

3. The `StringChecker` interface describes classes that check if strings are valid, according to some criterion.

```
public interface StringChecker
{
    /** Returns true if str is valid. */
    boolean isValid(String str);
}
```

A `CodeWordChecker` is a `StringChecker`. A `CodeWordChecker` object can be constructed with three parameters: two integers and a string. The first two parameters specify the minimum and maximum code word lengths, respectively, and the third parameter specifies a string that must not occur in the code word. A `CodeWordChecker` object can also be constructed with a single parameter that specifies a string that must not occur in the code word; in this case the minimum and maximum lengths will default to 6 and 20, respectively.

The following examples illustrate the behavior of `CodeWordChecker` objects.

Example 1

```
StringChecker sc1 = new CodeWordChecker(5, 8, "$");
```

Valid code words have 5 to 8 characters and must not include the string "\$".

Method call	Return value	Explanation
sc1.isValid("happy")	true	The code word is valid.
sc1.isValid("happy\$")	false	The code word contains "\$".
sc1.isValid("Code")	false	The code word is too short.
sc1.isValid("happyCode")	false	The code word is too long.

Example 2

```
StringChecker sc2 = new CodeWordChecker("pass");
```

Valid code words must not include the string "pass". Because the bounds are not specified, the length bounds are 6 and 20, inclusive.

Method call	Return value	Explanation
sc2.isValid("MyPass")	true	The code word is valid.
sc2.isValid("Mypassport")	false	The code word contains "pass".
sc2.isValid("happy")	false	The code word is too short.
sc2.isValid("1,000,000,000,000,000")	false	The code word is too long.

Write the complete `CodeWordChecker` class. Your implementation must meet all specifications and conform to all examples.