Various Useful Class Types
Class
StringBuffer/StringBuilder

• Since the String class is immutable, it might not suit our purposes all the time.

• Use class StringBuffer/StringBuilder for efficient construction of Strings
  - append() method appends a string representation of any object or value to the buffer
  - insert() is able to insert objects or values at any position in the string
  - toString() method converts result to a String object
StringBuffer/StringBuilder (cont.)

[JAVA]
StringBuffer append(Object obj)
StringBuffer insert(offset,...)
void setCharAt(index, char)
void setLength(length)
String toString();

[C#]
StringBuilder append(Object obj)
StringBuilder insert(offset,...)
String toString();

- Consult the StringBuffer/StringBuilder documentation
StringBuffer (cont.)

• Example in Java. C# code is very similar:

```java
public String concatAll
    (String sa[])
{
    StringBuffer b = new StringBuffer();
    for (Integer i = new Integer(0);
            i.intValue()<sa.length;
            i = new Integer(i.intValue()+1))
    {
        b.append(sa[i]);
    }
    return b.toString();
} // end of concatAll
```
Vector/ArrayList Class

- Array sizes cannot be changed after declaration.
- In addition, one can place any combination of class type objects inside it.
- To declare it:
  - Vector v = new Vector();
  - ArrayList a = new ArrayList();
- To add elements one uses
  - [Java] public void addElement(Object obj)
  - [Java] public void insertElementAt(Object obj, index);
  - [C#] public void Add(Object obj)
  - [C#] public void Insert(index, Object obj)

To remove elements one uses
- [Java] public boolean removeElement(Object obj)
- [Java] public void removeElementAt(index)
- [C#] public void Remove(Object obj)
- [C#] public void RemoveAt(index)
Vector (cont.)

• To retrieve elements one uses:
  - public Object `elementAt(int index);`
  - public Object `firstElement();`
  - public Object `lastElement();`

• When retrieving elements, one needs to change the type of the returned object to the desired one
  Integer i = new Integer(2);
  Vector v = new Vector();
  v.addElement(i);
  Integer j;
  j = v.firstElement(); // will give compile error
  j = (Integer) v.firstElement(); // OK

• The size of the Vector is given by method `size()`

• In C# retrieving elements from ArrayList is a bit more complicated and uses OOP techniques more cleverly.
**Stack**

- A stack is a data structure that allows one to push and pop items just like everyday stacks.
- You need to import `java.util.Stack`/`System.Collections`.
- You create an object of Stack using the `new` operator.
- You use the following methods on objects of stack
  - `public Object pop()`
  - `public Object push(Object item)`
Quiz

• True or False: arrays can be made bigger when required?
• True or False: Strings can be changed and modified?
• In what way are StringBuffer objects different from Strings?
• What is the method that returns a character from a String at a specific index?
Quiz

• False
• False
• StringBuffers can be modified internally while strings are immutable
• charAt()/[]
Exercises

1. Write a method which invokes another method and a global variable which reside in another file

2. Write a static method called `removeString` having the following signature:
   ```java
   public static String removeString
       (String string,
        String substring)
   ```

   The function of `removeString` is to return a new string that does not have any occurrence of the given substring from the given string.

   For instance, the following call:
   ```java
   String result =
       removeString ("undaunted", "un")
   ```

   results in the string “dated”
Exercises

3. Make use of vectors/arraylist to store a series of numbers from the user, stopping when the user enters -1