You are presented with three stacks of discs. Two stacks are empty and one stack, the source stack, is filled with black and white discs. Your job is to sort the source stack so that it is itself empty and all black discs are on one stack and all white discs on the other.

Consider the following image as an example:

**Before Sorting**

![Source Stack Black Stack White Stack](image1)

**After Sorting**

![Source Stack Black Stack White Stack](image2)

The stacks used for this exercise follow the Last-in-First-out rules of a *Stack* data type. That means, you can only remove discs from or add discs the top of a stack. Stacks are represented by the Java API class *Stack* which you can look up here: [https://docs.oracle.com/javase/8/docs/api/java/util/Stack.html](https://docs.oracle.com/javase/8/docs/api/java/util/Stack.html)

For this problem you are given a skeleton Java class *StackSort* with some methods left blank. Please follow these steps to implement the solution:
1. Implement the StackSort constructor so that it initialises all three of its Stack instance members. The constructor gets a parameter sourceData which is read from the command line. Fill the source Stack with sourceData's String entries.

2. Implement the sortStacks method so that it empties source and fills black and white according to the specification.

1 StackSort Skeleton

```java
import java.util.Stack;

public class StackSort {

    public static final String BLACK = "B";
    public static final String WHITE = "W";

    private Stack<String> source;
    private Stack<String> black;
    private Stack<String> white;

    public StackSort(String[] sourceData) {
        // IMPLEMENT ME
    }

    public void sortStacks() {
        // IMPLEMENT ME
    }

    public String toString() {
        String result = "Source: " + source;
        result += "Black: " + black;
        result += "White: " + white;
        return result;
    }

    public static void main(String[] args) {
        StackSort ss = new StackSort(args);
        System.out.println("Before sorting:
" + ss);
        ss.sortStacks();
        System.out.println("After sorting:
" + ss);
    }
}
```
import java.util.Stack;

public class StackSort {
    public static final String BLACK = "B";
    public static final String WHITE = "W";

    private Stack<String> source;
    private Stack<String> black;
    private Stack<String> white;

    public StackSort(String[] sourceData) {
        source = new Stack<String>();
        for (String elem : sourceData)
            source.push(elem);

        black = new Stack<String>();
        white = new Stack<String>();
    }

    public void sortStacks() {
        while (!source.isEmpty()) {
            String value = source.pop();
            if (value.equals(BLACK))
                black.push(value);
            else if (value.equals(WHITE))
                white.push(value);
        }
    }

    public String toString() {
        String result = "Source: " + source;
        result += "\Black: " + black;
        result += "\White: " + white;
        return result;
    }

    public static void main(String[] args) {
        StackSort ss = new StackSort(args);

        System.out.println("Before sorting:\n" + ss);
        ss.sortStacks();
        System.out.println("After sorting:\n" + ss);
    }
}