String Operations in Java

Introduction

- String manipulation is the most common operation performed in Java programs. The easiest way to represent a String (a sequence of characters) is by using an array of characters.
  - Example:
  - char place[] = new char[4];
  - place[0] = ‘J’;
  - place[1] = ‘a’;
  - place[2] = ‘v’;
  - place[3] = ‘a’;
- Although character arrays have the advantage of being able to query their length, they themselves are too primitive and don’t support a range of common string operations. For example, copying a string, searching for specific pattern etc.
- Recognising the importance and common usage of String manipulation in large software projects, Java supports String as one of the fundamental data type at the language level. Strings related book keeping operations (e.g., end of string) are handled automatically.

String Operations in Java

- Following are some useful classes that Java provides for String operations.
  - String Class
  - StringBuffer Class

String Class

- String class provides many operations for manipulating strings.
  - Constructors
  - Utility
  - Comparisons
  - Conversions
- String objects are read-only (immutable)

Strings Basics

- Declaration and Creation:
  - **String** stringName;
  - stringName = **new String** ("string value");
  - Example:
    - String city;
    - city = new String ("Bangalore");
  - Length of string can be accessed by invoking length() method defined in String class:
    - int len = city.length();

String operations and Arrays

- Java Strings can be concatenated using the + operator.
  - String city = “New” + “York”;
  - String city1 = “Delhi”;
  - String city2 = “New “+city1;
- Strings Arrays
  - String city[] = new String[5];
  - city[0] = new String(“Melbourne”);
  - city[1] = new String(“Sydney”);
  - …
  - String megacities[] = {“Brisbane”, “Sydney”, “Melbourne”, “Adelaide”, “Perth”};

String class - Constructors

public String()	Constructs an empty String.
Public String(String value)	Constructs a new string copying the specified string.

String – Methods

public int length()	Returns the length of the string.
public charAt(int index)	Returns the character at the specified location (*index)*
public int compareTo(String anotherString) public int compareToIgnoreCase(String anotherString)	Compare the Strings.
reigonMatch(int start, String other, int ostart, int count)	Compares a region of the Strings with the specified start.

String – Methods

public String replace(char oldChar, char newChar)	Returns a new string with all instances of the *oldChar* replaced with *newChar.*
public trim()	Trims leading and trailing white spaces.
public String toLowerCase() public String toUpperCase()	Changes as specified.
Public boolean equals(String anotherString) Public bolean equalsIngnoreCase(String s1)	Returns true if Strings are equal.

String – methods

Public String concat(String S1)	*Returns concatenated string*
Public String substring(int beginIndex, int endIndex) Public String substring(int beginIndex)	*Returns substring starting from beginIndex character to endIndex character(not include endIndex character)* *Returns substring from beginIndex character to end .*
Public int indexOf(String s) Public int indexOf(String s, int beginIndex)	Returns the index within this string of the first occurrence of the specified string
String valueOf(Variable/Object)	Returns String representation of passed data argument.

String methods

Char[] toCharArray()	*Convert this string to character array.*

toString() Method

- toString() method is a special method that can be defined in any class.
- This method should return a String argument.
- When an object is used in a String concatenation operation or when specified in print statement, this method gets invoked automatically.

toString() Method -Example

```
class Circle    {
    double x, y,r;
     public Circle (double centreX, double centreY, double radius ) {
              x = centreX ; y = centreY;  r = radius;

     }

     public String toString()
     {
           String s = “I am a Circle with  centre [“ + x + “,” + y + “]
    and radius [“+  r + “]”;
            return  s;
    }

 }
```

toString() Method -Example

```
class CircleTest    {

    Circle c = new Circle(10,20, 30);

    System.out.println( c );
          // I am a circle with centre [10.0,20.0] and radius [30.0]

}
```

String Class - example

```
// StringDemo.java: some operations on strings
class StringDemo {
        public static void main(String[] args)
        {
                String s = new String("Have a nice Day");

                // String Length =  15
                System.out.println("String Length = " + s.length() );

                // Modified String =  Have a Good Day
                System.out.println("Modified String = " + s.replace('n', 'N'));

                // Converted to Uppercse =  HAVE A NICE DAY"
                System.out.println("Converted to Uppercase = " + s.toUpperCase());

                // Converted to Lowercase =  have a nice day"
                System.out.println("Converted to Lowercase = " + s.toLowerCase());
        }
}
```

StringDemo Output

- java StringDemo

```
String Length = 15
Modified String = Have a Nice Day
Converted to Uppercase = HAVE A NICE DAY
Converted to Lowercase = have a nice day
```

- Arrays [1:131]

Some questions on String

Which of these method of class String is used to obtain length of String object?
a) get()
b) Sizeof()
c) lengthof()
d) length()

Which of these method of class String is used to extract a single character from a String object?

a) CHARAT()

b) chatat()

c) charAt()

d) ChatAt()

Which of these constructors is used to create an empty String object?
a) String()
b) String(void)
c) String(0)
d) None of the mentioned

What is the output of this program?

```
class String_demo {
public static void main(String args[]){
char chars[] = {'a', 'b', 'c'};
String s = new String(chars);
System.out.println(s);
}}
```

a) a   b) b   c) c   d) abc

What is the output of this program?

```
class String_demo {
public static void main(String args[]){
int ascii[] = { 65, 66, 67, 68};
String s = new String(ascii, 1, 3);
System.out.println(s);
}}
```

a) ABC b) BCD c) CDA d) ABCD

What is the output of this program?

```
class String_demo {
public static void main(String args[]){
char chars[] = {'a', 'b', 'c'};
String s = new String(chars);
String s1 = "abcd";
int len1 = “abcd”.length();
int len2 = s.length();
System.out.println(len1 + " " + len2);
}}
```

a) 3 0 b) 0 3 c) 3 4 d) 4 3

What is the output of following (Assuming written inside main)

```
String s1 = "Amit";
String s2 = "Amit";
String s3 = new String("abcd");
String s4 = new String("abcd");
System.out.println(s1.equals(s2));
System.out.println((s1==s2));
System.out.println(s3.equals(s4));
System.out.println((s3==s4));
```

Ouput

```
true
 true
 true
 false
```

Stringbuffer class

StringBufferClass

- Unlike the String class, StringBuffer class is mutable (changeable).
- Use StringBufferClass class in operations where the string has to be modified.

StringBuffer class - Constructors

public StringBuffer()	Constructs a StringBuffer with an empty string.
public StringBuffer(String str)	Constructs a StringBuffer with initial value of str.

StringBuffer class – Some operations

public int length()	Returns the length of the buffer
public synchronized void setCharAt(int index, char ch)	Replaces the character at the specified position
s1.setLength(int n)	Truncates or extends the buffer. If(n<s1.length(), s1 is truncated; else zeros are added to s1.
public StringBuffer **append**(String str)	Appends the string to this string buffer.
public StringBuffer **append**(int i) Append of other data items (float, char, etc.) is supported.	Appends the string representation of the int argument to this string buffer.

Inserting a String in Middle of Existing StringBuffer

- StringBuffer str = new StringBuffer(“Object Language”);
- String aString = new String(str.toString());
- Int pos = aString.indexOf(“ Language”);
- str.insert(pos, “ Oriented “);
- what will out put of at this point:
  - System.out.println(“Modified String:”+str);
- What will be string after executing (modifying character):
  - str.setChar(6,’-’);

String buffer Class

The Java StringBuffer class (in the java.lang package) encapsulates the array-doubling strategy into an String-like class Constructor method:

```
  public StringBuffer (); // E.g new StringBuffer()
```

Instance Methods:

```
  public void append (String s);
  public void append (char c);
  public void append (int i);
  public void append (boolean b);
     … many other append() methods …
  public String toString ();
```

There are many other constructor and instance methods: see the API.

Summary

- Java provides enhanced support for manipulating strings and manipulating them appears similar to manipulating standard data type variables.
- String is immutable object
- A special method, toString(), can be defined in any Java class, which gets invoked when one tries to concatenation operation with Strings.