Methods of String

length():

```
String s = "Good Morning";
           0123456789 10 11
S.O.P(s.length()) // 12
```

toLowerCase():

```
String s = "Good Morning";
S.O.P(s.toLowerCase()); // good morning
```

toUpperCase():

```
S.O.P(s.toUpperCase) //
```

GOOD MORNING

Contains():

```
String s = "Mississippi";
S.O.P(s.contains("issi")); // TRUE
S.O.P(s.contains("xqy")); // False
```

- It returns boolean value & accepts string as parameter

startsWith()

```
String s = "Mississippi";
S.O.P(s.startsWith("Miss")); // true
```

endsWith()

```
String s = "Mississippi";
S.O.P(s.endsWith("pi")) // true
```

indexOf()

```
String s = "Mississippi";
           0 1 2 3 4 5 6 7 8 9 10
S.O.P(s.indexOf("i")); // (1, 4, 7, 10)
                       ↳ (1)
```

Note:

1) It accepts character as a parameter & it returns the index value, if it not present then it returns "-1"

```
String s = "Mississippi";
           0 1 2 3 4 5 6 7 8 9 10
S.O.P(s.indexOf("issi")) // 1
```

issi → (1, 2, 3, 4)
issi → (4, 5, 6, 7)
↳ (1) starting

1

trim():

```
String s = "  Arvind  ";
s.o.p(s.trim());  // Arvind
```

Note: It removes the extra spaces at the start & end of the string

compareTo()

① s1 a b c
97 98 99

s2 a b c
97 98 99

0 0 0

s.o.p(s1.compareTo(s2)) // 0

② s1 a b c

s2 a b

3 - 2 = 1

s1.compareTo(s2) // 1
s2.compareTo(s1) // -1

"abc"

ASCII Characters

- Alpha Chars
  - Upper (A-Z) 65-90
  - lower (a-z) 97-122
- Numeric (0-9) 48-57
- Special Characters

② s1 a b c
97 98 99

s2 a c x
97 99 120

0 -1

-1

Sorting
abcd
→ dbac
→ dcba

④ s1 a

s2 a b c d

1 - 4 = -3

Note:
(1) It accepts string as parameters & returns the difference b/w them in an integer value

CompareToIgnoreCase()

→ s1 a b c
97

s2 A B C
97

0

s1 A B C
65

s2 a b c
65

It does the same operation as compareTo() only difference is it ignores the case

s1 (==) s2

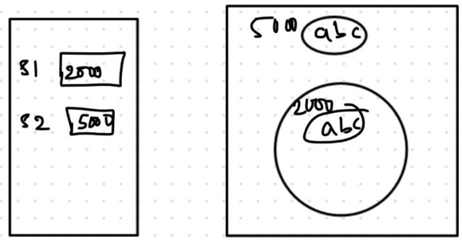
↑ address

2000 == 5000 (F)

→ s1 = "abc"

→ s2 = new String("abc")

s1.equals(s2)



equalsIgnoreCase()

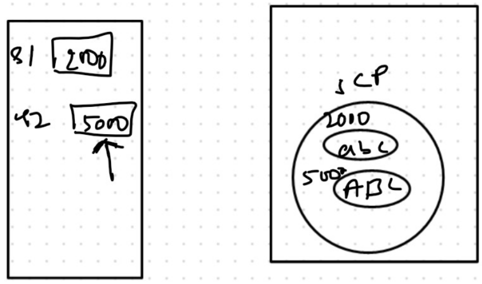
s1 = "abc"

s2 = "ABC"

s1==s2 // F

2000 == 5000 F

s1.equals(s2) // True



s1 == (s2).toLowerCase() //

s.o.p (s1.equalsIgnoreCase(s2)) ← equal //true

s.o.p (s1.equals(s2)) // false

a	b	c
97	98	99
a	b	c
97	98	99
0	0	0

lower →
upper →

-0 +1 -1 not equals

a	b	c
97	98	99
A	B	C
65	66	67
32		

CharAt()

String s1 = "abc";
0 1 2

SOP(s1.charAt(2)) // c

s1.CharAt(3) // String Index Out Of Bounds Exception

note: It accepts integer value as input & returns the character present at the particular index

If that index doesn't exist it throws above Exception

subString()

String s = "Mississippi"
0 1 2 3 4 5 6 7 8 9 10

s.o.p(s.substring(4,8))

// issi

(4,8) ↓ 4 —— < 8 4, 5, 6, 7 exclude

return all the character

(4, 8)
for (0 <end ; +1)
4<(i) i <8
} ③

→ String
→ Array
} Interview for (loop)

toCharArray()

Array

String s = "miss"

ar → | 'm' | 'i' | 's' | 's' |
0 1 2 3

char ar[] = s.toCharArray() → op

char array

split()

String s = "India_is_my_Country"

s → | "India" | "is" | "my" | "Country" |
0 1 2 3

size 4

String ar[] = s.split(" ");
s.o.p(ar[0]) // India
ar[3] // Country

s → "India_is_my_country"

op → India is my country

String s = "" s.split("_")

for ( ){
s+ar[i]
}

replace():

String s = "India_is_my_country"

s = s.replace("_", "_")

s.o.p(s) → India is my country

15

Write a program (WAP) to reverse a string

s.reverse()
"abc"
"cba"

String S = "abc" "abcd"
012 cba
210
cba

210

String r = "";
for (int i = s.length()-1 ; i >=0 ; i --)
r = r+s.charAt(i)

}
s.o.p(r)

"" + 'c' = "c"
"c" + 'b' = "cb"
"cb" + 'a' = "cba"

Scanner()

next() nextLine()

nextInt() ←

sc = new Scanner(System.in);

sc.nextInt() ← 8
sc.nextInt() ← 9

8 9 10

[Hello world]