# 03-XA

# 列表

## Java序列

One of the things that Java is good at is giving you this homogeneous view of a reality that's usually very heterogeneous.

邓俊辉
deng@tsinghua.edu.cn

# Interface: 定义

❖ Java支持ADT的一种机制：在同一接口规范下，允许不同的实现

❖
```
interface Geometry { //几何物体
    final double PI = 3.1415926; //常量定义，类定义可直接使用
    double area(); //无参数的接口方法
    boolean inside( Point p ); //带参数的接口方法
}
```

❖ interface不能直接实例化为对象

符合interface定义的任何类，都需要具体地实现其中的接口方法

# Interface: 实现

```
class Disk implements Geometry { //符合Geometry接口的Disk类
   Point c;  double r;
   public Disk( Point center, double radius ) //构造方法
      {c = center; r = radius;}
   public double perimeter() { return 2 * PI * r; } //类方法
   public double area() { return PI * r * r; } //接口实现
   public boolean inside( Point p ) { //接口实现
      double dx = p.x - c.x, dy = p.y - c.y;
      return dx*dx + dy*dy < r*r;
   }
}
```

# 向量接口：Vector.java

```
public interface Vector {
   public int getSize();
   public boolean isEmpty();
   public Object getAtRank( int r ) throws ExceptionBoundaryViolation;
   public Object replaceAtRank( int r, Object obj )
      throws ExceptionBoundaryViolation;
   public Object insertAtRank( int r, Object obj )
      throws ExceptionBoundaryViolation;
   public Object removeAtRank( int r ) throws ExceptionBoundaryViolation;
}
```

# 向量实现1：Vector_Array.java

```
public class Vector_Array implements Vector {
   private final int N = 1024; //数组容量固定
   private Object[] A;  private int n = 0;
   public Vector_Array() { A = new Object[N]; n = 0; }
   public int getSize() { return n; }
   public boolean isEmpty() { return 0 == n; }
   public Object insertAtRank( int r, Object obj ) throws ExceptionBoundaryViolation {
      if ( 0 > r || r > n ) throw new ExceptionBoundaryViolation( "out of range" );
      if ( n >= N ) throw new ExceptionBoundaryViolation( "overflow" );
      for ( int i = n; i > r; i-- ) A[i] = A[i - 1];
      A[r] = obj; n++; return obj;
   }
   /* ..... */
}
```

# 向量实现2：Vector_ExtArray.java

```
public class Vector_ExtArray implements Vector {
   private int N = 8; //数组的初始容量，可不断增加
   /* ..... */
   public Object insertAtRank( int r, Object obj ) throws ExceptionBoundaryViolation {
      if ( 0 > r || r > n ) throw new ExceptionBoundaryViolation( "out of range" );
      if ( N <= n ) { //空间溢出的处理
         N *= 2; Object B[] = new Object[ N ]; //容量加倍
         for ( int i = 0; i < n; i++ ) B[i] = A[i]; A = B; //用B[]替换A[]
      }
      for ( int i = n; i > r; i-- ) A[i] = A[i - 1]; //后续元素顺次后移
      A[r] = obj;  n++;  return obj;
   }
   /* ..... */
}
```

# 序列接口及其实现

❖ interface List

```
{ /* ... */ }
```

class List_DLNode

implements List

```
{ /* ... */ }
```

❖ interface Sequence

extends Vector, List

```
{ /* ... */ }
```

class Sequence_DLNode

extends List_DLNode

implements Sequence

```
{ /* ... */ }
```