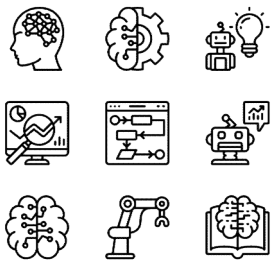
Computer Science for Practicing Engineers

Danh sách liên kết (Linked List)



TS. Huỳnh Bá Diệu
Email: dieuhb@gmail.com
Phone: 0914146868

Danh sách liên kết

Nội dung

1. Kiểu danh sách liên kết
2. Các loại danh sách liên kết
3. Các thao tác trên kiểu dữ liệu danh sách
4. Sử dụng kiểu danh sách trên các ngôn ngữ lập trình

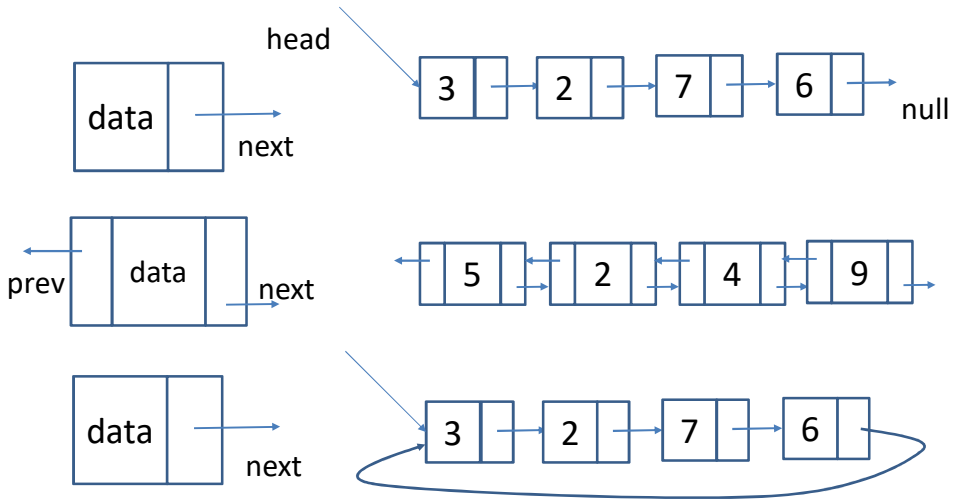
Các cấu trúc dữ liệu trừu tượng ADT (Abstract Data Type)

- Danh sách liên kết (Đơn, Đôi, Vòng)
- Ngăn xếp (Stack)
- Hàng đợi (Queue)
- Cây (Tree)
- Bảng băm (Hash Table)

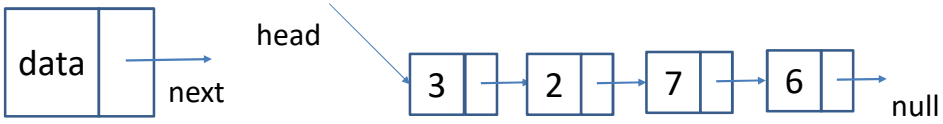
Mỗi ngôn ngữ lập trình đều có các kiểu dữ liệu nguyên thuỷ (cơ sở). Từ kiểu dữ liệu nguyên thuỷ này ta có thể định nghĩa các kiểu dữ liệu mới.

Các kiểu dữ liệu nguyên thuỷ thường là: ký tự, số, chuỗi, mảng..

Các loại danh sách liên kết (Linked List)



Khai báo danh sách liên kết (Linked List)

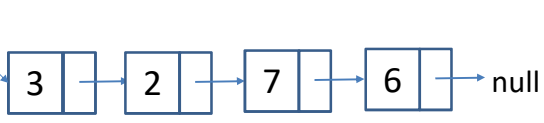



```
class Node
{
        int data;
        Node next;
        …….

}
```

```
typedef struct  Node
{
        int data;
        Node *next;

};
```

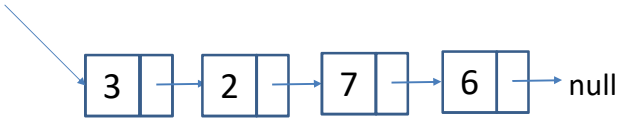
Các thao tác trên danh sách liên kết đơn

- Duyệt danh sách
- Thêm phần tử vào danh sách
- Xoá phần tử từ danh sách
- Sắp xếp danh sách
- Đảo danh sách



Các thao tác trên danh sách liên kết đơn

- Duyệt danh sách
- Thêm phần tử vào danh sách
- Xoá phần tử từ danh sách
- Sắp xếp danh sách
- Đảo danh sách



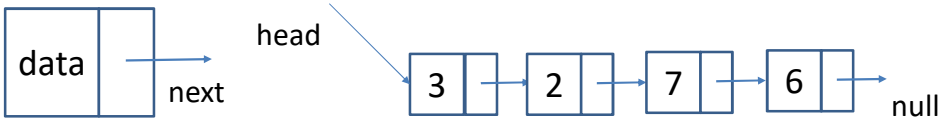
Cho biết các vị trí có thể thêm và xoá trong danh sách???

Các thao tác trên danh sách liên kết đơn

- Thêm phần tử vào đầu danh sách
- Thêm phần tử vào cuối danh sách
- Thêm vào vị trí k

Tương tự cho trường hợp xoá

Khai báo một phần tử trong Danh sách liên kết



struct Node { int data; Node *next;};

Danh sách liên kết đơn (Linked List)

```
class SList {
        struct Node { int data; Node *next;};
        Node *head;
        public:
                SList() { head=NULL;}
                void add(int x) { }
                void print()    { }
};
```

In danh sách liên kết (Linked List)

```
int main()
{
        SList L;
        L.add(1); L.add(3);L.add(5);L.add(7);
        L.print();
        ----------
}
```

In danh sách liên kết (Linked List)

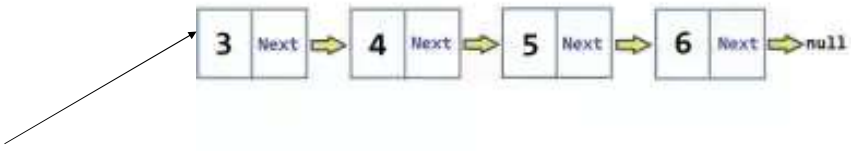
```
void add(int x){
        Node *t = new Node;   t->data=x;  t->next=head;
        head=t;
}
void print() {
        cout<<"\n Noi dung danh sach:\n ===>";  Node *p= head;
        while(p!=NULL) { cout<<p->data<<" --> "; p=p->next; }
        cout<<" NULL\n";
}
```

Nhập danh sách liên kết (Linked List)

```
void nhap(){
        head= NULL;  Node *cuoi= NULL;  int x;
        while(1)  {
                cout<<"\n nhap x (<>0) them vao danh sach:"; cin>>x;
                if(x==0) break;
                Node *t= new Node; t->data=x; t->next=NULL;
                if(head==NULL) head=cuoi=t;  else { cuoi->next=t; cuoi=t; }
        }
}
```

Tính tổng các phần tử trong danh sách liên kết

Cách làm như thế nào???



Tính tổng các phần tử trong danh sách liên kết

Tính tổng các nốt trong danh sách liên kết

Thuật toán:

- Cho p trỏ đầu danh sách, khởi gán k=0
- Trong khi p khác rỗng
  + k = k + p->data;
  + p=p->next
- Trả về giá trị của k

Danh sách liên kết (Linked List)

```
int tong()
{
      int k=0;
      Node *p= head;
      while(p!= NULL)
         { k+= p->data; p= p->next;}
      return k;
}
```

int tongdq(Node *L)

```
{
    if(L==NULL) return 0;
    else return L->data + tongdq(L->next);
}
```

int tongdq()

```
{
        return tongdq(head);
}
```

Danh sách liên kết (Linked List)

Thực hiện theo phương pháp lặp và đệ qui (10 phút)

Đếm số phần tử lẻ trong danh sách?

Kiểm tra trong danh sách có chứa giá trị x hay không?

Kiểm tra danh sách có tăng dần hay không?

Đếm số phần tử có giá trị là bội của 3 trong danh sách?

Tìm giá trị lớn nhất trong danh sách liên kết?

Chèn giá trị x vào cuối danh sách liên kết

Các trường hợp có thể có???

Chèn giá trị x vào cuối danh sách liên kết

Thuật toán

Nếu danh sách rỗng thì tạo nốt mới và gán head = nốt vừa tạo

Ngược lại

+ Cho p trỏ đến nốt cuối của danh sách
+ Tạo 1 nốt mới t có dữ liệu là x và liên kết là NULL
+ Nối nốt mới sau nốt cuối (p->next =t)

Chèn giá trị x vào cuối danh sách liên kết

```
void chenc(int x)    {
    if (head ==NULL) { head=  new Node; head->data=x; head->next=NULL;}
    else  {
       // den not cuoi va tao mot not noi sau not cuoi
       Node *p= head; while(p->next!= NULL) p=p->next;
       Node *t= new Node; t->data=x; t->next=NULL; p->next=t;
    }
}
```

Chèn giá trị x vào vị trí k trong danh sách liên kết

Các trường hợp có thể có?

Chèn giá trị x vào vị trí k trong danh sách liên kết

Thuật toán

Nếu k<1 thì báo không chèn được
Ngược lại
    nếu k=1 thì chèn đầu danh sách
    ngược lại
    {
        Cho p trỏ đến nốt k-1 của danh sách
        Nếu p bằng rỗng thì báo vị trí chèn không hợp lệ
        Ngược lại tạo nốt mới t và gắn t nằm giữa p và p->next
    }

Chèn giá trị x vào vị trí k trong danh sách liên kết

```
void chenk(int x, int k)   {
      if(k<1) cout<<"\n Vi tri chen khong hop le!";
      else
         if (k==1) { Node *t = new Node; t->data=x; t->next=head; head =t; }
         else
         {
               // bo sung vao day
         }
   }
```

Chèn giá trị x vào vị trí k trong danh sách liên kết

```
void chenk(int x, int k) {
    if(k<1) cout<<"\n Vi tri chen khong hop le!";
    else  if (k==1) { Node *t= new Node; t->data=x; t->next= head; head= t;}
    else {
        int vt=1; Node *p =head;   while(vt<k-1 && p!=NULL) { vt++; p=p->next;}
        if(p==NULL) cout<<"\n Vi tri chen khong hop le!!!";
        else {
              Node *t= new Node; t->data=x; t->next= p->next; p->next=t;
              cout<<"\n da chen xong not tai vi tri "<<k<<"\n";
            }
        }
}
```

Xoá nốt trong danh sách liên kết (Linked List)

Xoá nốt trong danh sách liên kết

Xoá nốt đầu, xoá nốt cuối, xoá nốt thứ k

Khi xoá một phần tử thuộc danh sách thì yêu cầu trước tiên là danh sách không phải là danh sách rỗng.

```
void xoad()
{
        if(head==NULL) cout<<"\n Danh sach rong, khong xoa duoc!";
        else { Node *q=head; head=head->next; delete q; }
}
```

Xoá nốt thứ k trong danh sách liên kết

Các trường hợp có thể có???

Xoá nốt thứ k trong danh sách liên kết

Thuật toán

Nếu k<1 || danh sách rỗng thì báo không xoá được

Ngược lại :

Nếu k=1 thì xoá nốt đầu tiên

ngược lại:

{

Cho p trỏ đến nốt thứ k-1;

Nếu p rỗng hoặc p->next = rỗng thì báo không xoá được

ngược lại cho p->next= p->next->next (bỏ qua nốt ở vị trí p->next)

}

Xoá nốt thứ k trong danh sách liên kết

```
void xoak(int k) {
        if(k<1|| head==NULL) cout<<"\n Vi tri xoa khong hop le!";
        else   if(k==1) {Node *q=head; head=head->next; delete q;}
        else {
          // cho p tro den vi tri k-1
          int vt=1;  Node *p= head;    while(p!= NULL&& vt<k-1) { vt++ ; p=p->next;}
          // kiem tra co not o vi tri k khong
          if(p==NULL||p->next==NULL) cout<<"\n Vi tri xoa khong hop le!";
          else { Node *q=p->next; p->next= q->next; delete q;}
        }
}
```

Chèn nốt vào vị trí k trong danh sách liên kết (ĐQ)

Thuật toán: **Node *Chenk_dq(Node *&L, int x, int k)**

- Nếu k==1 thì chèn đầu danh sách

Ngược lại

Nếu k>1 và L == NULL thì báo ko chèn được

Ngược lại: chèn vào vị trí thứ k-1 của danh sách kế tiếp

// L->next = Chenk_dq(L->next, x, k-1)

- **return L;**

void chenkdq (int x, int k) { head= Chenk_dq(head, x, k); }

Chèn nốt vào vị trí k trong danh sách liên kết (ĐQ)

```
Node *chenk_dq(Node *L, int x, int k)  {
   if(k<1||(L==NULL&& k>1))
      {cout<<"\n vi tri chen ko hop le!"; return L;}
   else
      if(k==1) { Node *t= new Node; t->data=x; t->next=L; return t;}
      else {L->next= chenk_dq(L->next,x, k-1); return L;}
}
void chenk_dq(int x, int k)   { head= chenk_dq(head, x, k);          }
```

Xóa nốt ở vị trí k trong danh sách liên kết (ĐQ)

Thuật toán: Xoak_dq(Node *L, int k)

- Nếu k <1 hoặc danh sách rỗng thì báo không xóa được

Ngược lại

Nếu k=1 thì xóa nốt trỏ bởi L (L=L->next)

Ngược lại : xóa nốt thứ k-1 của danh sách kế tiếp sau L

// L->next = Xoak_dq(L->next, k-1)

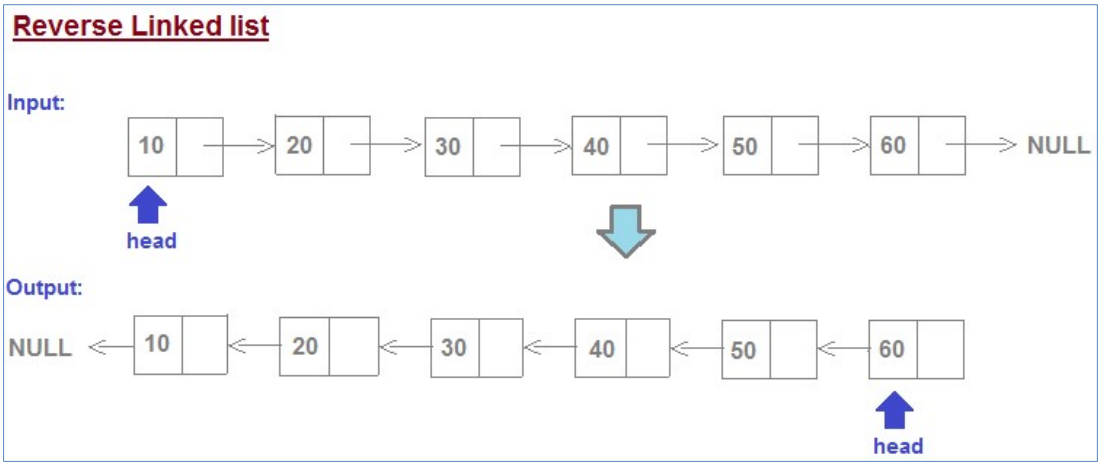
- return L;

Xóa nốt ở vị trí k trong danh sách liên kết (ĐQ)

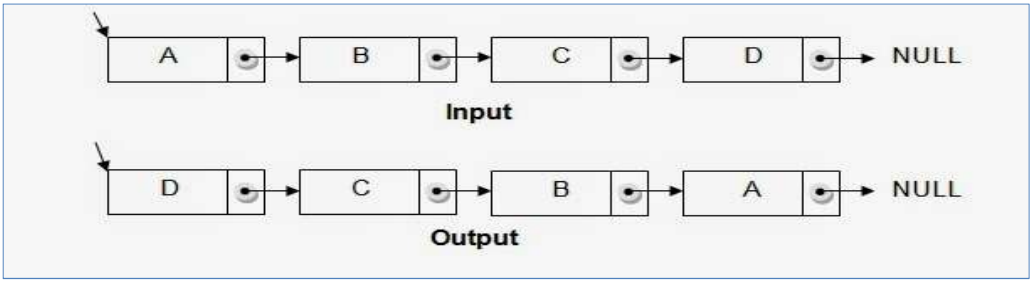
Thuật toán

```
Node* Xoak_dq(Node *&L, int k )
{
        if(L==NULL||k<1) cout<<" Ko xoa duoc";
        else   if(k==1) { Node *q= L; L= L->next; delete q;}
                else L->next = Xoak_dq( L->next, k-1 );
        return L;
}
```

Đảo danh sách liên kết



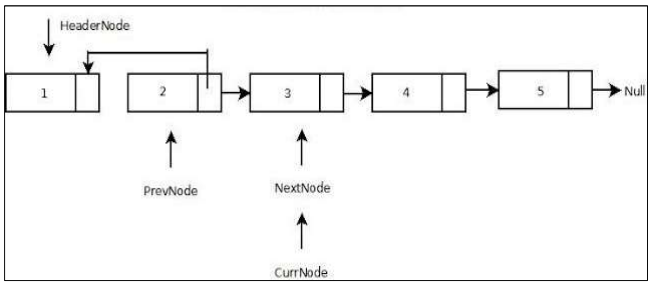
Đảo danh sách liên kết



Đảo danh sách liên kết

Ví dụ danh sách có 5 phần tử

- Đảo từ 1 đến 1,
- Đảo từ 2 đến 1
- Đảo từ 3 đến 1
- Đảo từ 4 đến 1
- Đảo từ 5 đến 1



Khi đảo tử 5 đến 1 ta đảo từ 1 đến 4, sau đó cho nốt 5 chỉ đến phần đảo từ 1 đến 4 và cập nhật đầu danh sách là nốt thứ 5

Tương tự khi đảo từ 4 đến 1, ta thực hiện đảo từ 1 đến 3 sau đó cho nốt kế tiếp nốt 4 là phần đảo của danh sách từ 1 đến 3

Đảo danh sách liên kết

Đảo danh sách liên kết

Thuật toán

Khi đảo tử 5 đến 1 ta đảo từ 1 đến 4, sau đó cho nốt 5 chỉ đến phần đảo từ 1 đến 4 và cập nhật đầu danh sách là nốt thứ 5

Tương tự khi đảo từ 4 đến 1, ta thực hiện đảo từ 1 đến 3 sau đó cho nốt kế tiếp nốt 4 là phần đảo của danh sách từ 1 đến 3

Như vậy ta cần:

- **1 con trỏ p lưu trữ nốt đang xét**
- **1 con trỏ q lưu trữ nốt phía sau nốt đang xét**
- **1 con trỏ lưu trữ t phần đảo phía trước nốt đang xét (ban đầu t= rỗng)**

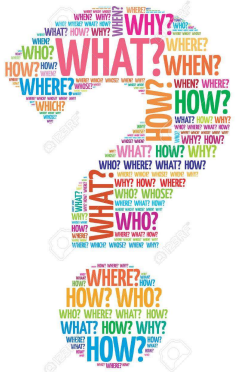
Đảo danh sách liên kết

Thuật toán:

- Node t= rỗng
- Node p= head;
- Trong khi p <> rỗng
  - + Node q= p->next; // lưu trữ nốt phía sau
  - + p->next = t; cho nốt kế tiếp của p là phần đảo phía trước
  - + t= p; // cập nhật phần đảo từ nốt đầu tiên đến nốt p
  - + p= q; // cho p trỏ đến nốt kế tiếp trong danh sách để đảo
- head= t; // cập nhật lại đầu danh sách (là phần đảo)

Đảo danh sách liên kết

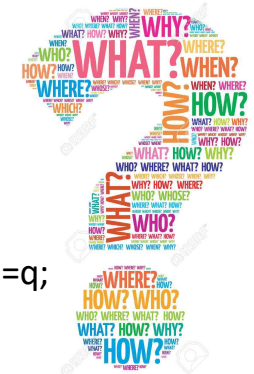
CODE hoàn chỉnh như thế nào???



Đảo danh sách liên kết

CODE hoàn chỉnh như thế nào???

```
void dao()    {
     Node *p=head,*t=NULL;
     while (p!=NULL)
          {
                Node *q= p->next;  p->next=t; t= p; p=q;
          }
    head=t;
}
```

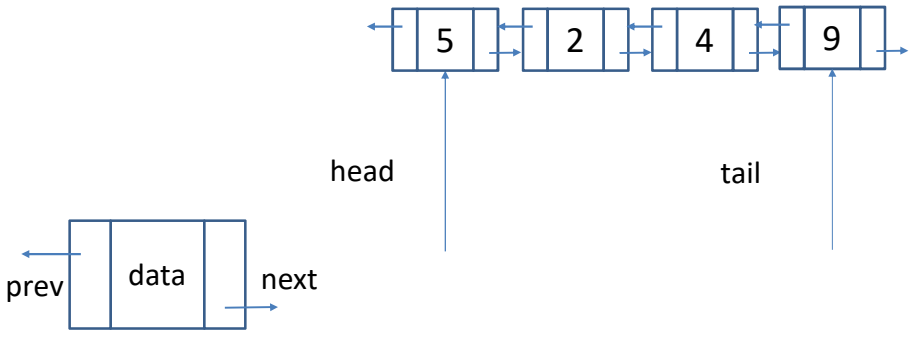


Bài tập về nhà

1. Hoán vị nốt thứ i và j trong danh sách (chỉ thay đổi liên kết)
2. Xóa các số lẻ trong danh sách
3. Sắp xếp danh sách tăng dần
4. Tách danh sách thành hai danh sách, 1 danh sách chứa các số lẻ, 1 danh sách chứa số chẵn
5. Kiểm tra danh sách chứa quá 3 số lẻ không

Bài làm trên vở, Không THÌ MÀ LÀ NHƯNG DO!!!

Danh sách liên kết đôi (Double Linked List)



struct DNode { int data; DNode *next, *prev;};

Danh sách liên kết đôi

```
class DList {
        struct DNode { int data; DNode *next, *prev;};
        DNode *head, *tail;
        public:
                DList() { head=NULL; tail=NULL;}
                void add(int x)   {       }
                void print() {      }
}
int main() { DList LL; LL.add(1); LL.add(3); LL.add(5); LL.print();}
```

Danh sách liên kết đôi

```
void add(int x)
{
        DNode *t = new DNode;
        t->data=x; t->next=head; t->prev= NULL;
        if(head!=NULL) head->prev=t;
        head=t;
}
```

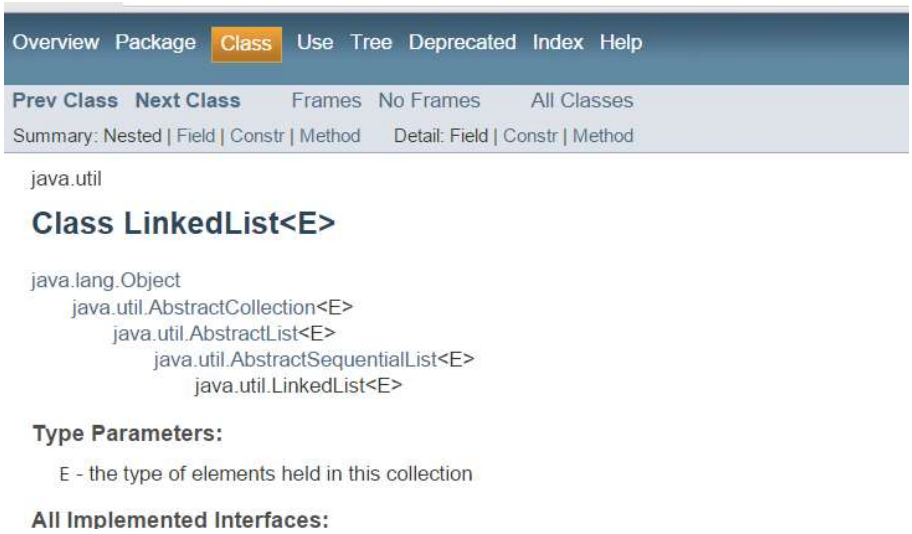
Danh sách liên kết đôi

```
void print(){
        cout<<"\n Noi dung danh sach:\n ===>";
        DNode *p= head;
        while(p!=NULL) {
                cout<<p->data;
                if (p->next!=NULL ) cout<<" <--> "; else cout<<" -->";
                p=p->next;
        }
        cout<<" NULL\n";
}
```

Danh sách liên kết đôi

```
void nhap() {
    head= NULL; tail=NULL;   int x;
    while(1) {
        cout<<"\n nhap x (<>0) them vao danh sach:"; cin>>x;
        if(x==0) break;
        DNode *t= new DNode; t->data=x; t->next=NULL; t->prev=NULL;
        if(head==NULL) head=tail=t;
        else { tail->next=t; t->prev= tail; tail=t; }
    }
}
```

Sử dụng Kiểu danh sách liên kết trong java



Overview Package Class Use Tree Deprecated Index Help

Prev Class Next Class Frames No Frames All Classes

Summary: Nested | Field | Constr | Method Detail: Field | Constr | Method

java.util

Class LinkedList<E>

java.lang.Object
  java.util.AbstractCollection<E>
    java.util.AbstractList<E>
      java.util.AbstractSequentialList<E>
        java.util.LinkedList<E>

Type Parameters:

E - the type of elements held in this collection

All Implemented Interfaces:

Sử dụng Kiểu danh sách liên kết trong java

SN	Methods	
1	**void add(int index, Object o)**	Thêm vào vị trí i đối tượng o
2	**boolean add(Object o)**	Thêm đối tượng o vào danh sách
3	**boolean addAll(Collection c)**	
4	**boolean addAll(int index, Collection c)**	
5	**void addFirst(Object o)**	
6	**void addLast(Object o)**	
7	**void clear()**	
8	**Object clone()**	
9	**boolean contains(Object o)**	Kiểm tra có chứa đối tượng o không?
10	**Object get(int index)**	Cho lại đối tượng ở vị trí i
11	**Object getFirst()**	

Sử dụng Kiểu danh sách liên kết trong java

SN	Methods	
12	**Object getLast()**	
13	**int indexOf(Object o)**	Cho lại vị trí của đối tượng o trong danh sách
14	**int lastIndexOf(Object o)**	
15	**Object remove(int index)**	Xóa phần tử thứ i
16	**boolean remove(Object o)**	Xóa đối tượng o
17	**Object removeFirst()**	
18	**Object removeLast()**	
19	**Object set(int index, Object element)**	Cho vị trí phần tử ở vị trí thứ i thành giá trị o
20	**int size()**	Cho lại số lượng phần tử trong danh sách
21	…	

Sử dụng Kiểu danh sách liên kết trong java

```
import java.util.LinkedList;
public class LExample {
public static void main(String args[]) {
   LinkedList<String> L = new LinkeList<String>();
   L.add("Item1"); L.add("Item2"); L.add("Item3");
   System.out.println("Linked List Content: " +L);
   L.addFirst("First Item");
   L.addLast("Last Item");
   System.out.println("Linked List Content :" +L);
   Object firstvar = L.get(0);
   System.out.println("First element: " +firstvar);
}
```

Sử dụng Kiểu danh sách liên kết trong java

```
import java.util.LinkedList;
public class LExample {
public static void main(String args[]) {
LinkedList <String> L = new L<String>();
L.add("Item1"); L.add("Item2"); L.add("Item3");
System.out.println("Linked List Content: " +L);
L.addFirst("First Item"); L.addLast("Last Item");
System.out.println("L Content after addition: " +L);

}
```

Sử dụng Kiểu danh sách liên kết trong java

```
public static void main(String args[]) {

L.set(0, "Changed first item");
Object firstvar2 = L.get(0);
L.removeFirst(); L.removeLast();
System.out.println("L after deletion : " +L);
L.add(0, "Newly added item"); L.remove(2);
System.out.println("Final Content: " +L); }
}
```

Sử dụng Kiểu danh sách liên kết trong java

Bài tập:

Cho file LIST.inp chứa các từ. Dùng kiểu LinkedList của C++ viết chương trình:

- Đọc các từ trong file dữ liệu sau đó cho vào danh sách liên kết. Các từ đã có thì không đưa vào danh sách.
- Ghi các từ trong danh sách ra file LIST.out

Tên chương trình: List1.CPP

Đọc ghi file trong java

Làm thế nào để đọc file???

Làm thế nào để ghi file???

Sử dụng Kiểu danh sách liên kết trong java

```
Scanner kb = new Scanner(new File("List.inp"));
String x= kb.next();  // đọc 1 từ
```

```
PrintStream ps = new PrintStream(new File("List.out"));
ps.print(x)  // ghi 1 từ
```

Sử dụng Kiểu danh sách liên kết trong java

```
public class List1 {
        LinkedList <String> L;
        List1(){  L = new LinkedList<String>();  }
        void readFormFile() throws FileNotFoundException{
                Scanner kb = new Scanner(new File("List.inp"));
                while(kb.hasNextLine()){
                try{
                   String temp = kb.next();  if(!L.contains(temp)) L.addLast(temp);
                } catch(Exception ex) { System.out.println("Error reading file!"); }
        }
}
```

Sử dụng Kiểu danh sách liên kết trong java

```
        void writeToFile() throws FileNotFoundException{
                PrintStream ps = new PrintStream(new File("List.out"));
                for(int i=0;i<L.size();i++) ps.print(L.get(i)+" ");
        }
        public static void main(String[] args) throws FileNotFoundException {
                List1 l = new List1();
                l.readFormFile();
                l.writeToFile();
        }
}
```

Sử dụng Kiểu danh sách liên kết trong java

Cho file LIST2.inp chứa các từ. Dùng kiểu LinkedList của Java viết chương trình:

- Đọc các từ từ file dữ liệu sau đó cho vào danh sách liên kết
- Ghi các từ trong danh sách ra file LIST2.out theo dạng:

+ Từ1 số lần xuất hiện

+ Từ2 số lần xuất hiện

Yêu cầu các từ trong file kết quả xếp theo thứ ***tự từ điển***, không phân biệt chữ thường chữ hoa.

Tên chương trình: List2.java

Thảo luận theo nhóm cách lưu trữ và chèn sao cho hiệu quả!!!!

Tài liệu đọc thêm về danh sách liên kết

https://www.geeksforgeeks.org/data-structures/linked-list/

https://www.hackerearth.com/practice/data-structures/linked-list/singly-linked-list/tutorial/

https://www.programiz.com/dsa/linked-list-types

Link YouTube

https://www.youtube.com/watch?v=NgdwfP7K5n8

https://es.coursera.org/lecture/data-structures/singly-linked-lists-kHhgK