

# Fun with Strings

# An Interesting Article

“How Revolutionary Tools  
Cracked a 1700s Code”

<http://www.nytimes.com/2011/10/25/science/25code.html>

|oghnamōrλuvzīgriγm|érzueenjyra=rzrþhmhmzý|úλx  
 zufrzrziπhλurhχ|nōmýx|lδa+|+u|súst|λtχ+wrpen|snrþr  
 ππ|nχíδuλh|pýcz|πh|δoλh|rnu|ōph|rλmōzλu|h|lōt|ηno=rzκ  
 hē|zún:scφ|λfπíz|λnzu|vōimzý|πrλu|rz|h|w|λ|f|p|δ|c|p  
 c|p|δ|u|oz|p|n|s|g|p|y|g|ō|p|δ|ē|t|d|c|p|h|δ|ē|l|b|y|+|p|c|p|u|f|x|z|g|t|z|η|p|u|h  
 =r|j|á|b|h|f|:zē|l|s|ú|y|x|u|ez|p|g|:z|c|ú|λ|g|m|u|z|p|r|=+|p|p|ō|z|z|h|g|p|y  
 πrλfz|η|á|g|r|ē|h|á|s|h|r|z|h|z|=|s|z|u|t|λ|δ|j|n|λ|λ|g|z|n|δ|j|η|h|λ|o|e  
 • púoogzió|ú|+c|n|z|x|x|k|mic:|x|y|p|d|+|r|p|h|r|z|λ|s|λ|c  
 • x|n|z|f|s|=h|p|m|η|δ|u|δ|s|á|λ|u|b|+|z|η|p|p|z|g|s|x|δ|=r|λ|ú|j|t|í|z|ō|g|u|b|h|t|c|u  
 • x|n|z|x|í|c|δ|p|λ|u|v|δ|l|ē|n|h|z|ú|z|η|n:|δ|j|me.  
 z|p|r|v|π|z|y|δ|z|í|z|á|λ|b|c|p|λ|f|η|h|+|d|ō|v|z|h|n:|w|p|u|r|x|p|ic:|u  
 c=|g|p|b|c|í|h|j|c|í|y|x|n|+|=|d|f|p|p|o|=r|z|b|g|h|z|ú|+|l|p|ē|n|z|j|ú|+|d|n|u|λ|  
 t|z|ē|g|í|+|t|p|=|m|p|z|z|n|k|r|x|á|b|λ|ō|j|n|λ|c|x|m|z|f|π|u|g|λ|h|=  
 λ|h|p|δ|u|j|p|y|h|δ|λ|a|t|+|p|δ|ē|λ|s|z|á|z|f|p|y|j|η|δ|z|u|z|ō|λ|z|u|p|ō|h|g|z|π|p|  
 λ|u|v|z|n|+|η|λ|f|δ|c:|í|r:|λ|δ|z|z|h|=|u|+|λ|κ|z|u|b|m|ō|r|λ|u|f:  
 • π|p|z|ú|j|c|í|=m|á|λ|ú|n|p|r|x|z|z|b|δ|c:|n|z|η|ē|δ|j|m|s|p|y|p|d|g|η|δ  
 • +h|c|g|h|g|z|í|z|m|p|í|c|s|=g|ē|h|z|u|h|p|ō|j|f|c|p|r|c|í|p|u|w|δ|o|=m|z|h|y|m|l  
 λ|η|p

ερελ  
ηερ

• z|η|á|z|p|n|=l|r|y|p|λ|η|δ|u|c|δ|f|η|p|p|x|p|z|k|=p|u|e|o|h|ú|j|δ|p|δ|u|b|δ|ō|r|η  
 • p|í|z|λ|h|f|í|o|m|π|p|p|r|δ|ō|g|l|=m|z|l|h|g|u|h|r|t|z|p|z|á|z|x|j|ō|h|λ|f|l:|p|g|ē|z|p|r:  
 • z|v|n|δ|ō|r|g|ú|g|u|p|m|h|h|r|u|d|λ|x|p|á|z|g|c|p|c|η|p|u|c|δ|z|x|δ|ú|r:|u  
 • b|+|q|:|ē|c.  
 V|z|á|z|x|p|l:|m|í|h|p|ú|c|á|h|z|ē|λ|g|h|h|d|n|c|z|ō|h:|p|y|r|g|z|g|u|f|δ|p|m:  
 s|á|p|u|f|z|í|r|w|λ|c|=m|π|v|z|λ|α|χ|η|p|λ|h|η|t|+|r|z|h|f|f|n|p|h|z|ē|n|λ|í  
 m|á|c|p|ú|b|y|f|r|p|h|n|h|y|f|ō|z|í|ú|m|δ|a|g|h|z|ú|+|g|=z|δ|r|x|r|δ|s|z|á|j|g|δ|o  
 p|r|δ|c|q|j|ē|λ|b|z|h|k|f|p|r|g|p|h|+|f|á|l|n|=|g|x|g|l|r|u|δ|á|h|ú|y|+|u|p|r|a|u|f|λ|=λ|y  
 ō|r|u|a|m|ō|h|z|=p|p|u|g|m|n|h|z|u|b|δ|á|j|p|g|z|f|p|λ|n|η|ē|x|ú|c|λ|=n|z|p|δ|h|z  
 p|λ|h|η|t|+|r|z|η|á|x|p|r|c|p|x|p|y|p|u|ē|m|g|=g|p|v|r|z|í|z|g|s|=b|ō|h|z|p|r|g|h|p  
 z|p|λ|ē|η|f|l:|z|ú|f|ō|p|l|n|=|s|x|s|y|p|g|u|í|m|z|m|t|λ|p|h|t|+|d|n|á|p|g|h|c  
 c|u|d|n|r|m|ú|u|z|u|c|x|c|p|δ|í|x|r|p|z|z|ō|j|f|z|z|η|δ|y|ē|j|u|z|á|λ|n|p|r|t|c|y|p  
 λ|n|z|r|+|η|λ|g|z|z|c|Δ|m.  
 P|m|á|o|o|λ|í|z|m|λ|η|=c|b  
 δ|ē|r:|s|y|p|u|d|r|ú|n|p|v|λ.  
 H|ō|í|g|g|λ|b|m|z|z|f|m|η|p|λ|c|δ|m|ō|h|c|p|h|z|=j|p|c|ú|r|δ|p|r:|λ|o  
 z|h|p|r|z|g|=n|p|z|ē|δ|t|y|f|j|á|b|l|s|λ|í|δ|m|c|p|u|g|x|r|z|t|r|ē|p|u|c|δ|n:|á|í|ō|g

A **string** is a sequence of characters.

H e l l o !

H	e	l	l	o	!
---	---	---	---	---	---

0

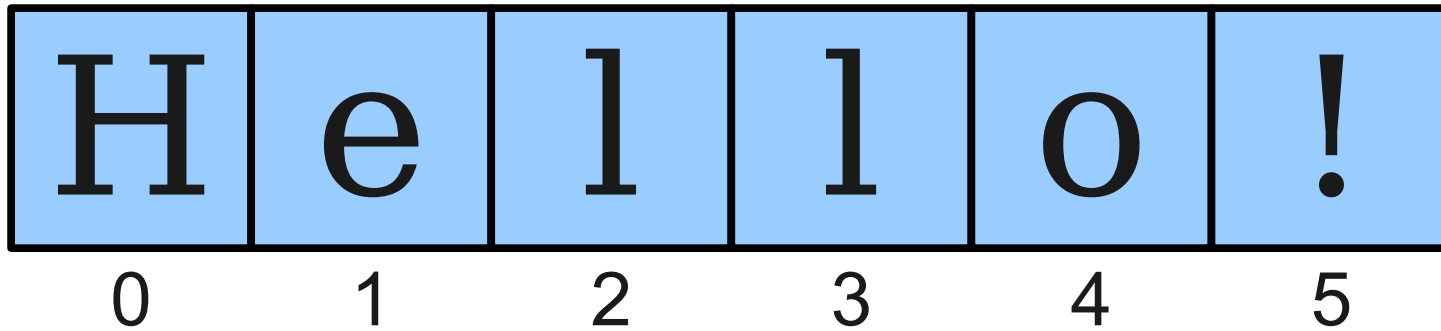
1

2

3

4

5



***string***.charAt (***index***)

# Highlights from Character

**static boolean isDigit(char ch)**

Determines if the specified character is a digit.

**static boolean isLetter(char ch)**

Determines if the specified character is a letter.

**static boolean isLetterOrDigit(char ch)**

Determines if the specified character is a letter or a digit.

**static boolean isLowerCase(char ch)**

Determines if the specified character is a lowercase letter.

**static boolean isUpperCase(char ch)**

Determines if the specified character is an uppercase letter.

**static boolean isWhitespace(char ch)**

Determines if the specified character is **whitespace** (spaces and tabs).

**static char toLowerCase(char ch)**

Converts **ch** to its lowercase equivalent, if any. If not, **ch** is returned unchanged.

**static char toUpperCase(char ch)**

Converts **ch** to its uppercase equivalent, if any. If not, **ch** is returned unchanged.



# Appropriate Poetry

I met a traveller from an antique land  
Who said: Two vast and trunkless legs of stone  
Stand in the desert...Near them, on the sand,  
Half sunk, a shattered visage lies, whose frown,  
And wrinkled lip, and sneer of cold command,  
Tell that its sculptor well those passions read  
Which yet survive, stamped on these lifeless things,  
The hand that mocked them, and the heart that fed:  
And on the pedestal these words appear:  
'My name is Ozymandias, king of kings:  
Look on my works, ye Mighty, and despair!'  
Nothing beside remains. Round the decay  
Of that colossal wreck, boundless and bare  
The lone and level sands stretch far away.

- "Ozymandias," Percy Shelley

# Appropriate Poetry

I met a traveller from an antique land  
Who said: Two vast and trunkless legs of stone  
Stand in the desert...Near them, on the sand,  
Half sunk, a shattered visage lies, whose frown,  
And wrinkled lip, and sneer of cold command,  
Tell that its sculptor well those passions read  
Which yet survive, stamped on these lifeless things,  
The hand that mocked them, and the heart that fed:

**And on the pedestal these words appear:**

**'My name is Ozymandias, king of kings:**

**Look on my works, ye Mighty, and despair!'**

**Nothing beside remains. Round the decay**

**Of that colossal wreck, boundless and bare**

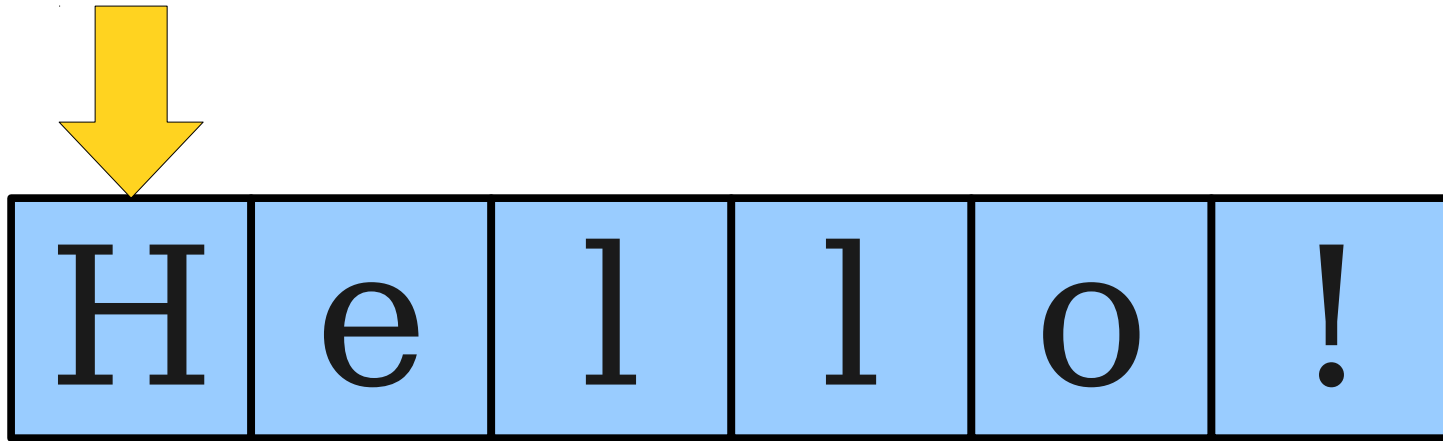
**The lone and level sands stretch far away.**

- "Ozymandias," Percy Shelley

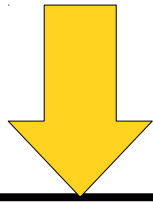
# Strings are Immutable

- Java strings are **immutable**: once a string has been created, its contents cannot change.
- To change a string:
  - Create a new string holding the new value you want it to have.
  - Reassign the **String** variable to hold the new value.

# Reversing a String



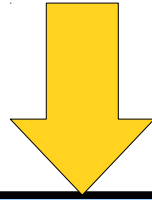
# Reversing a String



H	e	l	l	o	!
---	---	---	---	---	---

H
---

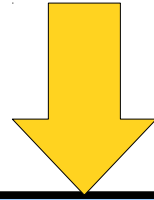
# Reversing a String



H	e	l	l	o	!
---	---	---	---	---	---

e	H
---	---

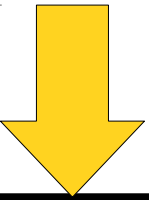
# Reversing a String



H	e	l	l	o	!
---	---	---	---	---	---

l	e	H
---	---	---

# Reversing a String



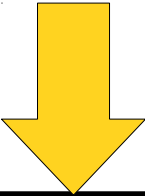
Character array: H e l l o !

Character array: l l e H



# Reversing a String

H e l l o !



o l l e H

# Reversing a String

H	e	l	l	o	!
---	---	---	---	---	---

!	o	l	l	e	H
---	---	---	---	---	---

# Palindromes

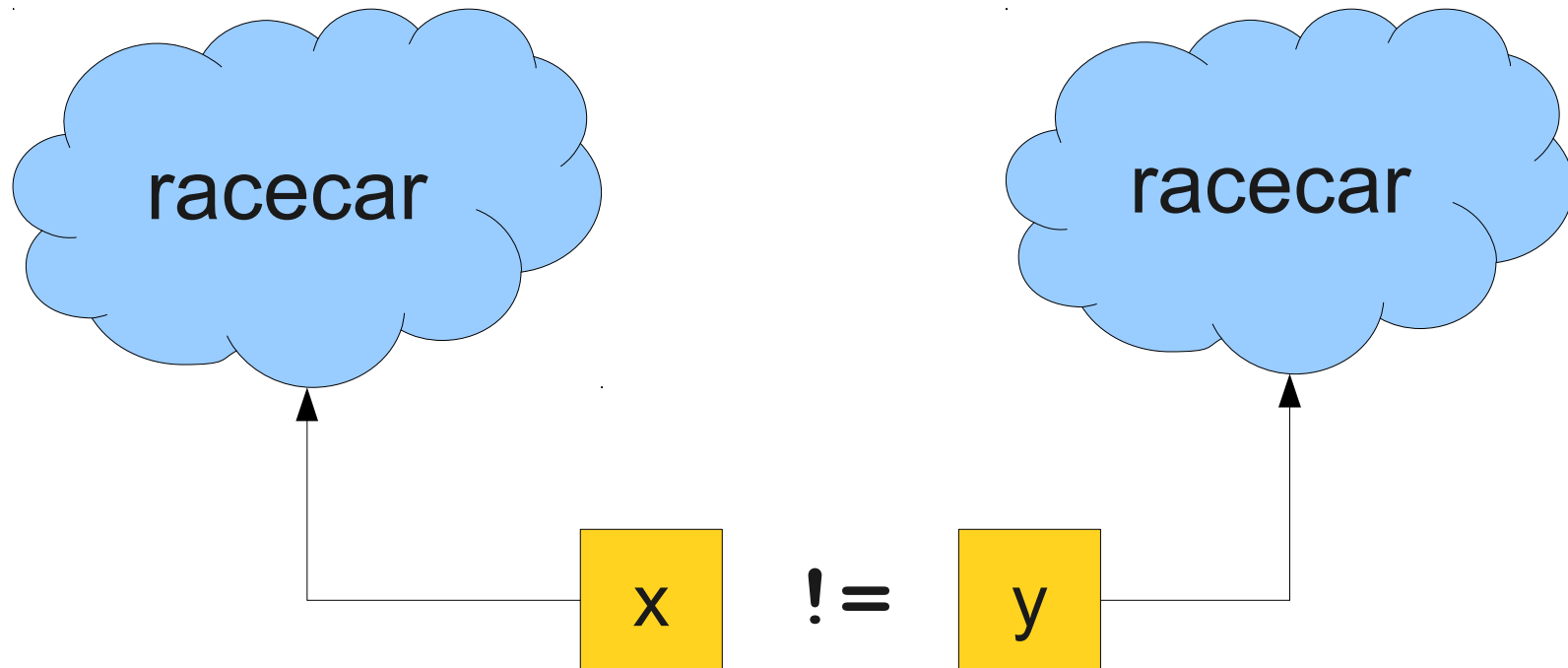
- A **palindrome** is a string that reads the same forwards and backwards.
- For example:
  - Racecar
  - Kayak
  - Mr. Owl ate my metal worm.
  - Go hang a salami! I'm a lasagna hog.

# Checking for Palindromes

What Went Wrong?

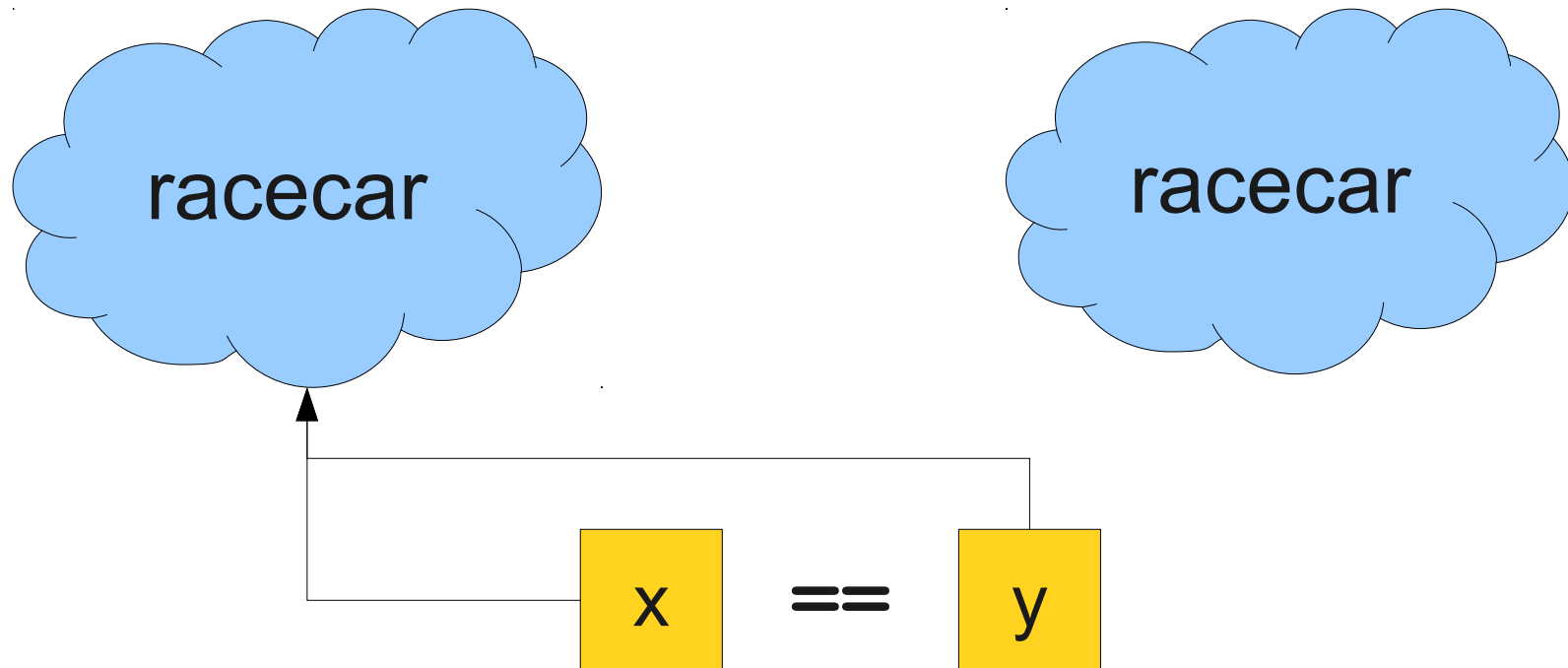
# The == Operator

- When applied to objects, the == operator reports whether the two objects are the same object, not whether the *values* of those objects are equal.



# The == Operator

- When applied to objects, the == operator reports whether the two objects are the same object, not whether the *values* of those objects are equal.



# Comparing Strings for Equality

- To determine if two strings are equal, use the `.equals()` method:

```
String s1 = "racecar";  
String s2 = reverseString(s1);  
if (s1.equals(s2)) {  
    /* ... s1 and s2 are equal ... */  
}
```



A man, a plan, a caret, a ban, a myriad, a sum, a lac, a liar, a hoop, a pint, a catalpa, a gas, an oil, a bird, a yell, a vat, a caw, a pax, a wag, a tax, a nay, a ram, a cap, a yam, a gay, a tsar, a wall, a car, a luger, a ward, a bin, a woman, a vassal, a wolf, a tuna, a nit, a pall, a fret, a watt, a bay, a daub, a tan, a cab, a datum, a gall, a hat, a tag, a zap, a say, a jaw, a lay, a wet, a gallop, a tug, a trot, a trap, a tram, a torr, a caper, a top, a tonk, a toll, a ball, a fair, a sax, a minim, a tenor, a bass, a passer, a capital, a rut, an amen, a ted, a cabal, a tang, a sun, an ass, a maw, a sag, a jam, a dam, a sub, a salt, an axon, a sail, an ad, a wadi, a radian, a room, a rood, a rip, a tad, a pariah, a revel, a reel, a reed, a pool, a plug, a pin, a peek, a parabola, a dog, a pat, a cud, a nu, a fan, a pal, a rum, a nod, an eta, a lag, an eel, a batik, a mug, a mot, a nap, a maxim, a mood, a leek, a grub, a gob, a gel, a drab, a citadel, a total, a cedar, a tap, a gag, a rat, a manor, a bar, a gal, a cola, a pap, a yaw, a tab, a raj, a gab, a nag, a pagan, a bag, a jar, a bat, a way, a papa, a local, a gar, a baron, a mat, a rag, a gap, a tar, a decal, a tot, a led, a tic, a bard, a leg, a bog, a burg, a keel, a doom, a mix, a map, an atom, a gum, a kit, a baleen, a gala, a ten, a don, a mural, a pan, a faun, a ducat, a pagoda, a lob, a rap, a keep, a nip, a gulp, a loop, a deer, a leer, a lever, a hair, a pad, a tapir, a door, a moor, an aid, a raid, a wad, an alias, an ox, an atlas, a bus, a madam, a jag, a saw, a mass, an anus, a gnat, a lab, a cadet, an em, a natural, a tip, a caress, a pass, a baronet, a minimax, a sari, a fall, a ballot, a knot, a pot, a rep, a carrot, a mart, a part, a tort, a gut, a poll, a gateway, a law, a jay, a sap, a zag, a tat, a hall, a gamut, a dab, a can, a tabu, a day, a batt, a waterfall, a patina, a nut, a flow, a lass, a van, a mow, a nib, a draw, a regular, a call, a war, a stay, a gam, a yap, a cam, a ray, an ax, a tag, a wax, a paw, a cat, a valley, a drib, a lion, a saga, a plat, a catnip, a pooh, a rail, a calamus, a dairyman, a bater, a canal - Panama!

# Searching a String

- You can search a string for a particular character or string by using the **indexOf** method:

***string*.indexOf(*pattern*)**

- **indexOf** returns the index of the first match if one exists.
- Otherwise, it returns -1 as a sentinel.

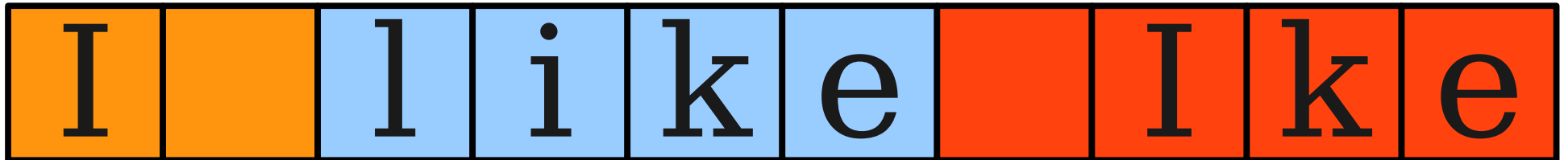
# Replacing a Substring

- Because strings are immutable, it can be difficult to replace pieces of a string.
- To replace a segment of a string:
  - Obtain the **substring** of the string up to the point to replace.
  - Obtain the substring of the string after the point to replace.
  - Glue the two pieces back together with the replacement in the middle.

I		l	i	k	e		I	k	e
---	--	---	---	---	---	--	---	---	---

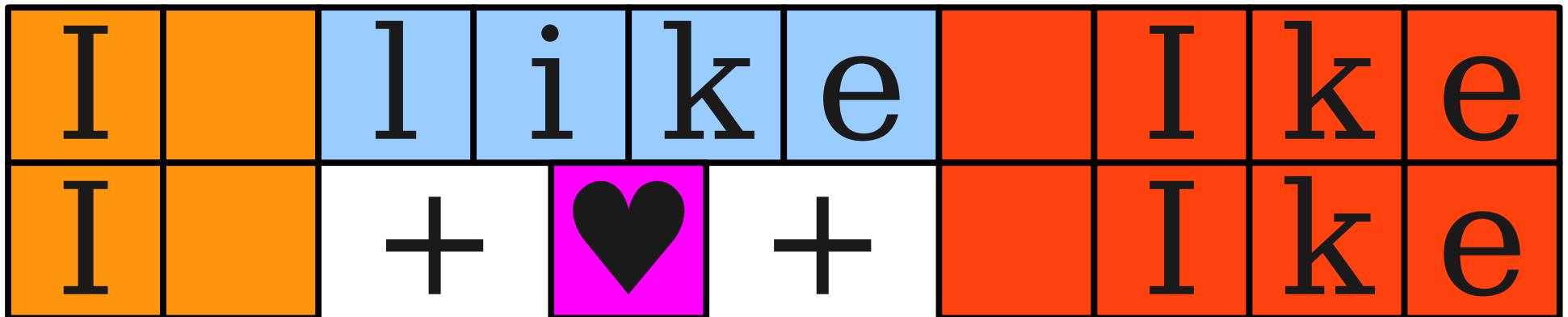
# Replacing a Substring

- Because strings are immutable, it can be difficult to replace pieces of a string.
- To replace a segment of a string:
  - Obtain the **substring** of the string up to the point to replace.
  - Obtain the substring of the string after the point to replace.
  - Glue the two pieces back together with the replacement in the middle.



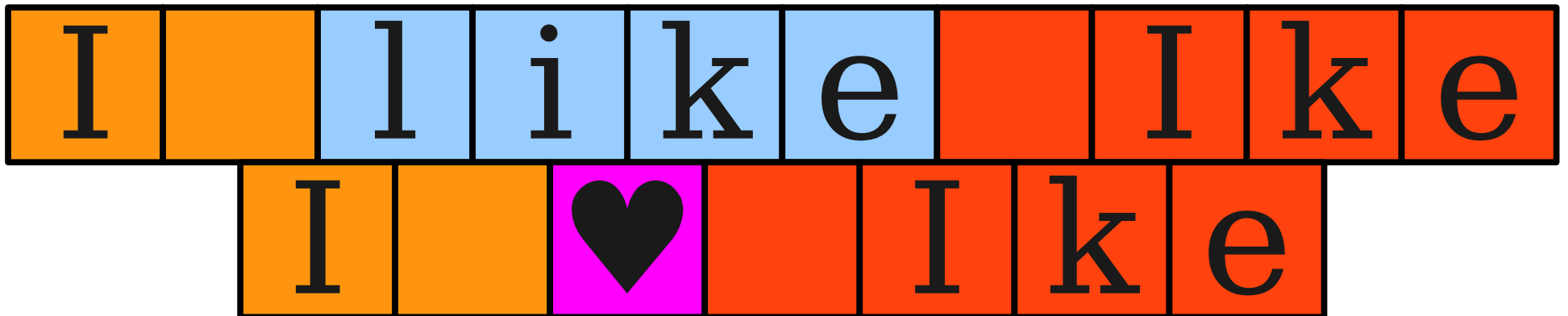
# Replacing a Substring

- Because strings are immutable, it can be difficult to replace pieces of a string.
- To replace a segment of a string:
  - Obtain the **substring** of the string up to the point to replace.
  - Obtain the substring of the string after the point to replace.
  - Glue the two pieces back together with the replacement in the middle.



# Replacing a Substring

- Because strings are immutable, it can be difficult to replace pieces of a string.
- To replace a segment of a string:
  - Obtain the **substring** of the string up to the point to replace.
  - Obtain the substring of the string after the point to replace.
  - Glue the two pieces back together with the replacement in the middle.





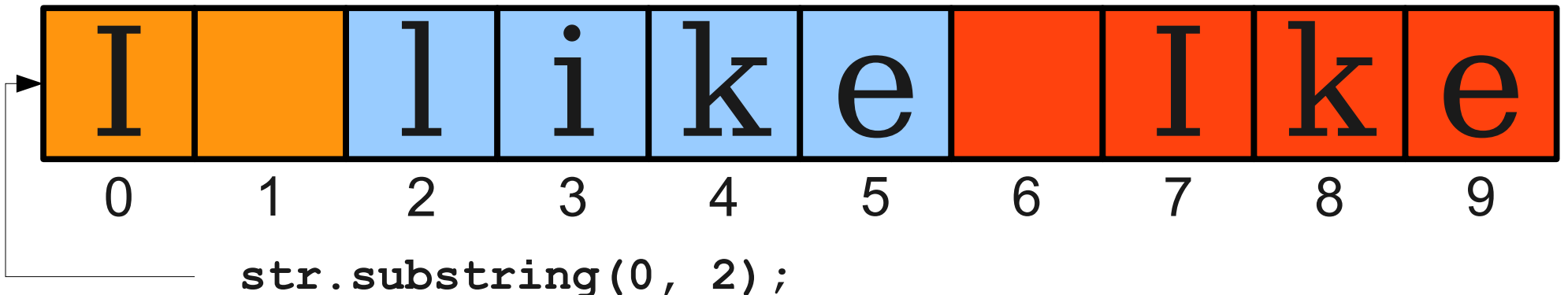
# Obtaining Substrings

- To get all of the characters in the range [start, stop), use

***string***.substring(***start***, ***stop***)

- To get all of the characters from some specified point forward, use

***string***.substring(***start***)





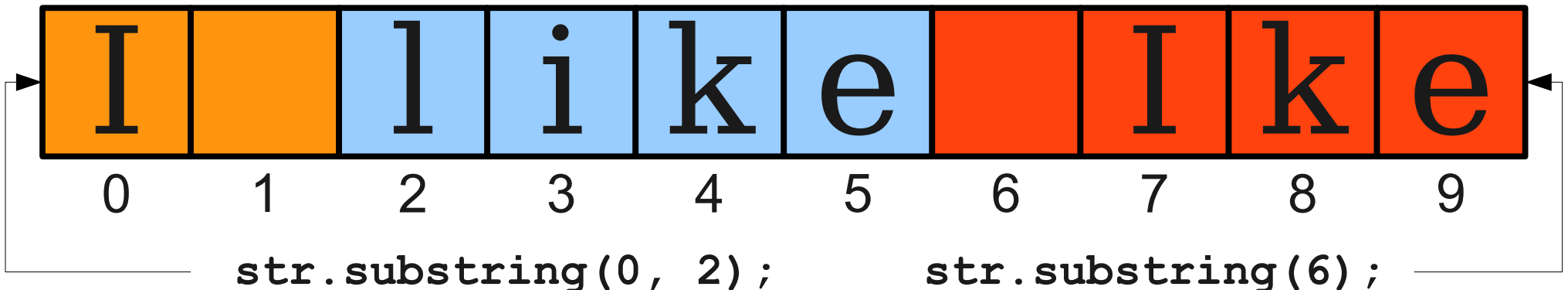
# Obtaining Substrings

- To get all of the characters in the range [start, stop), use

***string*.substring(*start*, *stop*)**

- To get all of the characters from some specified point forward, use

***string*.substring(*start*)**



# Useful `String` Methods

**`int length()`**

Returns the length of the string

**`char charAt(int index)`**

Returns the character at the specified index. Note: Strings indexed starting at 0.

**`String substring(int p1, int p2)`**

Returns the substring beginning at **`p1`** and extending up to but not including **`p2`**

**`String substring(int p1)`**

Returns substring beginning at **`p1`** and extending through end of string.

**`boolean equals(String s2)`**

Returns true if string **`s2`** is equal to the receiver string. This is case sensitive.

**`int compareTo(String s2)`**

Returns integer whose sign indicates how strings compare in lexicographic order

**`int indexOf(char ch) or int indexOf(String s)`**

Returns index of first occurrence of the character or the string, or -1 if not found

**`String toLowerCase() or String toUpperCase()`**

Returns a lowercase or uppercase version of the receiver string