IMPORTANT CONSTRUCTORS AND METHODS OF STRINGBUFFER CLASS

Constructors of StringBuffer class:

1. **StringBuffer():** It creates an empty string buffer with an initial capacity of 16.

2. **StringBuffer(String str):** It creates a string buffer with the given string.

3. **StringBuffer(CharSequence seq):** It Constructs a string buffer that contains the same characters as the specified CharSequence.

4. **StringBuffer(int capacity):** It creates an empty string buffer with the specified capacity as length.

Methods of StringBuffer class:

1. **append(datatype varname):** The append() method concatenates the given argument with this string.

```
class A {
    public static void main(String[] args)
    {
        StringBuffer sb = new StringBuffer("Red ");
        sb.append("Apple"); // now original string is
changed
        System.out.println(sb);
    }
}
```

Output

```
Red Apple
```

2. **insert(int offset, datatype varname):** The insert() method inserts the given argument at the given position

```
class A {
    public static void main(String args[])
    {
        StringBuffer sb = new StringBuffer("ABCD ");
        sb.insert(1, "PQR");
        // Now original string is changed
        System.out.println(sb);
    }
}
Output
APQRBCD.
```

3. **replace(int start, int end, String str):** The replace() method replaces the given string from the specified beginIndex up to endIndex-1.

```
class A {
    public static void main(String args[])
    {
        StringBuffer sb = new StringBuffer("Hello");
        sb.replace(1, 3, "Mohan");
        System.out.println(sb);
    }
}
```

```
Output
HMohanlo
```

4. **delete(int start, int end):** The delete() method of the StringBuffer class deletes the string from the specified beginIndex to endIndex-1.

```
class A {
   public static void main(String args[])
   {
      StringBuffer sb = new
StringBuffer("Hello");
      sb.delete(1, 3);
      System.out.println(sb);
   }
}
Output
Hlo
```

5. **reverse():** The The reverse() method of the StringBuffer class reverses the current string.

```
class A {
   public static void main(String args[])
   {
      StringBuffer sb = new
StringBuffer("Hello");
      sb.reverse();
      System.out.println(sb);
   }
}
Output
olleH
```

6. **capacity():** The capacity() method of the StringBuffer class returns the current capacity of the buffer.

   The default capacity of the buffer is 16. If the number of characters increases from its current capacity, it increases the capacity by (oldcapacity*2)+2.

   For instance, if your current capacity is 16, it will be (16*2)+2=34.

```
class A {
   public static void main(String args[])
   {
      StringBuffer sb = new StringBuffer();
      System.out.println(sb.capacity()); // default 16
      sb.append("Hello");
      System.out.println(sb.capacity()); // now 16
      sb.append("Mohan is writing a code in java");
      System.out.println(sb.capacity());
      // Now (16*2)+2=34     i.e (oldcapacity*2)+2
   }
}
```

Output

```
16
16
34
```

7. **ensureCapacity(int min) ():** This method ensures that the StringBuffer capacity is at least equal to the mentioned minimum.

8. **charAt(index):** It will return the character at the specified index.

9. **length():** The length of a StringBuffer can be found by the length( ) method.