JavaScript String Functions Demonstration

```
(1) var str="New Hampshire Dept. of Corrections Special School District" ;
```

Original string: **New Hampshire Dept. of Corrections Special School District**

```
(2) var length1 = str.length ;
```

The original string contains **58** characters.

Note that the string length (character count) includes spaces. Also note that `length` *is a* ***property****, not a function. This is why* `length` *is not followed by parentheses.*

```
(3) var space1 = str.indexOf( " " ) ;
```

The first space is at character position: **3**

Note that the character position is 0-based!

```
(4) var word1 = str.substr( 0, space1 ) ;
```

The first word is: **New**

The `substr` *parameters are the starting position and the number of characters.*

```
(5) var space2 = str.indexOf( " ", space1+1 ) ;
```

The second space is at character position: **13**

The second parameter tells JavaScript where to start the character search.

```
(6) var word2 = str.substr( space1+1, space2-space1 ) ;
```

The second word — extracted using the `substr` function — is: **Hampshire**

Remember that the second parameter to the `substr` *function is the* ***number of characters****, not the ending space. This is why we have to subtract* `space1` *from* `space2`.

```
(7) var word2b = str.substring( space1+1, space2 ) ;
```

The second word — extracted using the `substring` function — is: **Hampshire**

The second parameter to the `substring` *function is indeed the ending space. This is why we do* ***not*** *subtract* `space1` *from* `space2`.

```
(8) var phrase1 = word2 + " " + word1 ;
```

Concatenating `word1` onto `word2` yields: **Hampshire New**

Remember that the second parameter is the number of characters, not the ending space. This is why we have to subtract `space1` *from* `space2`.

```
(9) var arrWords = str.split( " " ) ;
```

We can get all the words at once using the `split` function:

0. New
1. Hampshire
2. Dept.
3. of
4. Corrections
5. Special
6. School
7. District

Note that array indexes are 0-based!

```
(10) document.writeln( arrWords[5] + " " + arrWords[7] + " " + arrWords[3] + " " +
    arrWords[1] + " " + arrWords[0] + " " + arrWords[6] + " " + arrWords[4] ) ;
```

We can then print them any way we want!

Special District of Hampshire New School Corrections

```
(11) var phrase2 = str.replaceAll( " ", "|" ) ;
```

Replacing all spaces with vertical bars yields:
New|Hampshire|Dept.|of|Corrections|Special|School|District

The first parameter is the search string and the second is the replacement string.

```
(12) var bStartsWithNew = str.startsWith( "New" ) ;
```

Testing whether the string starts with **New** yields: **true**

```
(13) var bEndsWithNew = str.startsWith( "Hampshire" ) ;
```

Testing whether the string ends with **Hampshire** yields: **false**

```
(14) var strUpper = str.toUpperCase() ;
```

Converting the string to all uppercase yields: **NEW HAMPSHIRE DEPT. OF CORRECTIONS SPECIAL SCHOOL DISTRICT**

```
(15) var strLower = str.toLowerCase() ;
```

Converting the string to all lowercase yields: **new hampshire dept. of corrections special school district**

```
(16) var bIncludesSchool1 = str.includes( "School" ) ;
```

Testing whether the string includes (contains) **School** yields: **true**

Note that the `includes` *function is case-sensitive. This is verified by the next example.*

```
(17) var bIncludesSchool2 = str.includes( "school" ) ;
```

Testing whether the string includes **school** yields: **false**