cài đặt danh sách ▲

```
typedef int item; //kieu cac phan tu dinh nghia la item
```

```
typedef struct Node //Xay dung mot Node trong danh sach
{
   item Data; //Du lieu co kieu item
   Node *next; //Truong next la con tro, tro den 1 Node tiep t
heo
};
typedef Node *List; //List la mot danh sach cac Node
```

2 Khởi tạo danh sách rỗng ▲

```
void Init (List &L) // &L lay dia chi cua danh sach ngay khi truy
en vao ham
{
   L=NULL; //Cho L tro den NULL
}
```

Trong các bài trước để có thể thay đổi được giá trị của đối mà ta truyền vào hàm ta thường dùng biến con trỏ (*) và trong lời gọi hàm ta cần có & trước biến tuy nhiên khi chúng ta sử dụng cách truyền địa chỉ ngay khi khởi tạo hàm thì trong lời gọi hàm ta tiên hành truyền biến bình thường mà không phải lấy địa chỉ (thêm &) trước biến nữa.
(Đây là một cách truyền địa chỉ cho biến trong hàm ở C++.)

3 Kiểm tra danh sách rỗng hay không ▲

Cái này khỏi giải thích nhiều:

```
int Isempty (List L)
{
   return (L==NULL);
}
```

4 Tính độ dài danh sách ▲

Ta dùng 1 Node để duyệt từ đầu đến cuối, vừa duyệt vừa đếm

```
int len (List L)
{
   Node *P=L; //tao 1 Node P de duyet danh sach L
   int i=0; //bien dem
   while (P!=NULL) //trong khi P chua tro den NULL (cuoi danh
 sach thi lam)
   {
      i++; //tang bien dem
      P=P->next; //cho P tro den Node tiep theo
   }
   return i; //tra lai so Node cua l
}
```

môn học an toàn bảo mật thông tin.

Bài viết mới

- [WordPress] Thay thế Older posts bằng số trang
- [Java] Sử dụng interface
- [wordpress] Tạo Child Theme
- [WordPress] Show edit link in post
- Cài wordpress trên localhost trong ubuntu

Nhận tin qua Email

Email Addres Đăng ký

Chuyên mục

Chọn chuyên mục

Bài được xem nhiều

- [School_PPNCKH] Hệ thống các phương pháp nghiên cứu khoa học
- [Thuật toán] Tìm đường đi ngắn nhất Dijkstra, Floyd
- [Linux - C/C++] Lập trình C/C++ trên Ubuntu (Linux)
- [Pascal - TUT] Bài 10: Tệp trong pascal - Kiểu file
- [Thuật toán - C/C++]

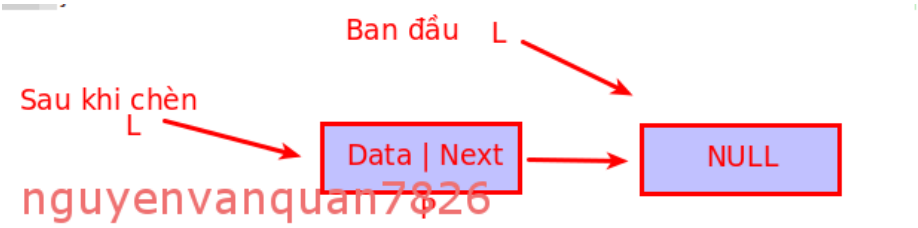
5 Tạo 1 Node trong danh sách

Việc tạo 1 Node chứa thông tin trong danh sách sẽ giúp ta dễ dàng chèn, xóa và quản lý danh sách hơn. Trước tiên ta sẽ phải cấp phát vùng nhớ cho Node và sau đó gán Data vào là ok

```
Node *Make_Node (Node *P, item x) //tao 1 Node P chua tho
ng tin la x
{
   P = (Node *) malloc (sizeof (Node)); //Cap phat vung nho ch
o P
   P->next = NULL; //Cho truong Next tro den NULL
   P->Data = x; //Ghi du lieu vao Data
   return P;
}
```

6 Chèn Node P vào vị trí đầu tiên

Để chèn P vào đầu danh sách trước tiên ta cho P trỏ đến L, sau đó chỉ việc cho L trỏ lại về P là ok



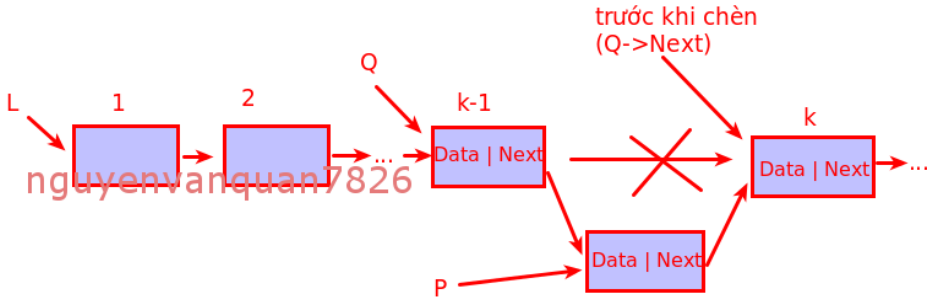
Chèn vào đầu danh sách liên két

```
void Insert_first (List &L, item x)  //Chen x vao vi tri dau tien tr
ong danh sach
{
   Node *P;
   P = Make_Node(P,x); //tao 1 Node P
   P->next = L; //Cho P tro den L
   L = P; //L tro ve P
}
```

7. Chèn Node P vào vị trí k trong danh sách

Trước tiên ta kiểm tra vị trí chèn có hợp lệ không, nếu hợp lệ kiểm tra tiếp chèn vào vị trí 1 hay k >1 . Với k >1 ta thực hiện duyệt bằng Node Q đến vị trí k-1 sau đó cho P->Next trỏ đến Node Q->Next, tiếp đến cho Q->Next trỏ đến P

Quick Sort - Các vấn đề liên quan



Chèn phần tử vào vị trí k trong danh sách liên kết

```
void Insert_k (List &L, item x, int k) //chen x vao vi tri k trong d
anh sach
{
   Node *P, *Q = L;
   int i=1;
   if (k<1 || k> len(L)+1) printf("Vi tri chen khong hop le !"); //ki
em tra dieu kien
   else
   {
      P = Make_Node(P,x); //tao 1 Node P
      if (k == 1) Insert_first(L,x); //chen vao vi tri dau tien
      else //chen vao k != 1
      {
         while (Q != NULL && i != k-1) //duyet den vi tri k-1
         {
            i++;
            Q = Q->next;
         }
         P->next = Q->next;
         Q->next = P;
      }
   }
}
```

8 Tìm phần tử co giá trị x trong danh sách

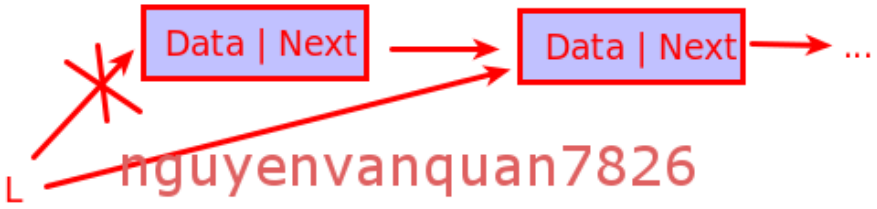
Ta duyệt danh sách cho đến khi tìm thấy hoặc kết thúc và trả về vị trí nếu tìm thấy, ngược lại trả về 0

```
int Search (List L, item x) //tim x trong danh sach
{
   Node *P=L;
   int i=1;
   while (P != NULL && P->Data != x) //duyet danh sach den kh
i tim thay hoac ket thuc danh sach
   {
      P = P->next;
      i++;
   }
```

```
    if (P != NULL) return i; //tra ve vi tri tim thay
    else return 0; //khong tim thay
}
```

9 Xóa phần tử ở vị trí đầu tiên

Trước tiên ta lưu giá trị của phần tử đầu tiên vào biến x, sau đó tiền hành cho L trỏ đến L->Next

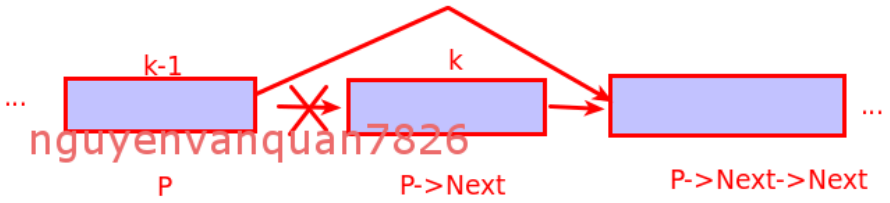


Xóa phần tử đầu tiên trong danh sách

```
void Del_frist (List &L, item &x) //Xoa phan tu dau tien
{
    x = L->Data; //lay gia tri ra neu can dung
    L = L->next; //Cho L tro den Node thu 2 trong danh sach
}
```

10. Xóa phần tư ở vị trí k

Dùng P duyệt đến vị trí k-1 và tiến hành cho P->Next trỏ đến phần tư kế tiếp k mà bỏ qua k. Lưu ý trong hình mình quên không lưu lại giá trị cần xóa tuy nhiên các bạn cần lưu lại như khi xóa ở vị trí đầu tiên.



Xóa phần tử thứ k trong danh sách liên kết

```
void Del_k (List &L, item &x, int k) //Xoa Node k trong danh sa
ch
{
    Node *P=L;
    int i=1;
    if (k<1 || k>len(L)) printf("Vi tri xoa khong hop le !"); //kiem
tra dieu kien
    else
    {
        if (k==1) Del_frist(L,x); //xoa vi tri dau tien
        else //xoa vi tri k != 1
        {
            while (P != NULL && i != k-1) //duyet den vi tri k-1
            {
                P=P->next;
```

```
            i++;
         }
         P->next = P->next->next; //cho P tro sang Node ke tiep
vi tri k
      }
   }
}
```

11. Xóa phần tử có giá trị x

Đon giản là ta tìm x trong danh sách bằng hàm Search và xóa tại vị trí tìm thấy mà ta nhận được

```
void Del_x (List &L, item x) //xoa phan tu x trong danh sach
{
   while (Search(L,x)) Del_k (L,x,Search(L,x)); //trong khi van ti
m thay x thi van xoa
}
```

12. Chương trình hoàn chỉnh (full)

```
#include<stdio.h>
#include<stdlib.h>
typedef int item; //kieu cac phan tu dinh nghia la item
typedef struct Node //Xay dung mot Node trong danh sach
{
   item Data; //Du lieu co kieu item
   Node *next; //Truong next la con tro, tro den 1 Node tiep t
heo
};
typedef Node *List; //List la mot danh sach cac Node

void Init (List &L); //khoi tao danh sach rong
int len (List L); // Do dai danh sach
Node *Make_Node (Node *P, item x); //Tao 1 Node P voi thon
g tin chu trong no
void Insert_first (List &L, item x); //Chen phan tu vao dau dan
h sach
void Insert_k (List &L, item x, int k); //Chen phan tu vao vi tri k
trong danh sach
void Input (List &L);//Nhap danh sach
void Output (List L);//Xuat danh sach
int Search (List L, item x); //Tim phan tu x trong danh sach, ha
m tre ve vi tri cua phan tu tim duoc
void Del_frist (List &L, item &x); //Xoa phan tu dau danh sach
void Del_k (List &L, item &x, int k); //Xoa phan tu vi tri k trong
danh sach
void Del_x (List &L, item x);//Xoa phan tu co gia tri x trong dan
h sach
```

```
void Init (List &L) // &L lay dia chi cua danh sach ngay khi truy
en vao ham
{
   L=NULL; //Cho L tro den NULL
}

int Isempty (List L)
{
   return (L==NULL);
}

int len (List L)
{
   Node *P=L; //tao 1 Node P de duyet danh sach L
   int i=0; //bien dem
   while (P!=NULL) //trong khi P chua tro den NULL (cuoi danh
 sach thi lam)
   {
      i++; //tang bien dem
      P=P->next; //cho P tro den Node tiep theo
   }
   return i; //tra lai so Node cua l
}

Node *Make_Node (Node *P, item x) //tao 1 Node P chua tho
ng tin la x
{
   P = (Node *) malloc (sizeof (Node)); //Cap phat vung nho ch
o P
   P->next = NULL; //Cho truong Next tro den NULL
   P->Data = x; //Ghi du lieu vao Data
   return P;
}

void Insert_first (List &L, item x)  //Chen x vao vi tri dau tien tr
ong danh sach
{
   Node *P;
   P = Make_Node(P,x); //tao 1 Node P
   P->next = L; //Cho P tro den L
   L = P; //L tro ve P
}

void Insert_k (List &L, item x, int k) //chen x vao vi tri k trong d
anh sach
{
   Node *P, *Q = L;
   int i=1;
   if (k<1 || k> len(L)+1) printf("Vi tri chen khong hop le !"); //ki
```

```
em tra dieu kien
   else
   {
      P = Make_Node(P,x); //tao 1 Node P
      if (k == 1) Insert_first(L,x); //chen vao vi tri dau tien
      else //chen vao k != 1
      {
         while (Q != NULL && i != k-1) //duyet den vi tri k-1
         {
            i++;
            Q = Q->next;
         }
         P->next = Q->next;
         Q->next = P;
      }
   }
}

int Search (List L, item x) //tim x trong danh sach
{
   Node *P=L;
   int i=1;
   while (P != NULL && P->Data != x) //duyet danh sach den kh
i tim thay hoac ket thuc danh sach
   {
      P = P->next;
      i++;
   }
   if (P != NULL) return i; //tra ve vi tri tim thay
   else return 0; //khong tim thay
}

void Del_frist (List &L, item &x) //Xoa phan tu dau tien
{
   x = L->Data; //lay gia tri ra neu can dung
   L = L->next; //Cho L tro den Node thu 2 trong danh sach
}

void Del_k (List &L, item &x, int k) //Xoa Node k trong danh sa
ch
{
   Node *P=L;
   int i=1;
   if (k<1 || k>len(L)) printf("Vi tri xoa khong hop le !"); //kiem
tra dieu kien
   else
   {
      if (k==1) Del_frist(L,x); //xoa vi tri dau tien
      else //xoa vi tri k != 1
      {
```

```
        while (P != NULL && i != k-1) //duyet den vi tri k-1
        {
           P=P->next;
           i++;
        }
        P->next = P->next->next; //cho P tro sang Node ke tiep
vi tri k
      }
   }
}

void Del_x (List &L, item x) //xoa phan tu x trong danh sach
{
   while (Search(L,x)) Del_k (L,x,Search(L,x)); //trong khi van ti
m thay x thi van xoa
}

void Input (List &L) //nhap danh sach
{
   int i=0;
   item x;
   do
   {
      i++;
      printf ("Nhap phan tu thu %d : ",i);
      scanf("%d",&x);
      if (x != 0) Insert_k(L,x,len(L)+1);
   } while(x != 0); //nhap 0 de ket thuc
}

void Output (List L) //xuat danh sach
{
   Node *P=L;
   while (P != NULL)
   {
      printf("%5d",P->Data);
      P = P->next;
   }
   printf("\n");
}

int main()
{
   List L;
   Init(L);
   Input(L);
   Output(L);

   int lua_chon;
   printf("Moi ban chon phep toan voi DS LKD:");
```

```
    printf("\n1: Kiem tra DS rong");
    printf("\n2: Do dai DS");
    printf("\n3: Chen phan tu x vao vi tri k trong DS");
    printf("\n4: Tim mot phan tu trong DS");
    printf("\n5: Xoa phan tu tai vi tri k");
    printf("\n6: XOa phan tu x trong DS");
    printf("\n7: Thoat");
    do
    {
       printf("\nBan chon: ");
       scanf("%d",&lua_chon);
    switch (lua_chon)
    {
       case 1:
       {
          if (Isempty(L)) printf("DS rong !");
          else printf ("DS khong rong !");
          break;
       }
       case 2: printf ("Do dai DS la: %d.",len(L));break;
       case 3:
       {
          item x;
          int k;
          printf ("Nhap phan tu can chen vao DS: ");
          scanf("%d",&x);
          printf ("Nhap vi tri can chen: ");
          scanf ("%d",&k);
          Insert_k (L,x,k);
          printf ("DS sau khi chen:\n");
          Output(L);
          break;
       }
       case 4:
       {
          item x;
          printf ("Moi ban nhap vao phan tu can tim: ");
          scanf("%d",&x);
          int k=Search(L,x);
          if (k) printf ("Tim thay %d trong DS tai vi tri thu: %d",x,k)
;
          else printf ("Khong tim thay %d trong danh sach !",x);
          break;
       }
       case 5:
       {
          int k;
          item x;
          printf ("Nhap vi tri can xoa: ");
          scanf ("%d",&k);
```

```
            Del_k (L,x,k);
            printf ("DS sau khi xoa:\n");
            Output(L);
            break;
        }
        case 6:
        {
            item x;
            printf ("Nhap phan tu can xoa: ");
            scanf ("%d",&x);
            Del_x (L,x);
            printf ("DS sau khi xoa:\n");
            Output(L);
            break;
        }
        case 7: break;
    }
    }while (lua_chon !=7);
    return 0;
}
```

Bài viết được thực hiện trong loạt bài hướng dẫn lập trình c cơ bản

Tháng Mười Hai 21, 2014 nguyenvanquan7826 LT C - C++ Leave a response danh sách liên kết

Additional Reading...

- Lập trình C: Bài 16 – Xây dựng ngăn xếp (Queue)
- Lập trình C: Bài 15 – Cài đặt ngăn xếp (Stack)
- Lập trình C: Bài 14 – Danh sách liên kết kép
- Lập trình C: Bài 12 – Danh sách liên kết cài bằng mảng
- fflush(stdin) trong ubuntu (linux)

Trả lời

Thư điện tử của bạn sẽ không được hiện thị công khai. Các trường bắt buộc được đánh dấu *

Tên *