Lecture Notes: Esc 101

Date: 12/02/2008

Last class we have studied two 'String' class constructors. Now we will see more methods in the String class.

Please find below some methods of String class.

```
Class String
{
   String ();  // Default Constructor
               // This ctor constructs the string of length 0. And it is not null.

  String(String s); // Clone constructor:
                    // It initializes the newly created object with value passed.


  char charAt(int index);
               // This function returns the char present at index location in the string.

   int length();
               // This method is used to find the length of a String.

   int compareTo(String s);
              // This method compares two strings, one from the this object and other from the S.
              //  Returns,
              // 0 : if both strings are equa.
             // negative value  : if second string is greater than first string
            // positive value    : if first string is greater than second string
}
```

Here in the method 'charAt' it returns 'char' data. So we now see what is char datatype in the next few lines.

Char datatype: This is used to represent the Unicode characters. Please note that this is different that the 'C 'char datatype.
There are almost 40000 different symbols in the Unicode representation. So java char can represent these many symbols.
It stores a code in a Unicode space.

e.g. 'A' -> this would be number in Unicode space. And this is a constant. Let us see different representation of constants.

- Decimal Representation
  e.g. 5

- Hexadecimal Representation
  e.g. 0X32 (which is 50 in decimal representation)

- Octal Representation
  e.g. 062 (which is 50 in decimal representation)

Now let us see how to use the methods that we have discussed. Let us consider the example to reverse a string.

If we want to reverse a string "ABC", we write the code as;

```
String S = "ABC", t = "";
int i;
for(I = 0; I < 3; i++)
{
   t = S.charAt(i) + t;
}
```

However this code will not work if our length of the string is not 3. For example for string "AB" or "ABCD", the above code will fail to reverse the string. And also it fails when string S is null.
So we use the method 'length 'of a String class to take care of this and the newly modified code to reverse the string is as below.

```
String Reverse(String S)
{
   String t = "";
   int i;

   // To check that the string is null or not
   if(S == null)
      return null;

  // Following code will work for any length string as we are using length method of string.
   for(I = 0; I < S.length(); i++)
   {
      t = S.charAt(i) + t;
   }
   return t;
}
```

Now let us see how to check that a given string is 'palindrome' or not.
To do this we use the 'Reverse' method that we have discussed.

```
Boolean isPalindrome(String S)
{
   String t = Reverse(S);

   if(S.compareTo(t) == 0)
      return true;

   return false;
}
```

However above code will not work when the string S is null, because method 'compareTo' don't work on null objects. Hence we have to check that String S is null or not. And also we simplify the code using tertiary operator. And hence the new code to check string is palindrome or not is;

```
Boolean isPalindrome(String S)
{
  return  (s == null) || (S.compareTo(Reverse(S)) == 0) ? true  : false;
}
```

So just a one line code to check given string is 'Palindrome or not'.