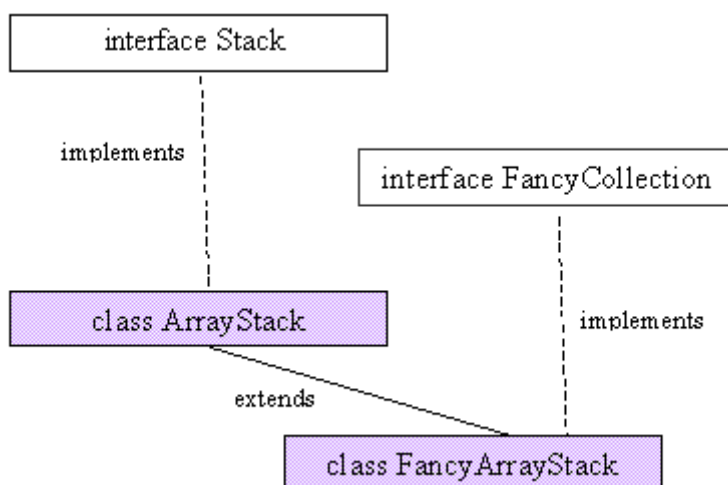


Интерфейсът **FancyCollection** представя операции на колекция с елементи от тип **Object**:

```
public interface FancyCollection {  
    public boolean isEmpty(); //Проверява дали текущата колекция е празна  
    public int size(); //Връща броя на елементите на текущата колекция  
    public void clear(); //Променя състоянието на текущата колекция на празна  
    public boolean contains(Object arg); //Проверява дали arg принадлежи на текущата колекция  
    public Object[] toArray(); //Връща масив от елементите на текущата колекция  
    public Object clone(); //Създава и връща копие на текущата колекция  
    public boolean equals(Object arg); //Проверява дали текущата колекция и колекцията arg съвпадат  
    public String toString(); //Създава и връща низ от елементите на текущата колекция  
}
```

Задача 1: Дадена е йерархията от интерфейси и класове:



Интерфейсът **Stack** представя операции на стек с елементи от тип **Object**:

```
public interface Stack {  
    public boolean isEmpty();  
    public void push(Object x);  
    public Object pop();  
    public Object top();  
}
```

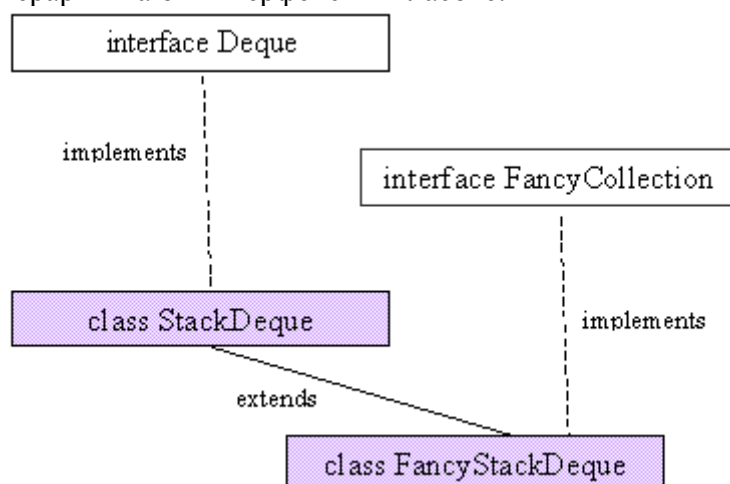
Класът **ArrayStack** е реализация на стек с елементи от тип **Object**:

```
public class ArrayStack implements Stack {  
    //Представяне на стек чрез масив  
    protected Object[] items;  
    protected int top;  
  
    //Конструктори  
    public ArrayStack() {...} //Създава празен стек  
    public ArrayStack(int n) {...} //Създава празен стек с капацитет n  
    public ArrayStack(Object[] arg) {...} //Създава стек от елементите на масива arg, като първият  
        //елемент на масива е на дъното на стека, а последният елемент – на върха  
  
    //Реализация на методите на интерфейса Stack, така че: 1) методите pop и top активират изключение,  
    //ако текущият стек е празен и 2) методът push активира изключение, ако текущият стек е пълен  
  
    //Други методи  
    public boolean isFull() {...}  
    public int capacity() {...}  
}
```

Да се реализира клас **FancyArrayStack**, съгласно спецификацията:

```
public class FancyArrayStack extends ArrayStack implements FancyCollection {
    //Конструктори
    public FancyArrayStack() {...} //Създава празен стек
    public FancyArrayStack(int n) {...} //Създава празен стек с капацитет n
    public FancyArrayStack(FancyCollection arg) {...} //Създава стек от елементите на колекцията arg
    public FancyArrayStack(Object[] arg) {...} //Създава стек от елементите на масива arg, като първият
        //елемент на масива е на дъното на стека, а последният елемент – на върха
    //Наследяване на методите на класа ArrayStack
    //Реализация на методите на интерфейса FancyCollection
    //Предефиниране на методите, наследени от клас Object
}
```

Задача 2: Дадена е йерархията от интерфейси и класове:



Интерфейсът **Deque** представя операции на *дек с елементи от тип Object*:

```
public interface Deque {
    public boolean isEmpty(); //Проверява дали текущия дек е празен
    public void addFront(Object x); //Включва x в началото (левия край) на текущия дек
    public void addRear(Object x); //Включва x накрая (десния край) на текущия дек
    public Object removeFront(); //Изключва елемента в началото на текущия дек и го връща
    public Object removeRear(); //Изключва елемента накрая на текущия дек и го връща
}
```

1. Да се реализира клас **StackDeque**, съгласно спецификацията:

```
public class StackDeque implements Deque {
    //Представяне на дек чрез два стека

    //Конструктори
    public StackDeque() {...} //Създава празен дек
    public StackDeque(int n) {...} //Създава празен дек с начален капацитет n
    public StackDeque(Object[] arg) {...} //Създава дек от елементите на масива arg, като първият
        //елемент на масива е в началото на дека, а последният елемент – накрая

    //Реализация на методите на интерфейса Deque, така че: 1) методите за изключване активират
    //изключение, ако текущият дек е празен и 2) методите за включване увеличават капацитета на текущия
    //дек, ако е необходимо

    //Други методи
}
```

2. Да се реализира клас **FancyStackDeque**, съгласно спецификацията:

```

public class FancyStackDeque extends StackDeque implements FancyCollection {
    //Конструктори
    public FancyStackDeque() {...} //Създава празен дек
    public FancyStackDeque(FancyCollection arg) {...} //Създава дек от елементите на колекцията arg
    public FancyStackDeque(Object[] arg) {...} //Създава дек от елементите на масива arg, като първият
        //елемент на масива е в началото на дека, а последният елемент - накрая
    //Наследяване на методите на интерфейса Deque
    //Реализация на методите на интерфейса FancyCollection
    //Предефиниране на методите, наследени от клас Object
}

```

Задача 3: Пример за използване на клас **StackDeque**: клас **PalindromChecker3** за проверка дали дума е палиндром:

```

public class PalindromChecker3 {

    public void check(String str) {
        Deque charDeque = new StackDeque();

        for(int i = 0; i < str.length(); i++)
            charDeque.addRear(str.charAt(i));

        boolean flag = true;
        while( !charDeque.isEmpty() ) {
            char charLeft = (Character)charDeque.removeFront();
            if( charDeque.isEmpty() ) break;
            char charRight = (Character)charDeque.removeRear();
            if(charLeft != charRight) {
                flag = false;
                break;
            }
        }
        System.out.println(flag ? "Yes" : "No");
    }
}

```

Задача 4: Пример за използване на интерфейс **FancyCollection**: клас **CollectionPrinter** за извеждане на елементите на колекция:

```

public class CollectionPrinter {

    public void print( FancyCollection arg ) {
        System.out.println( arg.toString() );
    }
}

class TestCollectionPrinter {
    public static void main(String[] args) {
        CollectionPrinter w = new CollectionPrinter();

        Character[] d = { new Character('H'), new Character('e'),
            new Character('l'), new Character('l'),
            new Character('o'), new Character('!') };

        System.out.println("Printing stack...");
        w.print(new FancyArrayStack(d));

        System.out.println("Printing deque...");
        w.print(new FancyStackDeque(d));
    }
}

```

Задача за упражнение: Променете клас **PalindromChecker3**, така че да се използва родов клас **java.util ArrayDeque<E>** - виж <http://java.sun.com/javase/6/docs/api/>.