```
Lecture Notes
K.Yellaswamy
    Assistant Professor
CMR College of Engineering & Technology

Building Strings and Exploring String Class:
--------------------------------------------


The String class
-----------------
String: A String is a sequence of characters.

ex:- name,address,creditcard no,etc..,
In java,Any string is object of String class.
    ->String is not a character array.

The String class has 11 constructors and more than
40 methods.

create String object:-
--------------------
1.We can declare a String type variable and
initialize it directly with a group of characters.
    ex:- String st = "yellaswamy";
2.We can create a String object using new operator
and pass a group of characters to the object.
    ex:- String s1 = new String("Ashok");
3.We can create a character array into a string by
passing it to the  String object.
    ex:- char arr[] = {'K','s','w','a','m','y'};
         String s2 = new String(arr);
    ex:- String s3 = new String(arr,1,4);
         o/p: swamy

Ex1:
```

```
33 class Test
34 {}{
35 public static void main(String... args)
36 {}{
37 //create a String object or constructing a String
38 //1.We can declare a String type variable and
   initialize it directly with a group of characters.
39 String str1="Yellaswamy";
40 System.out.println(str1);
41
42 //2.We can create a String object using new
   operator and pass a group of characters to the
   object.
43 String str2=new String("Ashok");
44 System.out.println(str1);
45
46 //3.We can create a character array into a string
   by passing it to the  String object.
47
48 char arr[] =
   {}{'K','y','e','l','l','a','s','w','a','m','y'};
49          String s1 = new String(arr);
50          System.out.println(s1);
51          String s2 = new String(arr,1,10);
52          System.out.println(s2);
53 }
54 }
55
56 output:
57
58 D:\Yellaswamy_ClassNotes\Strings>javac Test.java
59
60 D:\Yellaswamy_ClassNotes\Strings>java Test
61 Yellaswamy
62 Yellaswamy
63 Kyellaswamy
```

```
yellaswamy

D:\Yellaswamy_ClassNotes\Strings>
-----------------------------------------
Types of Obejcts:-
    There are 2 types.
    1.Mutable
    2.Immutable
1.Mutable:-
    A mutable object is an object where content can
    be modified.
2.Immutable:-
    An immutable object is an object whose content
    can not be modified.

=>String objects are immutable.its contents cannot
be changed.


String class methods:-
--------------------
    String class belongs to java.lang package.
1.String concat(String str):-
    concatenates the calling string with str.
    note:- '+' will also do the same.
    ex:-  String s1="hydera"; String s2 = "bad";
        String s3 = s1.concat(s2);
        o/p: hyderabad
2.int length():-
    Returns the length of the string.
    String s1 = "Vijayawada";
        int n = s1.length();
3.char charAt(int i):-
    It extracts only one character from given
    String.That      character at the same ith
    place.
```

```
 95 4.int compareTo(String str):-    (case sensitive)
 96      (or)
 97    int compareToIgnoreCase(String str):-  (case
       insensitive)
 98      They are used to compare two strings.
 99 5.boolean equals(String str):- (case sensitive)
100      (or)
101    boolean equalsIgnoreCase(String str):-   (case
       insensitive)
102      It returns true if the calling string equals to
         str.
103 6.boolean startsWith(String prefix):-
104      It returns true if the calling string starts
         with prefix(begins)
105      prefix - sub string (or) not.
106 7.boolean endsWith(String suffix):-
107      It returns true if the invoking string ends
         with suffix.
108
109 Note:- The above two methods use case sensitive
    comparison.
110 8.int indexOf(String str):-
111      It returns the position number of substring in
         the main      String.So we have to pass
         substring.
112      It returns the first occurance of str in the
         string.
113      EX:- "This is a book"
114           int n = str.indexOf("is");
115           o/p :- n = 2
116 9.int lastIndexOf(String str):-
117      It returns last Occurance in the string.
118 10.String replace(char oldchar,char newchar):-
119      It returns a new String that is obtained by
         replacing all characters of 'oldchar' in string
         with 'newchar'.
```

```
120
121 11.String substring(int beginIndex):-
122     It ruturns a new String consisting of all
        characters from beginindex until the end of the
        String.
123
124 12.String substring(int beginIndex,int endIndex):-
125     It returns a new String consisting all
        characters from  beginindex until
        endindex(exclusive).
126
127 13.String toLowerCase():-
128     It converts all characters into lowercase and
        returns.
129
130 14.String toUpperCase():-
131     IT converts all characters into uppercase and
        returns.
132
133 15.String trim():-
134     It eleminates all leading and trailing spaces.
135 ------------------------------------------------
    ---
136 StringBuffer Objects are mutable.
137
138 Creating StringBuffer objects:-
139 ----------------------------
140 1.StringBuffer sb = new StringBuffer("hello");
141 2.StringBuffer sb = new StringBuffer(50);
142 3.StringBuffer sb = new StringBuffer();
143
144 java.lang.StringBuffer methods:-
145 ------------------------------
146 1)StringBuffer append(x):-
147     x may be int,float,double,char,String (or)
        StringBuffer.It      will be appended to the
```

```
        calling StringBuffer.
148 2)StringBuffer insert(int offset,x):-
149     x may be int,float,double,char,String (or)
        StringBuffer.It     will be inserted into the
        StringBuffer at offset.
150 3)StringBuffer delete(int start,int end):-
151     Removes the characters from start to end
        position.
152 4)StringBuffer reverse():-
153     It reverses the all characters in the
        StringBuffer.
154 5)String toString():-
155     Converts the StringBuffer into the String.
156     purpose:- converts StringBuffer to string
        class.
157 6)int length():-
158     returns the length of the StringBuffer.
159 7)int indexOf(String str):-
160     It returns the first position of subString
        'str' in the          StringBuffer object.
161 8)int lastIndexOf(String str):-
162     It returns the last Occurance of substring
        'str' in the          StringBuffer object.
163
164
165 StringBuilder:-
166 --------------
167     This class has been added in jdk1.5.0 which has
        same features like StringBuffer class.These
        objects are also mutable as are the
        StringBuffer objects.
168
169
170 IIQ)What is the difference between the StringBuffer
    and StringBuilder     classes?
171 r) StringBuffer class is synchronized and
```

StringBuilder is not.When the programmer wants to use several threads,he should use StringBuffer as it gives reliable results.If only one thread is used,StringBuilder is prefered,as it improves execution time.

H.W:

1)sort a given group of strings into alphabetical order.

2)find the position of substring in a given main string.

3)Test whether a given string is palindrome or not.

Ex2:

```
D:\Yellaswamy_ClassNotes>javap java.lang.String
Compiled from "String.java"
public final class java.lang.String extends
java.lang.Object implements java.io.
Serializable,java.lang.Comparable,java.lang.CharSeq
uence{□{
    public static final java.util.Comparator
    CASE_INSENSITIVE_ORDER;
    public java.lang.String();
    public java.lang.String(java.lang.String);
    public java.lang.String(char[]);
    public java.lang.String(char[], int, int);
    public java.lang.String(int[], int, int);
    public java.lang.String(byte[], int, int, int);
    public java.lang.String(byte[], int);
    public java.lang.String(byte[], int, int,
    java.lang.String)       throws jav
```

```
197 a.io.UnsupportedEncodingException;
198     public java.lang.String(byte[], int, int,
        java.nio.charset.Charset);
199     public java.lang.String(byte[],
        java.lang.String)       throws java.io.Unsup
200 portedEncodingException;
201     public java.lang.String(byte[],
        java.nio.charset.Charset);
202     public java.lang.String(byte[], int, int);
203     public java.lang.String(byte[]);
204     public
        java.lang.String(java.lang.StringBuffer);
205     public
        java.lang.String(java.lang.StringBuilder);
206     java.lang.String(int, int, char[]);
207     public int length();
208     public boolean isEmpty();
209     public char charAt(int);
210     public int codePointAt(int);
211     public int codePointBefore(int);
212     public int codePointCount(int, int);
213     public int offsetByCodePoints(int, int);
214     void getChars(char[], int);
215     public void getChars(int, int, char[], int);
216     public void getBytes(int, int, byte[], int);
217     public byte[] getBytes(java.lang.String)
        throws java.io.UnsupportedEnc
218 odingException;
219     public byte[]
        getBytes(java.nio.charset.Charset);
220     public byte[] getBytes();
221     public boolean equals(java.lang.Object);
222     public boolean
        contentEquals(java.lang.StringBuffer);
223     public boolean
        contentEquals(java.lang.CharSequence);
```

```
224     public boolean
        equalsIgnoreCase(java.lang.String);
225     public int compareTo(java.lang.String);
226     public int
        compareToIgnoreCase(java.lang.String);
227     public boolean regionMatches(int,
        java.lang.String, int, int);
228     public boolean regionMatches(boolean, int,
        java.lang.String, int, int);
229     public boolean startsWith(java.lang.String,
        int);
230     public boolean startsWith(java.lang.String);
231     public boolean endsWith(java.lang.String);
232     public int hashCode();
233     public int indexOf(int);
234     public int indexOf(int, int);
235     public int lastIndexOf(int);
236     public int lastIndexOf(int, int);
237     public int indexOf(java.lang.String);
238     public int indexOf(java.lang.String, int);
239     static int indexOf(char[], int, int, char[],
        int, int, int);
240     public int lastIndexOf(java.lang.String);
241     public int lastIndexOf(java.lang.String, int);
242     static int lastIndexOf(char[], int, int,
        char[], int, int, int);
243     public java.lang.String substring(int);
244     public java.lang.String substring(int, int);
245     public java.lang.CharSequence subSequence(int,
        int);
246     public java.lang.String
        concat(java.lang.String);
247     public java.lang.String replace(char, char);
248     public boolean matches(java.lang.String);
249     public boolean
        contains(java.lang.CharSequence);
```

```
250     public java.lang.String
        replaceFirst(java.lang.String,
        java.lang.String);
251     public java.lang.String
        replaceAll(java.lang.String, java.lang.String);
252     public java.lang.String
        replace(java.lang.CharSequence,
        java.lang.CharSequen
253 ce);
254     public java.lang.String[]
        split(java.lang.String, int);
255     public java.lang.String[]
        split(java.lang.String);
256     public java.lang.String
        toLowerCase(java.util.Locale);
257     public java.lang.String toLowerCase();
258     public java.lang.String
        toUpperCase(java.util.Locale);
259     public java.lang.String toUpperCase();
260     public java.lang.String trim();
261     public java.lang.String toString();
262     public char[] toCharArray();
263     public static java.lang.String
        format(java.lang.String, java.lang.Object[]);
264
265     public static java.lang.String
        format(java.util.Locale, java.lang.String, ja
266 va.lang.Object[]);
267     public static java.lang.String
        valueOf(java.lang.Object);
268     public static java.lang.String valueOf(char[]);
269     public static java.lang.String valueOf(char[],
        int, int);
270     public static java.lang.String
        copyValueOf(char[], int, int);
271     public static java.lang.String
```

```
        copyValueOf(char[]);
272     public static java.lang.String
        valueOf(boolean);
273     public static java.lang.String valueOf(char);
274     public static java.lang.String valueOf(int);
275     public static java.lang.String valueOf(long);
276     public static java.lang.String valueOf(float);
277     public static java.lang.String valueOf(double);
278     public native java.lang.String intern();
279     public int compareTo(java.lang.Object);
280     static {};};
281 }
282
283
284 ----------------------------------------
285 Programs:
286 1.
287 import java.io.*;
288 public class str1
289 {}{
290   public static void main(String args[])
291   {
292         String st1=new String();
293         st1="Java";
294         byte b[]={65,66,67,68,69,70};
295         String st2=new String(b);
296         String st3=new String(b,1,3);
297         System.out.println(st1);
298         System.out.println(st2);
299         System.out.println(st3);
300    }
301 }
302
303
304 output:
305 D:\Yellaswamy_ClassNotes\Strings>java str1
```

```
Java
ABCDEF
BCD
-----------------
Ex2:
import java.io.*;
public class str2
{}{
  public static void main(String args[])
  {
        char ch[]={'a','b','c','d','e','f'};
        String st1=new String("Java");
        StringBuffer sb=new StringBuffer("cmrcet");
        String st2=new String(ch);
        String st3=new String(ch,2,4);
        String st4=new String(sb);
        System.out.println(st1);
        System.out.println(st2);
        System.out.println(st3);
        System.out.println(st4);
   }
}
output:

D:\Yellaswamy_ClassNotes\Strings>java str2
Java
abcdef
cdef
cmrcet

------------------------------
Ex3:
import java.io.*;
public class str3
{}{
  public static void main(String args[])
```

```
342    {
343        String s=new String("welcome");
344        int i;
345        char ch;
346        i=s.length();
347        ch=s.charAt(5);
348        System.out.println("Length of given string
                  :  "+i);
349        System.out.println("The character at given
           index   :  " +ch);
350    }
351 }
352
353 output:
354
355 D:\Yellaswamy_ClassNotes\Strings>java str3
356 Length of given string          :  7
357 The character at given index   :  m
358
359 ------------------------------------------------
    ----
360 Ex4:
361 import java.io.*;
362 public class str4
363 {}{
364    public static void main(String args[])
365    {
366           String s1=new String("java");
367           String s2=new String("JAVA");
368           String s3=new String("welcome");
369           int n;
370
371           n=s1.compareTo(s3);
372           if(n==0)
373               System.out.println("s1 and s3 are
                  equal");
```

```
374             else if(n>0)
375                  System.out.println("s1 is greater
                     than s3");
376             else
377                  System.out.println("s1 is less than
                     s3");
378
379             n=s1.compareTo(s2);
380              if(n==0)
381                  System.out.println("s1 and s2 are
                     equal");
382             else if(n>0)
383                  System.out.println("s1 is greater
                     than s2");
384             else
385                  System.out.println("s1 is less than
                     s2");
386
387             n=s1.compareToIgnoreCase(s2);
388              if(n==0)
389                  System.out.println("s1 and s2 are
                     equal");
390             else if(n>0)
391                  System.out.println("s1 is greater
                     than s2");
392             else
393                  System.out.println("s1 is less than
                     s2");
394         }
395 }
396 output:
397
398 D:\Yellaswamy_ClassNotes\Strings>java str4
399 s1 is less than s3
400 s1 is greater than s2
401 s1 and s2 are equal
```

```
402
403 ------------------------------------
404 Ex:5
405 import java.io.*;
406 class str5
407 {}{
408    public static void main(String args[])
409    {
410           boolean b;
411           StringBuffer sb=new
            StringBuffer("welcome");
412           String s1=new String("welcome");
413           String s2=new String(" to cmrcet");
414           String s3=new String();
415           s3=s1.concat(s2);
416           System.out.println("Concatenated String :
            "+s3);
417           b=s1.contentEquals(sb);
418           if(b==true)
419                 System.out.println("Given 2 strings
                  are equal");
420           else
421                 System.out.println("Given 2 strings
                  are not equal");
422       }
423 }
424 output:
425
426 D:\Yellaswamy_ClassNotes\Strings>java str5
427 Concatenated String : welcome to cmrcet
428 Given 2 strings are equal
429 ------------------------------------
430 Ex:6
431 import java.io.*;
432 class str6
433 {}{
```

```
434    public static void main(String args[])
435    {
436        char ch[]={'a','b','c','d','e','f'};
437        String s1=String.copyValueOf(ch);
438        String s2=String.copyValueOf(ch,1,4);
439        System.out.println(s1);
440        System.out.println(s2);
441     }
442 }
443 output:
444
445 D:\Yellaswamy_ClassNotes\Strings>java str6
446 abcdef
447 bcde
448
449 ----------------------------------------
450 Ex:7
451 import java.io.*;
452 public class str7
453 {}{
454    public static void main(String args[])
455    {
456          String s1=new String("WELCOME");
457          String s2=new String("welcome");
458          boolean b;
459
460         b=s1.endsWith("come");
461          if(b==true)
462             System.out.println("true");
463          else
464             System.out.println("false");
465
466          b=s1.equals(s2);
467          if(b==true)
468             System.out.println("Given 2 strings are
               equal");
```

```
469             else
470                System.out.println("Given 2 strings are
                   not equal");
471
472             b=s1.equalsIgnoreCase(s2);
473             if(b==true)
474                System.out.println("Given 2 strings are
                   equal");
475             else
476                System.out.println("Given 2 strings are
                   not equal");
477       }
478 }
479 output:
480
481 D:\Yellaswamy_ClassNotes\Strings>java str7
482 false
483 Given 2 strings are not equal
484 Given 2 strings are equal
485
486 ------------------------------------------------
487 Ex:8
488 import java.io.*;
489 public class str8
490 {}{
491    public static void main(String args[])
492    {
493          int i;
494          String st=new String("WELCOME");
495          byte bt[]=new byte[10];
496          char ch[]=new char[10];
497          bt=st.getBytes();
498          System.out.println("Byte Array");
499          for(i=0;i<bt.length;i++)
500          {
501               System.out.println(bt[i]+"   ");
```

```
502             }
503             st.getChars(3,7,ch,2);
504             System.out.println("Character Array");
505             for(i=0;i<ch.length;i++)
506             {
507                  System.out.println(ch[i]+"   ");
508             }
509         }
510 }
511 output:
512
513 D:\Yellaswamy_ClassNotes\Strings>java str8
514 Byte Array
515 87
516 69
517 76
518 67
519 79
520 77
521 69
522 Character Array
523
524
525 C
526 O
527 M
528 E
529
530 ---------------------------------------
531 Ex:9
532 import java.io.*;
533 public class str9
534 {}{
535    public static void main(String args[])
536    {
537          String s1=new String("welcome");
```

```
538             String s2=new String("abcdabcdabc");
539             int id,hc;
540             hc=s1.hashCode();
541             System.out.println("Hash Code of given
                String : "+hc);
542             id=s1.indexOf('e');
543             System.out.println("Index : " +id);
544             id=s1.indexOf('e',3);
545             System.out.println("Index : " +id);
546             id=s1.indexOf("come");
547             System.out.println("Index : " +id);
548             id=s2.indexOf("ab",5);
549             System.out.println("Index : " +id);
550         }
551 }
552 output:
553
554 D:\Yellaswamy_ClassNotes\Strings>java str9
555 Hash Code of given String : 1233099618
556 Index : 1
557 Index : 6
558 Index : 3
559 Index : 8
560
561 ---------------------------------------------
562
563
564
```