

Strings

Assignment 4

- Assignment 4 due on Monday at 3:15PM.
- ***Recommendation:*** Try to get the game completed by Friday so you have time to test and add extensions by Monday.
- Have questions?
 - Stop by the LaIR!
 - Stop by office hours!
 - Email your section leader!

Midterm Information

- Exam is next Tuesday from 7PM - 10PM, location TBA.
- Closed-book, closed-computer, limited notes.
 - You can have one 8.5" × 11" double-sided note sheet with you.
 - We will provide a Java and Karel reference sheet at the exam. You can see it online at the course website, and we'll distribute copies at tonight's practice exam as well
- Exam covers material up through and including today's lecture.
- Location TBA.
- Practice exam is tonight from 7PM - 10PM in Cemex Auditorium.
 - Can't make it? No worries! We'll post the exam and solutions online.

Recap from Last Time

A *string* is a sequence of characters.

H e l l o !

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

0

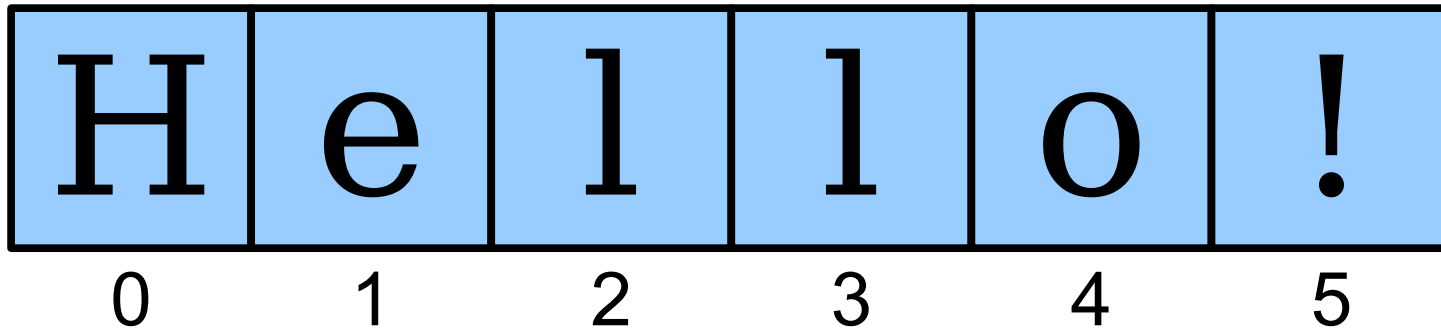
1

2

3

4

5

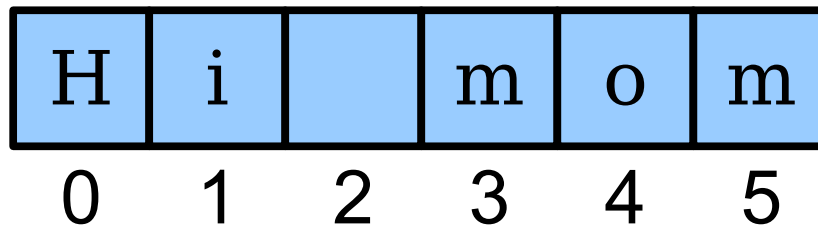


string.charAt(*index*)

Looping Over Characters

- Because each character in a string occupies a particular index, you can visit all the characters in a string one at a time using a loop.
- Canonical “loop over the characters in a string” loop:

```
for (int i = 0; i < string.length(); i++) {  
    char ch = string.charAt(i);  
    /* ... process ch ... */  
}
```
- The **string**.length() method returns the number of characters in the string. This is one larger than the last valid index in the string.



(length 6)

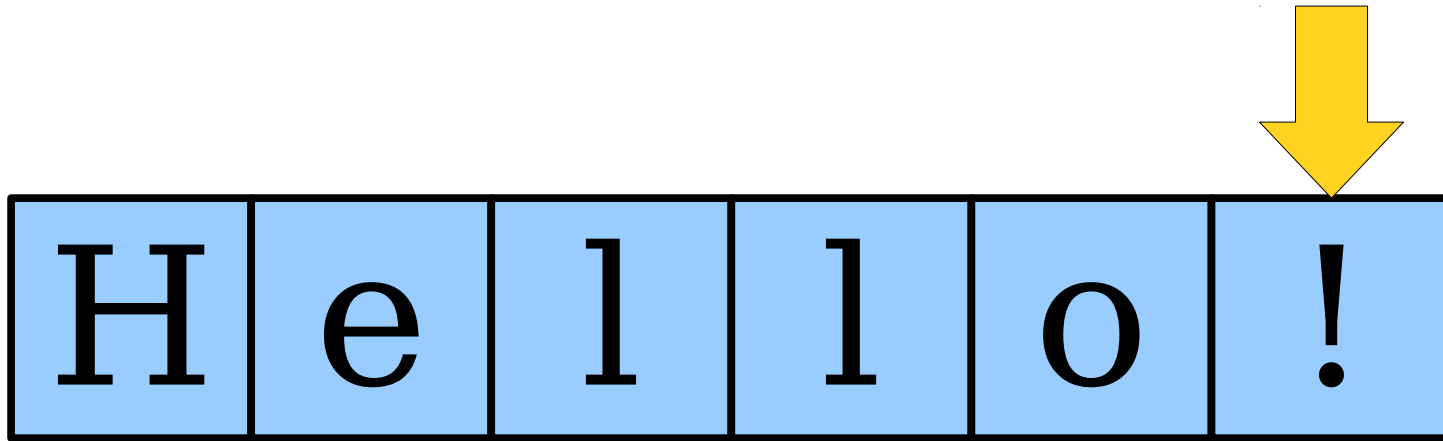
Strings are Immutable

- 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.
- ***Important consequence***: if you pass a String into a method, that method cannot modify that string.

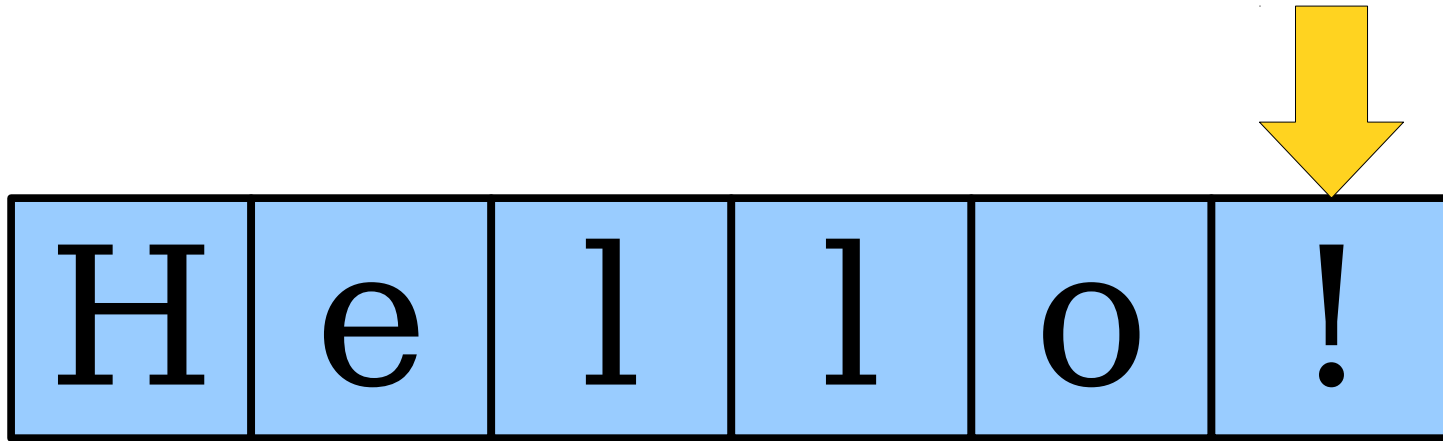


http://deathandtaxesmag.wpengine.netdna-cdn.com/wp-content/uploads/2013/01/quokka_1.jpg

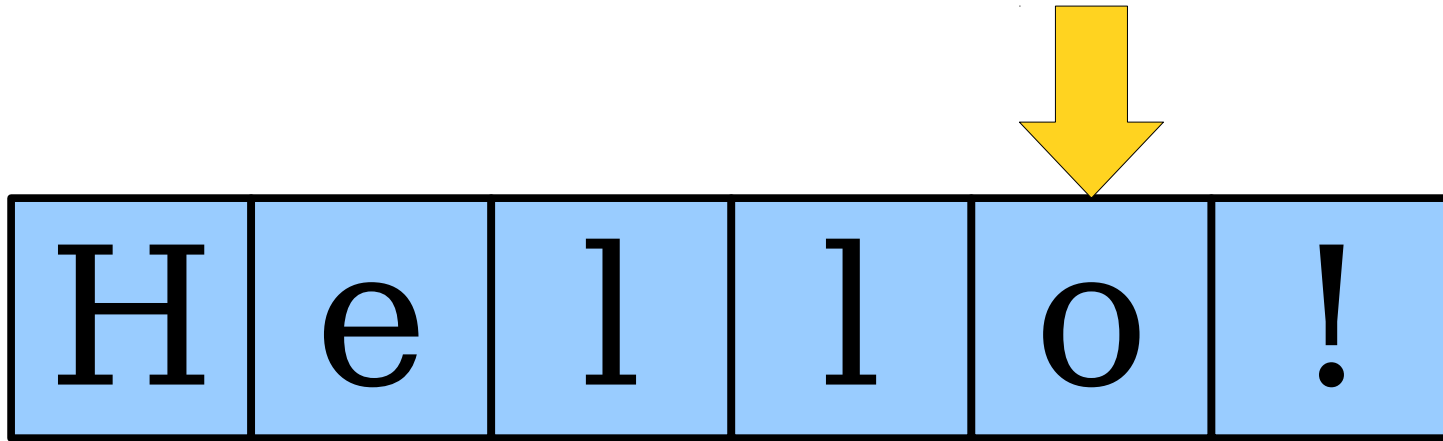
Reversing a String



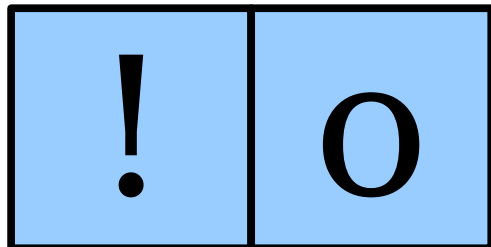
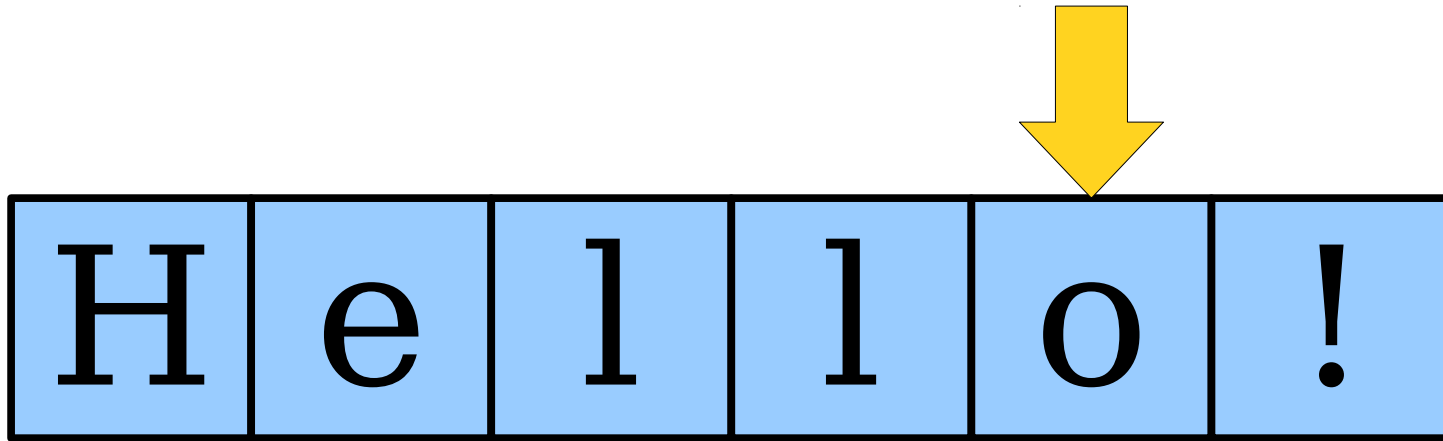
Reversing a String



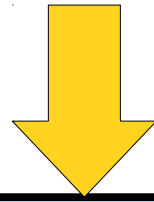
Reversing a String



Reversing a String



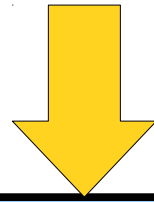
Reversing a String



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

!	o
---	---

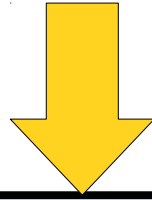
Reversing a String



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

!	o	l
---	---	---

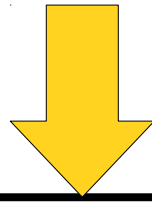
Reversing a String



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

!	o	l
---	---	---

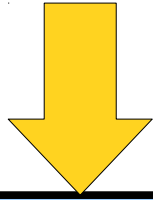
Reversing a String



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

!	o	l	l
---	---	---	---

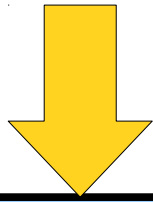
Reversing a String



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

!	o	l	l
---	---	---	---

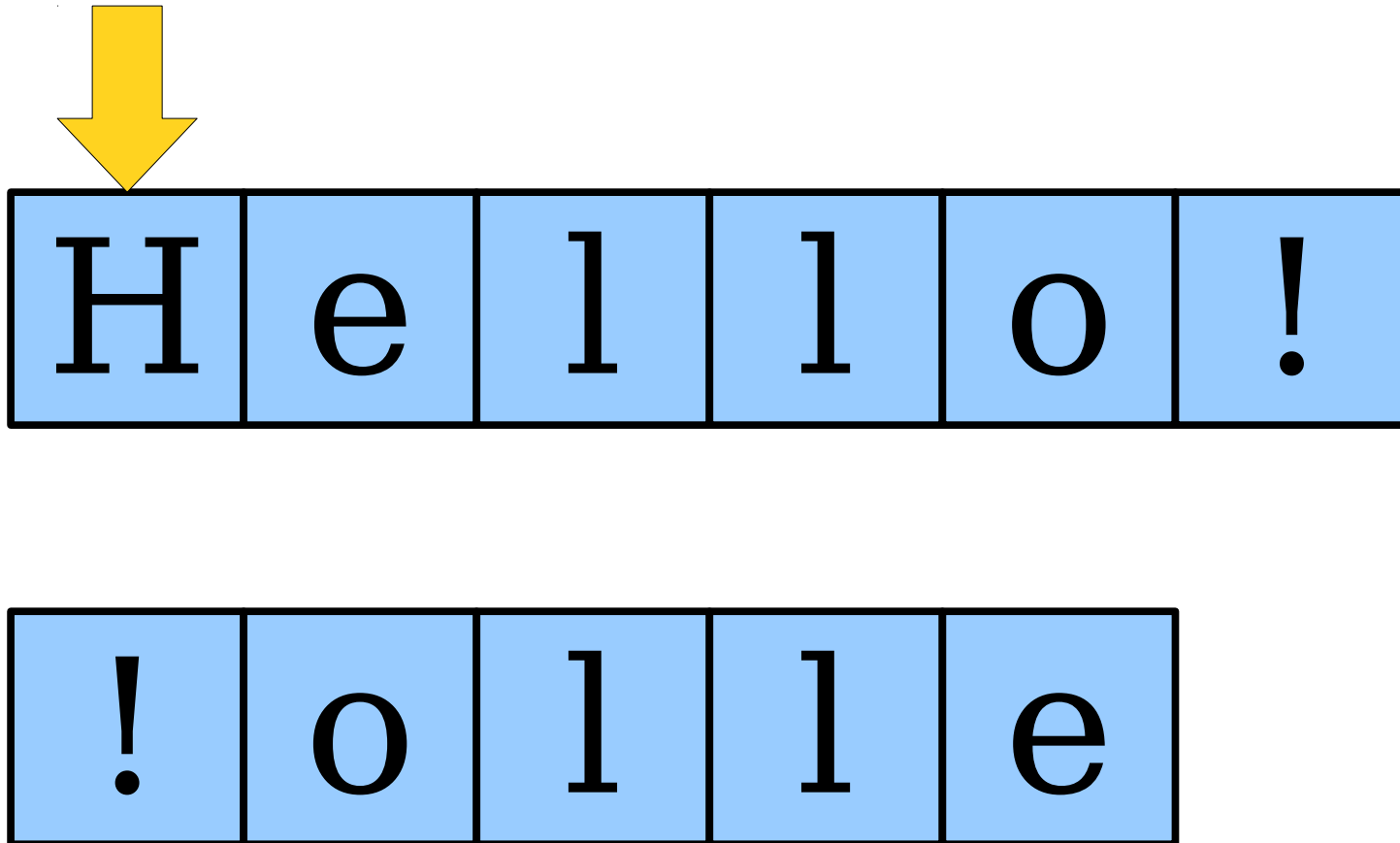
Reversing a String



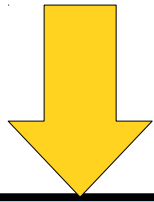
H	e	l	l	o	!
---	---	---	---	---	---

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

Reversing a String



Reversing a String



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

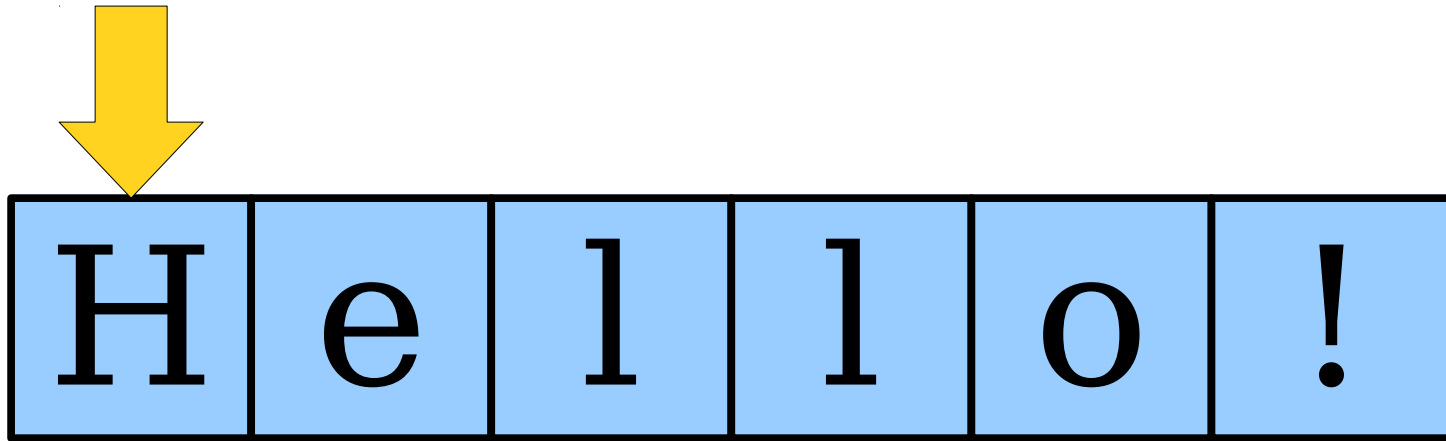
!	o	l	l	e	H
---	---	---	---	---	---

Reversing a String

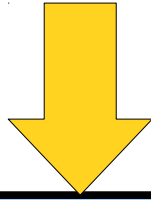
H	e	l	l	o	!
---	---	---	---	---	---

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

Reversing a String



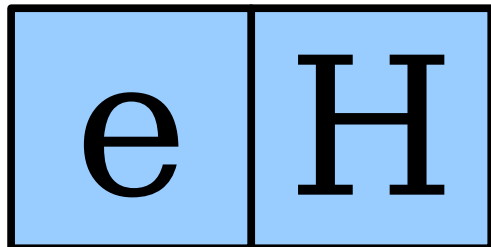
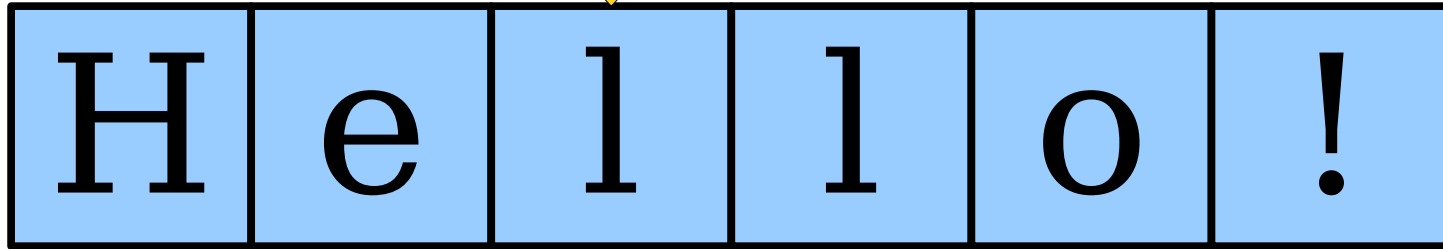
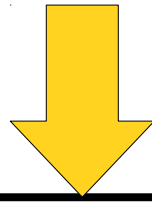
Reversing a String



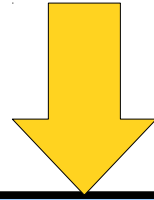
H	e	l	l	o	!
---	---	---	---	---	---

H

Reversing a String



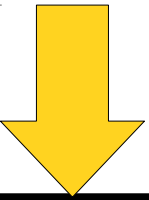
Reversing a String



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

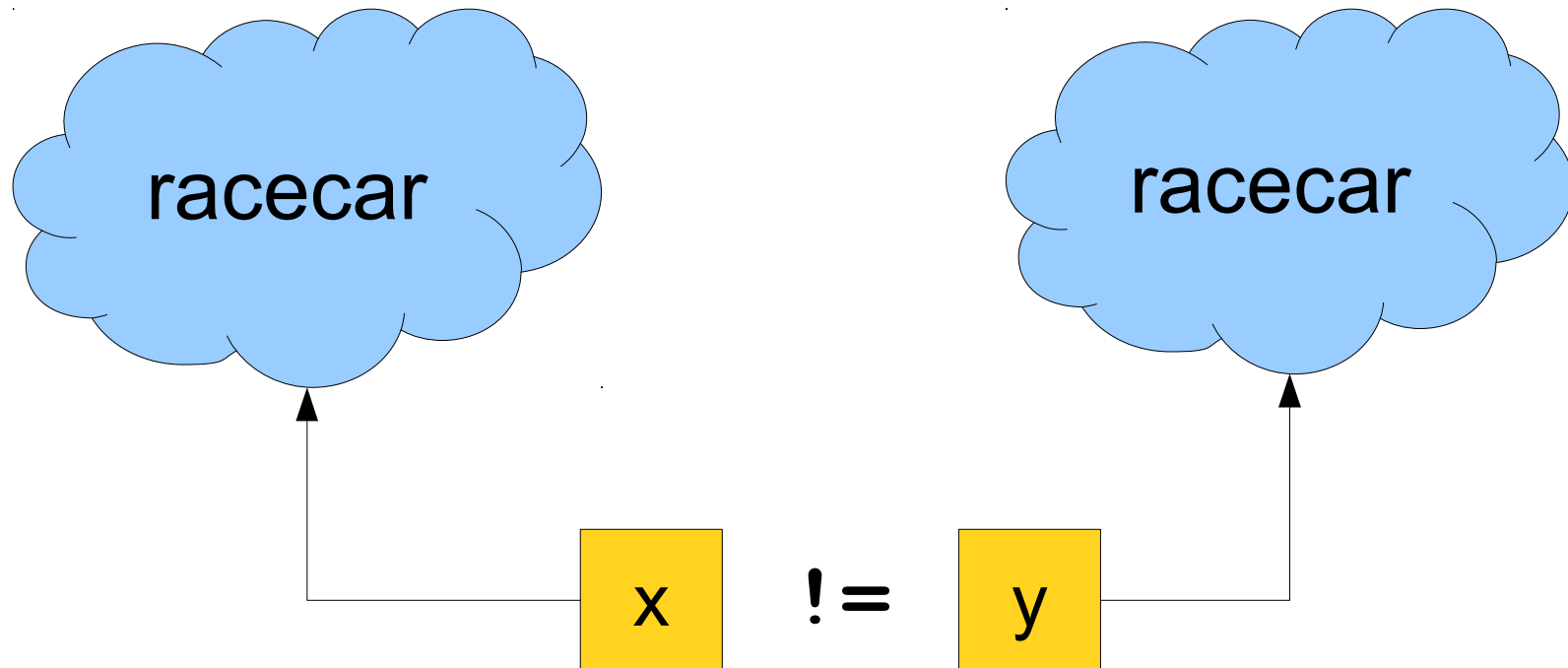
- A ***palindrome*** is a string that reads the same forwards and backwards.
- For example:
 - Racecar
 - Kayak
 - Mr. Owl ate my metal worm.
 - Go hang a salami! I'm a lasagna hog.
- My question to you: do other languages have the concept of a palindrome?

Checking for Palindromes

What Went Wrong?

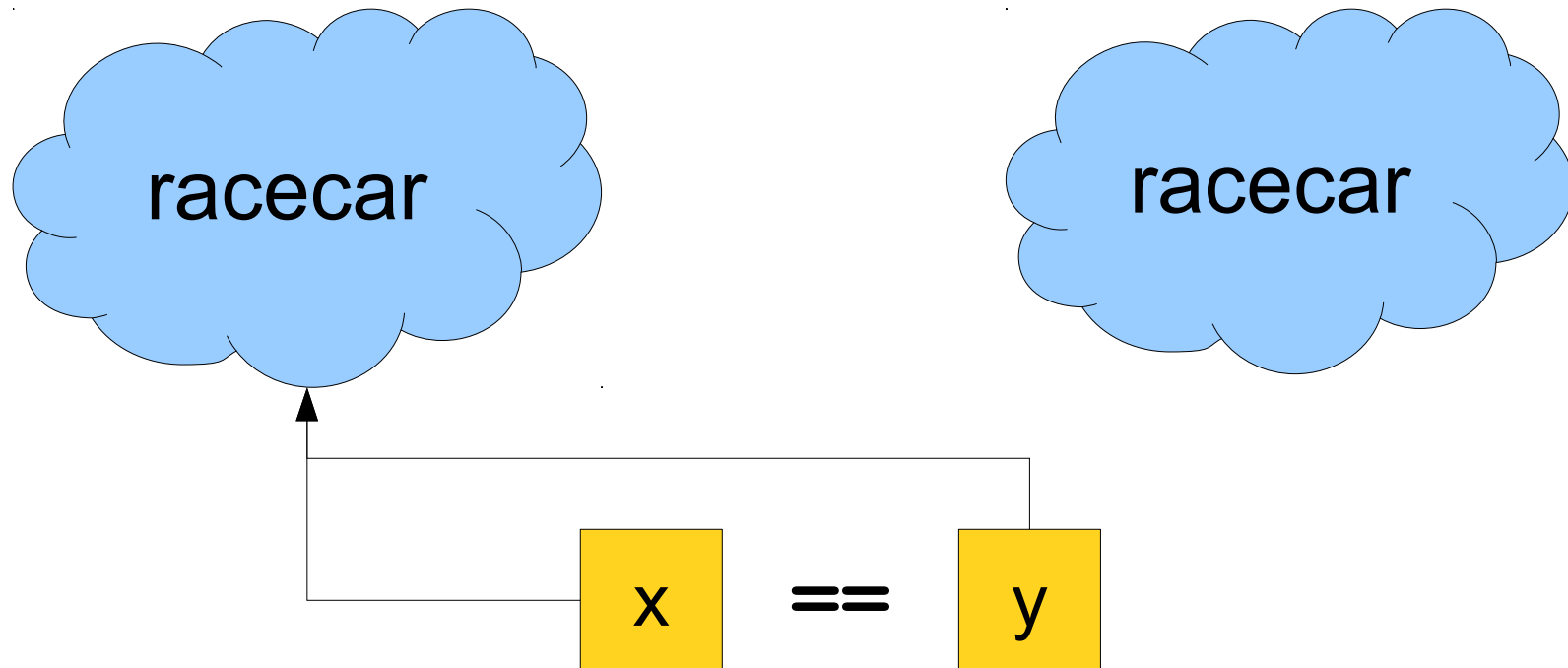
The == Operator

- 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 ... */  
}
```

Some Test Cases

- Let's test our program on some examples:
 - Racecar
 - Kayak
 - Mr. Owl ate my metal worm.
 - Go hang a salami! I'm a lasagna hog.
- Will it work?

Testing Properties of Characters

boolean Character.isDigit(char ch)

Determines if the specified character is a digit.

boolean Character.isLetter(char ch)

Determines if the specified character is a letter.

boolean Character.isLetterOrDigit(char ch)

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

boolean Character.isLowerCase(char ch)

Determines if the specified character is a lowercase letter.

boolean Character.isUpperCase(char ch)

Determines if the specified character is an uppercase letter.

boolean Character.isWhitespace(char ch)

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

char Character.toLowerCase(char ch)

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

char Character.toUpperCase(char ch)

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

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!