# 1

假设

```
String s1 = "Welcome to Java";
String s2 = s1;
String s3 = new String("Welcome to Java");
```

那么 下面表达式的结果是什么?

(1) s1 == s2
`true`  ==检查的引用的对象是否相同,s1和s2引用到相同的
(2) s1 == s3   `true`
(3) s1.equals(s2)
`true` equals比较内容
(4) s2.equals(s3)
`true` (5) s1.compareTo(s2);
`0`
(6) s2.compareTo(s3);
`0`
(7) s1.charAt(0);
`W`
(8) s1.indexOf('j');
`-1`
(9) s1.indexOf("to");
`8`
(10) s1.lastIndexOf("o",15)
`9`
(11) s1.substring(3, 11);
`come to`
(12) s1.endWith("Java")

```
true
```

(13) s1.startWith("wel");

```
false
```

(14) " We come ".trim();

```
We come
```

(15) s1.toUpperCase();

```
WELCOME TO JAVA
```

(16) s1.replace('o', 'T');

```
WelcTme tT Java
```

## 2

假设s1和s2是两个字符串，下面哪些语句是错误的

```
String s = new String(“new String”);
String s3 = s1 + s2;
String s4 = s1 - s2;
boolean b1 = s1 == s2;
boolean b2 = s1 >= s2;
int i1 = s1.compareTo(s2);
int i2 = s1.length();
char c = s1(0);
char c = s1[0];
char c = s1.charAt(s1.length());
```

1. T
2. T
3. F
4. T
5. F
6. T
7. F
8. F

9. F
10. F

## 3

```
StringBuffer s1 = new StringBuffer("Java");
StringBuffer s2 = new StringBuffer("HTML");
假设下列每个语句是独立的，每条语句结束后，写出s1的结果
(1) s1.append("  is fun");
(2) s1.append(s2);
(3) s1.insert(2, "is fun");
(4) s1.insert(1,s2);
(5) char c = s1.charAt(2);
(6) int i = s1.length();
(7) s1.deleteCharAt(3);
(8) s1.delete(1,3)
(9) s1.reverse();
(10) s1.replace(1,3, "Computer")
(11) s1.substring(1,3);
(12) s1.substring(2);
```

1. Java is fun
2. JavaHTML
3. Jais funva
4. JHTMLava
5. Java
6. Java
7. Jav
8. Ja
9. avaJ
10. JComputera
11. Java
12. Java

## 4

假设StringBuffer s = new StringBuffer("Welcome to JAVA");如何将s的内容清空?

```
sb.setLength(0)
sb = new StringBuffer();
sb.delete(0, sb.length());
```

## 5

利用StringBuffer实现回文字符串的判断。及用户任意输入一个字符串，判断该字符串是否是回文。

```
boolean isPalindrome(){
    Scanner scanner = new Scanner(System.in);
    String s = scanner.nextLine();
    return s.equals(new
StringBuffer(s).reverse().toString());
}
```

## 6

编写一个程序，提示用户输入两个字符串，检测第一个字符串是否是第二个字符串的子串。

```
boolean isSubString(){
    Scanner scanner = new Scanner(System.in);
    String a = scanner.nextLine();
    String b = scanner.nextLine();
```

```
        return a.contains(b) || b.contains(a);
    }
```

## 7

生成一个锯齿形的二维整型数组，一共N行，第一行N列，第二行N-1列，以此递减，第N行1列。并求所有元素的和。

```
public class ShuZu {
    public static void main(String[] args) {
        ShuZu shuZu = new ShuZu();
        System.out.println(shuZu.array(3));
    }

    int array(int N){
        int [][] d = new int[N][];
        int num = 0;
        for (int i = 0; i < N; i++) {
            d[i] = new int[N - i];
            for (int j = 0; j < d[i].length; j++) {
                d[i][j] = num ++;
            }
        }

        int res = 0;
        for (int i = 0; i < d.length; i++) {
            for (int j = 0; j < d[i].length; j++) {
                res += d[i][j];
            }
        }
        return res;
    }
}
```