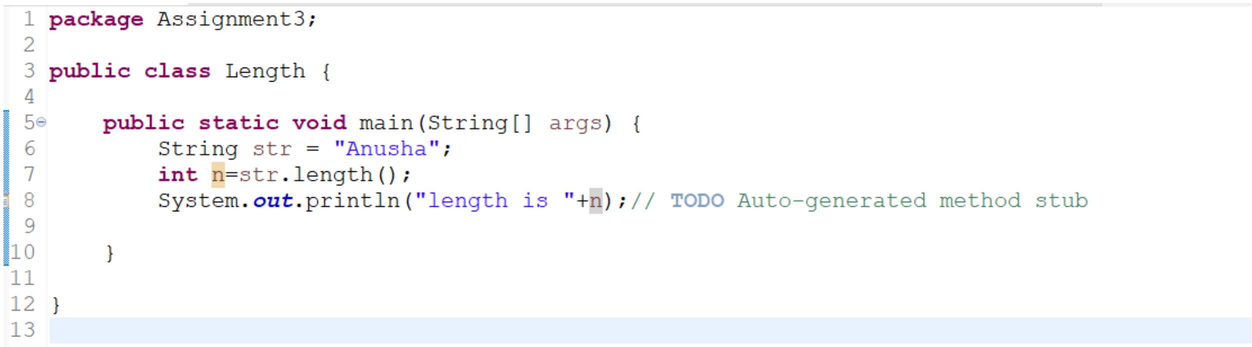
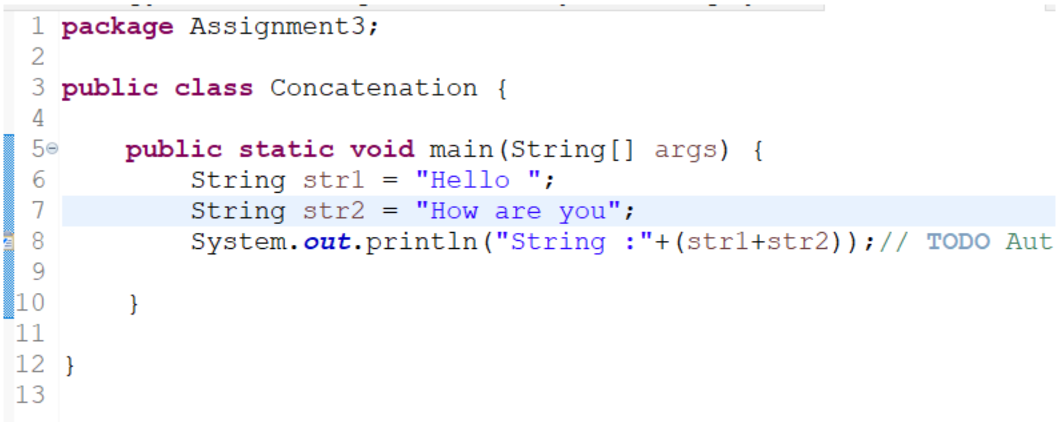
1) Write an application to determine the length of the String str = "**Hello World**". (Hint: Use String method)



```java
package Assignment3;

public class Length {

    public static void main(String[] args) {
        String str = "Anusha";
        int n=str.length();
        System.out.println("length is "+n);// TODO Auto-generated method stub

    }

}

```

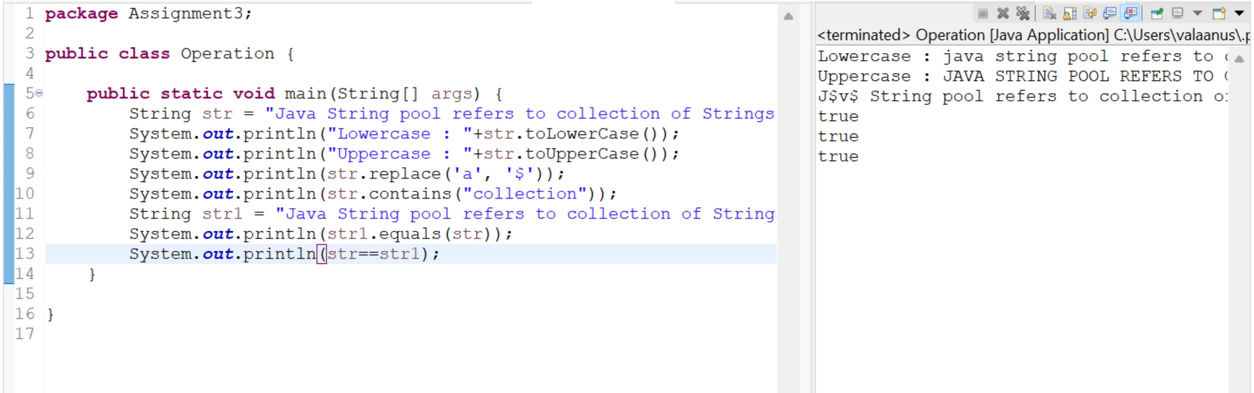
2) Write an application to join the two Strings "**Hello,** " & "**How are you?**" (Hint: Use String method)



```java
package Assignment3;

public class Concatenation {

    public static void main(String[] args) {
        String str1 = "Hello ";
        String str2 = "How are you";
        System.out.println("String :"+(str1+str2));// TODO Aut

    }

}

```

3) Given a String "**Java String pool refers to collection of Strings which are stored in heap memory**", perform the following operations (Hint: all operation can be performed using String methods)

a. Print the string to console in lowercase
b. Print the string to console in uppercase
c. Replace all 'a' character in the string with $ sign
d. Check if the original String contains the word "collection"
e. Check if the following String "**java string pool refers to collection of strings which are stored in heap memory**" matches the original
f. If the string does not match check if there is another method which can be used to check if the strings are equal



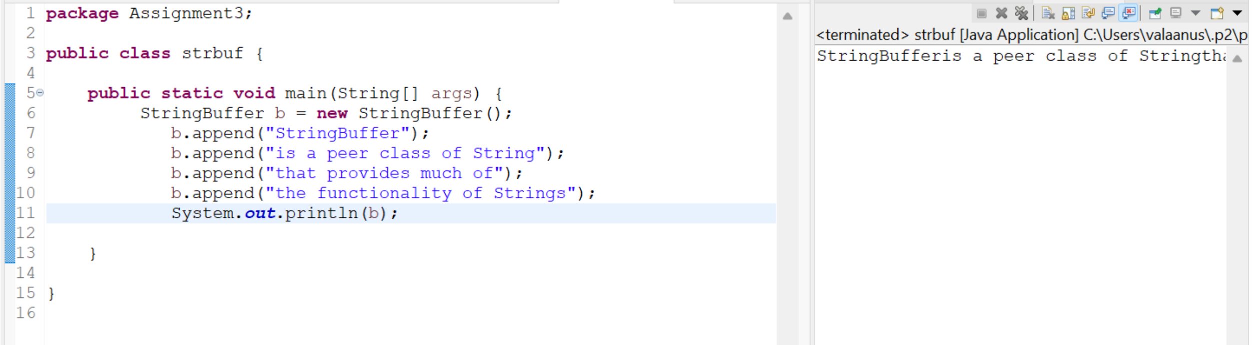
```java
package Assignment3;

public class Operation {

    public static void main(String[] args) {
        String str = "Java String pool refers to collection of Strings
        System.out.println("Lowercase : "+str.toLowerCase());
        System.out.println("Uppercase : "+str.toUpperCase());
        System.out.println(str.replace('a', '$'));
        System.out.println(str.contains("collection"));
        String str1 = "Java String pool refers to collection of String
        System.out.println(str1.equals(str));
        System.out.println(str==str1);
    }

}
```

```
<terminated> Operation [Java Application] C:\Users\valaanus\.
Lowercase : java string pool refers to
Uppercase : JAVA STRING POOL REFERS TO
J$v$ String pool refers to collection o
true
true
true
```

Assignments on StringBuffer Class

Note: StringBuffer is a peer class of String that provides much of the functionality of strings. String represents fixed-length, immutable character sequences while **StringBuffer** represents growable and writable character sequences. **StringBuffer** may have characters and substrings inserted in the middle or appended to the end. It will automatically grow to make room for such additions and often has more characters preallocated than are actually needed, to allow room for growth.

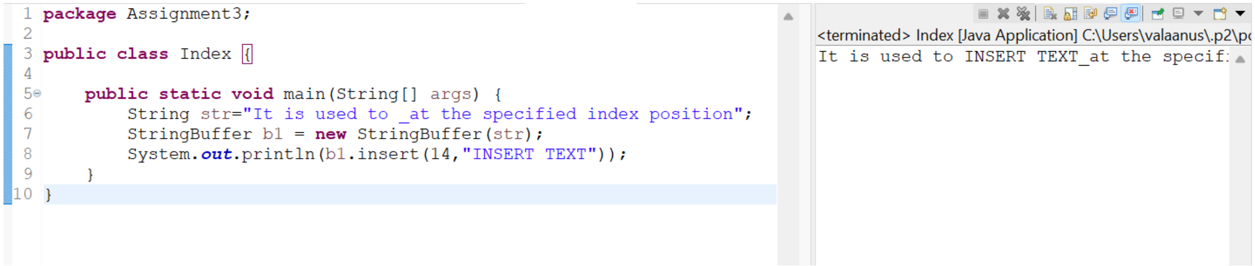
1) Write an application to append the following strings "StringBuffer", "is a peer class of String", "that provides much of ", "the functionality of strings" using a StringBuffer.



```java
package Assignment3;

public class strbuf {

    public static void main(String[] args) {
         StringBuffer b = new StringBuffer();
            b.append("StringBuffer");
            b.append("is a peer class of String");
            b.append("that provides much of");
            b.append("the functionality of Strings");
            System.out.println(b);

    }

}
```

```
<terminated> strbuf [Java Application] C:\Users\valaanus\.p2\p
StringBufferis a peer class of Stringth
```

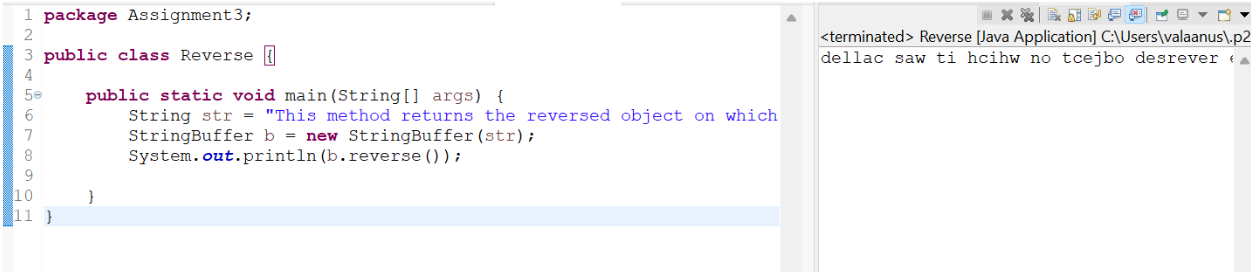
2) Insert the following string "insert text" into the string "It is used to _ at the specified index position" at the location denoted by the sign _



```java
package Assignment3;

public class Index {

    public static void main(String[] args) {
        String str="It is used to _at the specified index position";
        StringBuffer b1 = new StringBuffer(str);
        System.out.println(b1.insert(14,"INSERT TEXT"));
    }
}
```

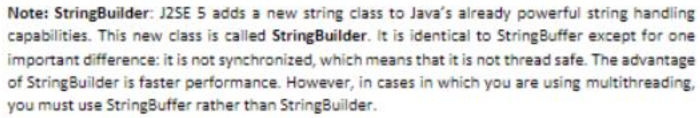
```
<terminated> Index [Java Application] C:\Users\valaanus\.p2\p
It is used to INSERT TEXT_at the specif
```

3) Reverse the following string "This method returns the reversed object on which it was called" using StringBuffer Class



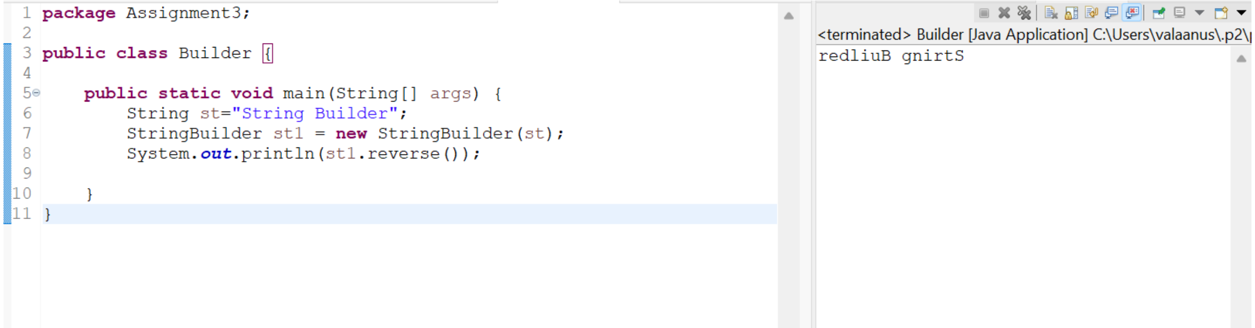
```
package Assignment3;

public class Reverse {

    public static void main(String[] args) {
        String str = "This method returns the reversed object on which
        StringBuffer b = new StringBuffer(str);
        System.out.println(b.reverse());

    }
}
```

```
<terminated> Reverse [Java Application] C:\Users\valaanus\.p2
dellac saw ti hcihw no tcejbo desrever
```



Note: StringBuilder: J2SE 5 adds a new string class to Java's already powerful string handling capabilities. This new class is called **StringBuilder**. It is identical to StringBuffer except for one important difference: it is not synchronized, which means that it is not thread safe. The advantage of StringBuilder is faster performance. However, in cases in which you are using multithreading, you must use StringBuffer rather than StringBuilder.

1) Provide solution for "**Assignments on StringBuffer Class**" using **StringBuilder** class



```
package Assignment3;

public class Builder {

    public static void main(String[] args) {
        String st="String Builder";
        StringBuilder st1 = new StringBuilder(st);
        System.out.println(st1.reverse());

    }
}
```

```
<terminated> Builder [Java Application] C:\Users\valaanus\.p2\
redliuB gnirtS
```