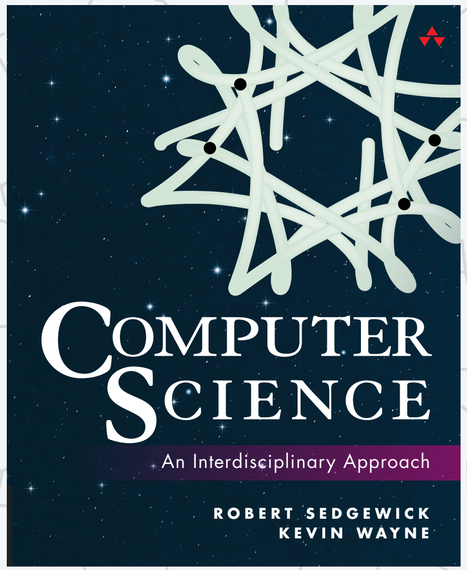
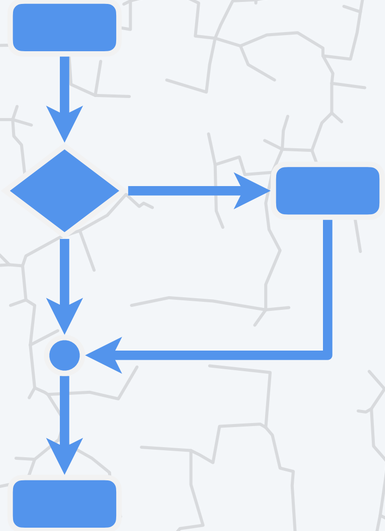
Robert Sedgewick | Kevin Wayne



https://introcs.cs.princeton.edu

1.3 Conditionals

- *booleans*
- *if statements*
- *if–else statements*
- *nested conditionals*
- *case study: text-to-speech (years)*



Basic building blocks of programs

any program you might want to write

objects

functions | libraries

graphics, sound, and image I/O

arrays

conditionals | loops

Math | text I/O

built-in data types | assignment statements

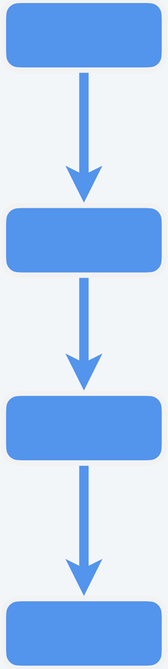
this lecture: decision making

last lecture: equivalent to a calculator

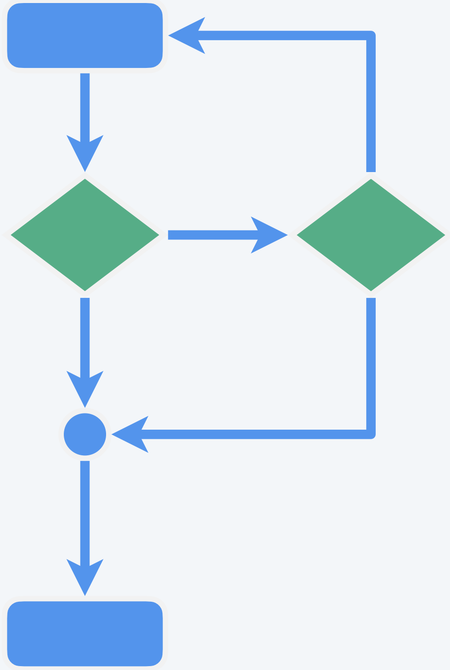
Control flow with conditionals and loops

Control flow. The sequence of statements that are actually executed in a program.

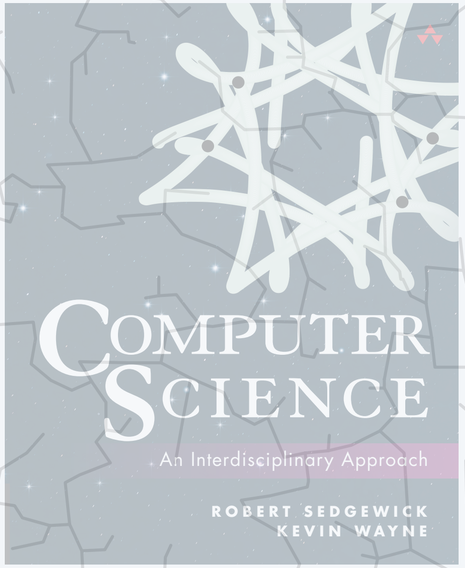
Conditionals and loops. Enable us to choreograph the program's control flow.



straight-line control flow
(last lecture)



control flow with conditionals and loops
(this week)



https://introcs.cs.princeton.edu

1.3 CONDITIONALS

- *booleans*
- *if statements*
- *if–else statements*
- *nested conditionals*
- *case study: text-to-speech (years)*

Review: Java's core language types

A data type (or type) is a set of values together with operations on those values.

type	set of values	example values	typical operations
`String`	*sequences of characters*	`"Hello, World"` `"I ❤️ COS 126!"`	concatenate
`int`	*integers*	`17` `-12345`	add, subtract, multiply, divide, compare, equality
`double`	*floating-point numbers*	`2.5` `-0.125`	add, subtract, multiply, divide, compare, equality
`boolean`	*truth values*	`true` `false`	and, or, not, equality ← *this lecture*
`char`	*characters*	`'A'` `'中'`	compare

The *boolean* data type

Typical usage: expressing logic; making decisions; controlling program flow. ⟵ *foundation for conditionals and loops*

values	*true and false*		
literals	`true` `false`		
logical operations	*not*	*and*	*or*
logical operators	`!`	`&&`	`||`

Truth table. Shows every possible input and its resulting output.

expression	value
`!false`	`true`
`!true`	`false`

truth table for NOT

expression	value
`false && false`	`false`
`false && true`	`false`
`true && false`	`false`
`true && true`	`true`

truth table for AND

expression	value
`false || false`	`false`
`false || true`	`true`
`true || false`	`true`
`true || true`	`true`

truth table for OR

Boolean meme

Comparison operators in Java

Comparison operators. Compare two numeric values.

- Operands: two numeric expressions. ← *can be literals, variables, or compound expressions*
- Evaluates to: a value of type `boolean`.

operator	meaning	true	false
==	*equal*	`2 == 2`	`2 == 3`
!=	*not equal*	`3 != 2`	`2 != 2`
<	*less than*	`2 < 13`	`13 < 2`
<=	*less than or equal*	`2 <= 2`	`3 <= 2`
>	*greater than*	`13 > 2`	`2 > 13`
>=	*greater than or equal*	`2 >= 2`	`2 >= 3`

comparison operators in Java

Examples of comparison operators in practice

zero denominator?	`denominator == 0`
non-negative discriminant?	`(b*b - 4.0*a*c) >= 0.0`
divisible by 60?	`(minutes % 60) == 0`
RGB color is not black?	`(red > 0) || (green > 0) || (blue > 0)`
valid month?	`(month >= 1) && (month <= 12)`
invalid month?	`!((month >= 1) && (month <= 12))`
floating-point roundoff error	`(0.1 * 3.0) == 0.3`
string equality?	`args[0] == "Hello"`

parentheses for clarity: arithmetic operators have higher precedence than comparison operators

compound boolean expressions

be careful when using == with floating-point numbers! (evaluates to `false`)

don't use == to compares strings! (evaluates to `false`)

Example of computing with booleans: leap-year test

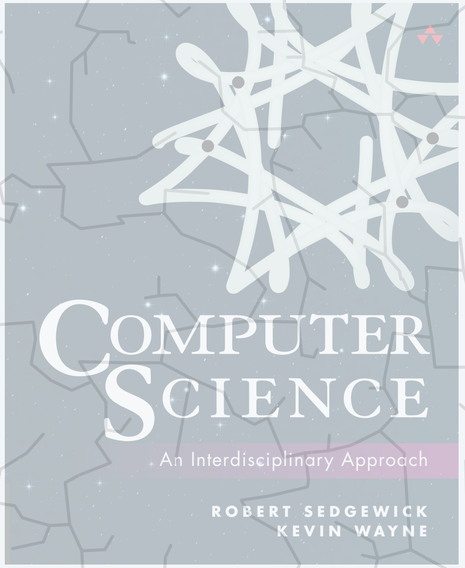
Q. Is a given year a leap year? ← *Gregorian calendar*

A. A year is a leap year if divisible by 400, or divisible by 4 but not 100.

```
public class LeapYear {
   public static void main(String[] args) {

      int year = Integer.parseInt(args[0]);
      boolean isLeapYear;

      // divisible by 4 but not 100
      isLeapYear = (year % 4 == 0) && (year % 100 != 0);

      // or divisible by 400
      isLeapYear = isLeapYear || (year % 400 == 0);

      System.out.println(isLeapYear);

   }
}
```

when you print a boolean, Java prints either `true` *or* `false`

```
~/cos126/datatypes> java LeapYear 2024
true

~/cos126/datatypes> java LeapYear 2023
false

~/cos126/datatypes> java LeapYear 1900
false

~/cos126/datatypes> java LeapYear 2000
true
```



https://introcs.cs.princeton.edu

1.3 CONDITIONALS

- *booleans*
- *if statements*
- *if–else statements*
- *nested conditionals*
- *case study: text-to-speech (years)*

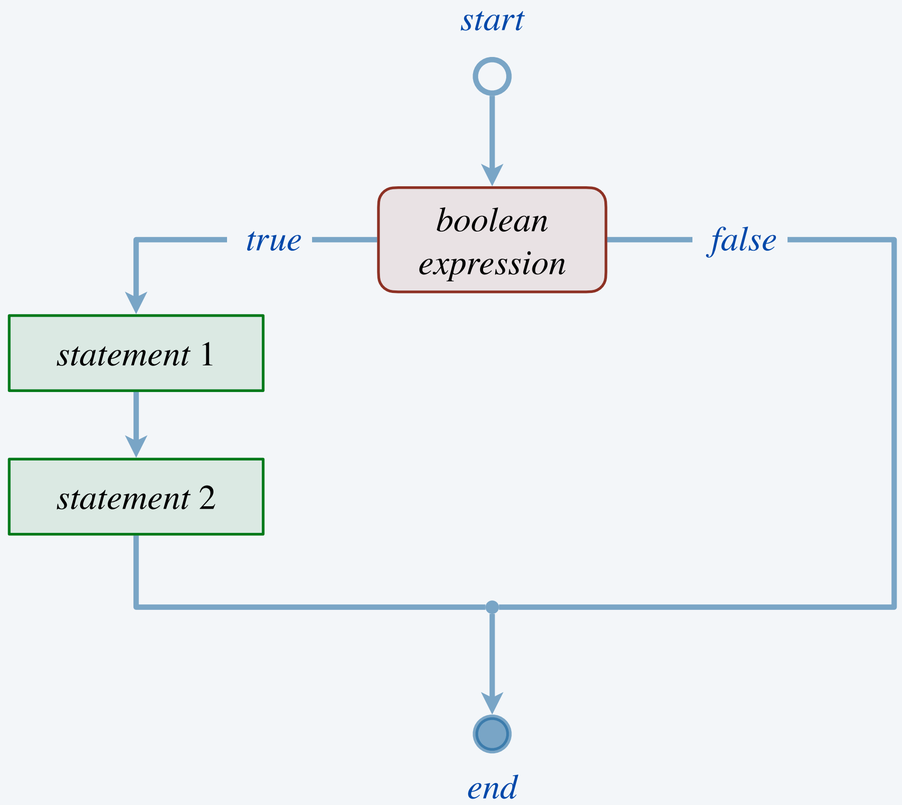
The *if* statement

Execute certain code only when a specified condition is true.

- Evaluate a boolean expression (the condition).
- If `true`, execute the statements in the code block (delimited by curly braces).
- If `false`, skip the block and continue with the next statement.

```
if (<boolean expression>) {
   <statement 1>
   <statement 2>
}
```

if statement template



if statement flowchart

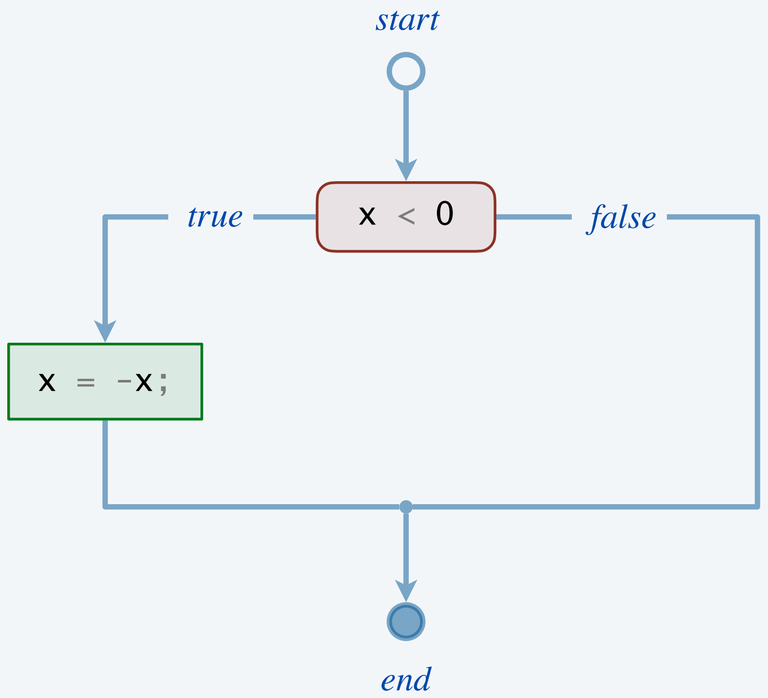
The *if* statement

Execute certain code only when a specified condition is true.

- Evaluate a boolean expression (the condition).
- If true, execute the statements inside the code block (delimited by curly braces).
- If false, skip the block and continue with the next statement.

```
if (x < 0) {
   x = -x;
}
```

replaces x with its absolute value



if statement flowchart

Code blocks

A code block groups a sequence of statements into a single unit.

- Assignment statements.
- Declaration statements. ← *a variable declared within a block is "local" (it is not accessible outside that block)*
- Other statements, including nested *if* statements.
- …

```
public class TwoSort {
   public static void main(String[] args) {
      int a = Integer.parseInt(args[0]);
      int b = Integer.parseInt(args[1]);

      if (b < a) {
         int temp = a;
         a = b;
         b = temp;
      }

      System.out.println(a);
      System.out.println(b);
   }
}
```

code block (swap values in a *and* b*)*

```
~/cos126/conditionals> java TwoSort 1234 126
126
1234

~/cos126/conditionals> java TwoSort 126 1234
126
1234
```

Examples of *if* statements in practice

task	Java code

choose singular or plural suffix	`String result = price + " dollar";`
	`if (price != 1) {`
	`  result = result + "s";`
	`}`

check if donor is ineligible to donate blood	`if ((age < 16) || (weight < 110)) {`
	`  System.out.println("ineligible");`
	`}`

normalize minutes and hours	`if (minutes >= 60) {`
	`  minutes = minutes - 60;`
	`  hours = hours + 1;`
	`}`

find the maximum of three integers	`int max = a;`
	`if (b > max) max = b;`
	`if (c > max) max = c;`

"compound" boolean expression

multiple statements inside the block

curly braces optional when that block contains only one statement

Conditionals: poll 1

What is the result of compiling and executing this program?

A. 6 6

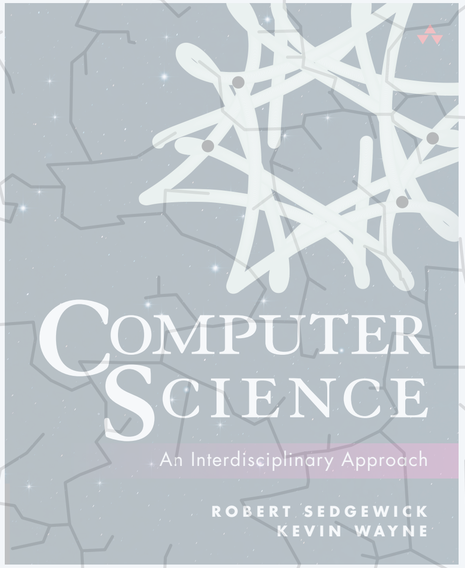
B. 6 7

C. 7 6

D. Compile-time error

E. Runtime exception

```
public class Mystery1 {
   public static void main(String[] args) {
      int a = 6;
      int b = 7;
      int smallest = a;
      int largest = b;

      if (smallest > largest)
         smallest = b;
         largest = a;

      System.out.println(smallest + " " + largest);
   }
}
```



https://introcs.cs.princeton.edu

1.3 CONDITIONALS

- *booleans*
- *if statements*
- *if–else statements*
- *nested conditionals*
- *case study: text-to-speech (years)*

The *if-else* statement

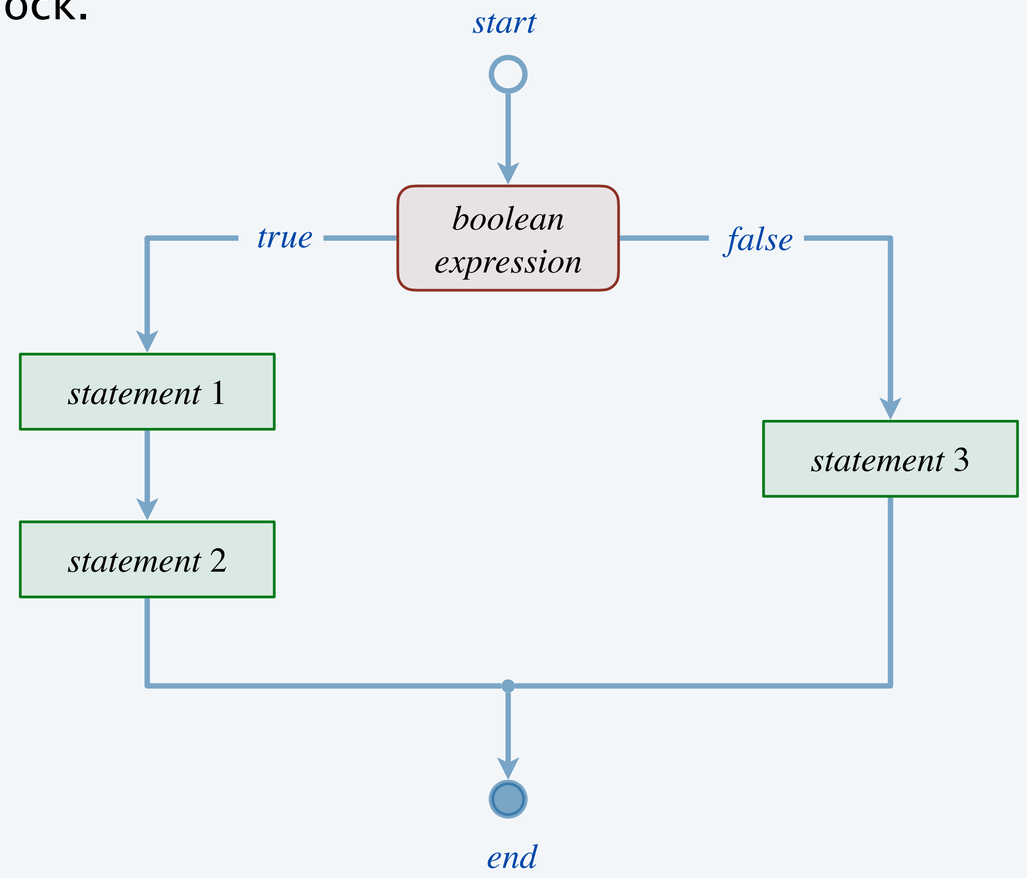
Execute one of two code blocks based on whether a specified condition is true.

- Evaluate a boolean expression (the condition).
- If true, execute the statements in the first code block.
- If false, execute the statements in the second code block.

```
if (<boolean expression>) {
   <statement 1>
   <statement 2>
}
else {
   <statement 3>
}
```

the `else` *clause*

if-else statement template



if-else statement flowchart

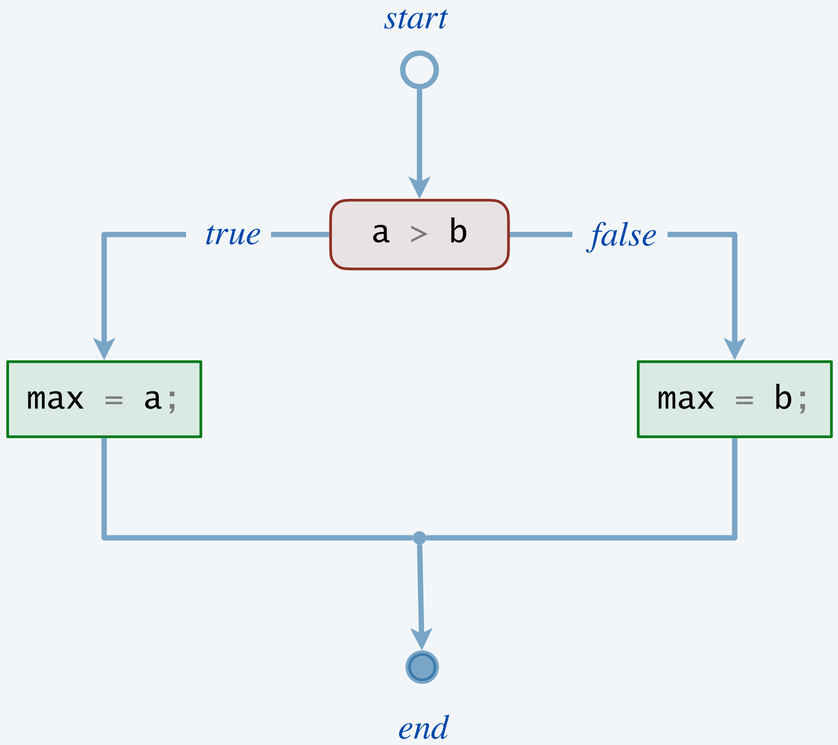
The *if-else* statement

Execute one of two code blocks based on whether a specified condition is true.

- Evaluate a boolean expression (the condition).
- If `true`, execute the statements in the first code block.
- If `false`, execute the statements in the second code block.

```
int max;
if (a > b) {
   max = a;
}
else {
   max = b;
}
```

sets max to the larger of a and b



if-else statement flowchart

Quadratic formula error check

Ex. Error check for roots of quadratic equation $ax^2 + bx + c = 0$, assuming $a \neq 0$.

```
public class Quadratic {
   public static void main(String[] args) {
      double a = Double.parseDouble(args[0]);
      double b = Double.parseDouble(args[1]);
      double c = Double.parseDouble(args[2]);
      double discriminant = b*b - 4.0*a*c;

      if (discriminant < 0.0) {
         System.out.println("no real roots");
      }

      else {
         double d = Math.sqrt(discriminant);
         double root1 = (-b + d) / (2.0*a);
         double root2 = (-b - d) / (2.0*a);
         System.out.println(root1);
         System.out.println(root2);
      }

   }
}
```

```
~/cos126/conditionals> java Quadratic 1.0 -3.0 2.0
2.0
1.0

~/cos126/conditionals> java Quadratic 1.0 -1.0 -1.0
1.618033988749895
-0.6180339887498949

~/cos126/conditionals> java Quadratic 1.0 1.0 1.0
no real roots
```

$x^2 - 3x + 2$

$x^2 - x - 1$

$x^2 + x + 1$

Simulating a fair coin flip

Goal. Simulate a fair coin flip.

Helpful: `Math.random()` returns a `double` value in the range $[0, 1)$.

```
public class CoinFlip {
   public static void main(String[] args) {
      double r = Math.random();

      if (r < 0.5) {
         System.out.println("Heads");
      }
      else {
         System.out.println("Tails");
      }
   }
}
```

```
~/cos126/conditionals> java CoinFlip
Heads

~/cos126/conditionals> java CoinFlip
Tails

~/cos126/conditionals> java CoinFlip
Tails
```

Bottom line. Conditionals let a program take different actions depending on a variable's value.

Examples of *if-else* statements in practice

task	Java code

simulate a fair bet	`double r = Math.random();`
	`if (r < 0.5) cash = cash + bet;`
	`else cash = cash - bet;`

check whether a number is odd or even	`String parity;`
	`if (n % 2 == 0) parity = "even";`
	`else parity = "odd";`

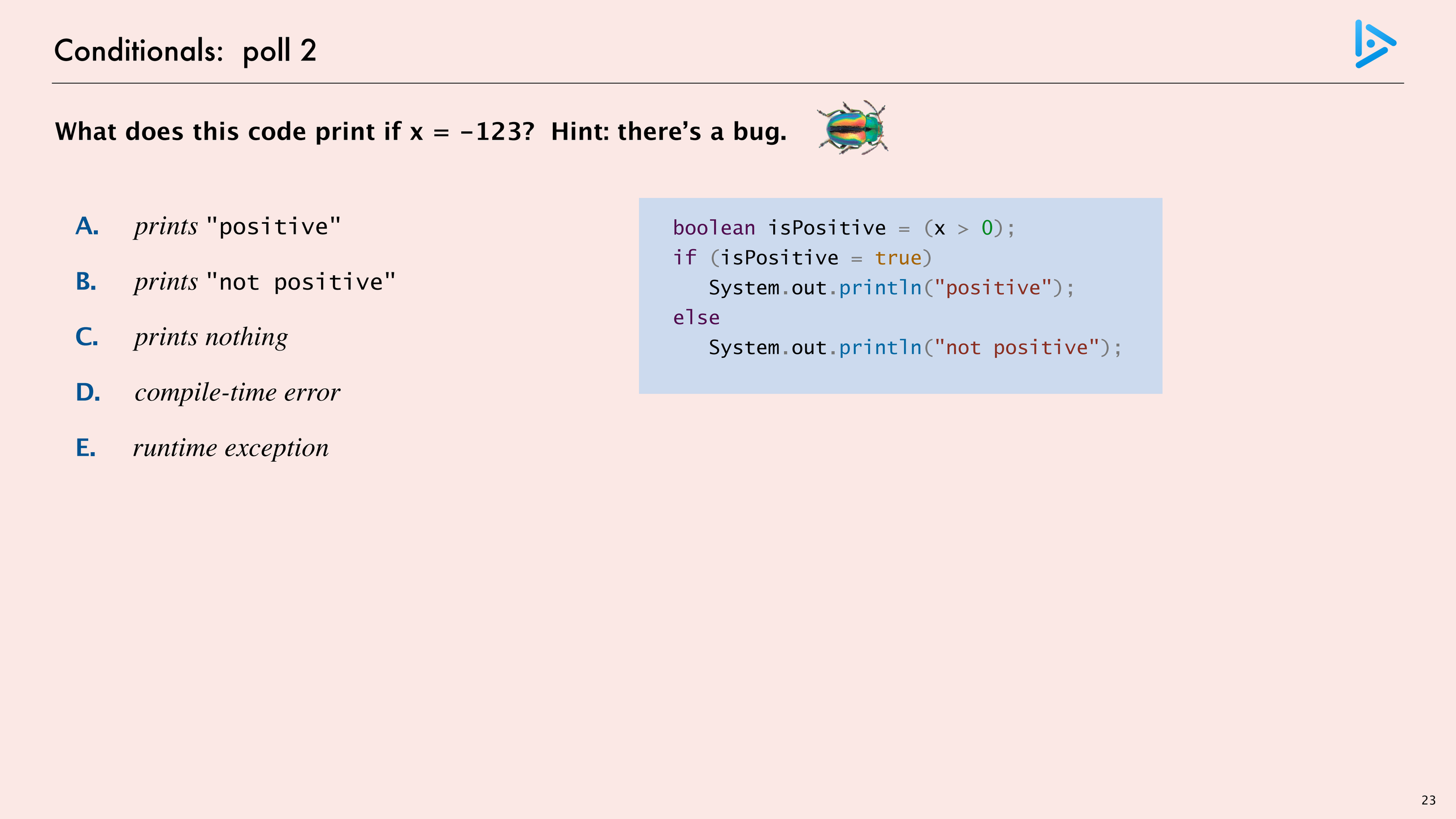
compute integer remainder (with guard clause)	`if (denominator == 0) {`
	`  System.out.println("division by zero");`
	`}`
	`else {`
	`  int remainder = numerator % denominator;`
	`  System.out.println("remainder = " + remainder);`
	`}`

curly braces are optional when that block contains only one statement

even: …, –4, –2, 0, 2, 4, …

preferred style: delimit every block with curly braces (even when optional)

Conditionals: poll 2



What does this code print if x = -123? Hint: there's a bug.

A. *prints* `"positive"`

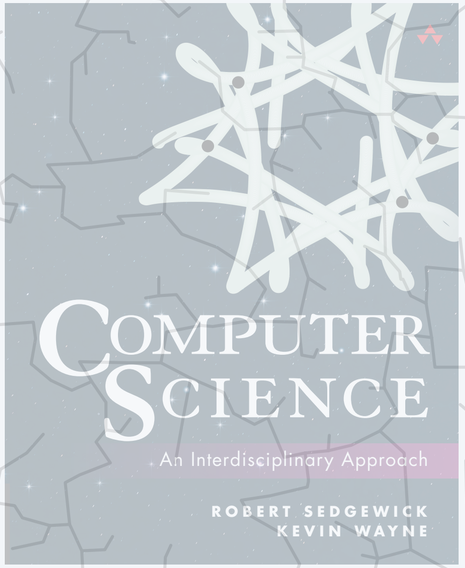
B. *prints* `"not positive"`

C. *prints nothing*

D. *compile-time error*

E. *runtime exception*

```
boolean isPositive = (x > 0);
if (isPositive = true)
   System.out.println("positive");
else
   System.out.println("not positive");
```



https://introcs.cs.princeton.edu

1.3 CONDITIONALS

- *booleans*
- *if statements*
- *if–else statements*
- *nested conditionals*
- *case study: text-to-speech (years)*

Nesting conditionals: rock, paper, scissors

Three-way selection: rock, paper, scissors.

```
public class RockPaperScissors {
   public static void main(String[] args) {
      int r = (int) (Math.random() * 3);

      if (r == 0) {
         System.out.println("Rock");
      }
      else {
         if (r == 1) {
            System.out.println("Paper");
         }
         else {
            System.out.println("Scissors");
         }
      }
   }
}
```

0, 1, *or* 2

`if-else` *statement nested inside the* `else` *clause of another* `if` *statement*

```
~/cos126/conditionals> java RockPaperScissors
Rock

~/cos126/conditionals> java RockPaperScissors
Scissors
```

Bottom line. Exactly one print statement executes.

Nesting conditionals: marginal tax rate

Multiway selection. Given income, calculate marginal tax rate.

```
public class TaxRate {
   public static void main(String[] args) {
      int income = Integer.parseInt(args[0]);
      double rate;

      if (income < 47450) rate = 0.22;
      else {
         if (income < 114650) rate = 0.25;
         else {
            if (income < 174700) rate = 0.28;
            else {
               if (income < 311950) rate = 0.33;
               else                 rate = 0.35;
            }
         }
      }

      System.out.println(rate);
   }
}
```

income	rate
0 – $47,449	22%
$47,450 – $114,649	25%
$114,650 – $174,699	28%
$174,700 – $311,949	33%
$311,950 +	35%

if statement nested within an if statement

if statement nested within an if statement within an if statement

if statement nested within an if statement within an if statement within an if statement

```
~/cos126/conditionals> java TaxRate 100000
0.25
```

Multiway selection shorthand

Curly braces optional here: Each block contains a single (compound) statement.

```
public class TaxRate {
   public static void main(String[] args) {
      int income = Integer.parseInt(args[0]);
      double rate;

      if      (income <  47450) rate = 0.22;
      else if (income < 114650) rate = 0.25;
      else if (income < 174700) rate = 0.28;
      else if (income < 311950) rate = 0.33;
      else                      rate = 0.35;

      System.out.println(rate);
   }
}
```

5 mutually exclusive alternatives

income	rate
0 – $47,449	22%
$47,450 – $114,649	25%
$114,650 – $174,699	28%
$174,700 – $311,949	33%
$311,950 +	35%

Bottom line. Exactly one assignment statement executes.

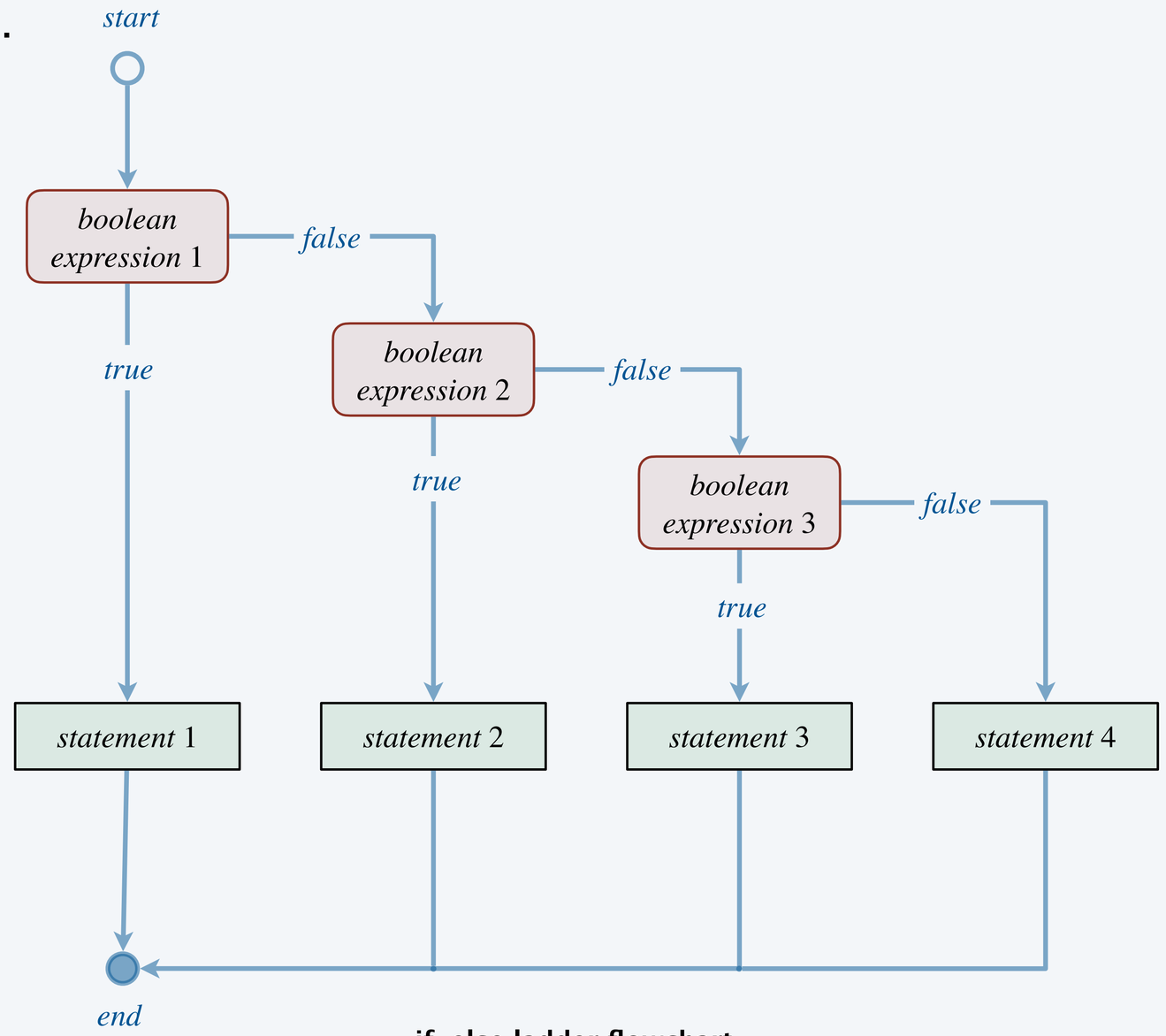
```
~/cos126/conditionals> java TaxRate 100000
0.25
```

A ladder of nested *if-else* statements

Multiway selection. Exactly one alternative executes.

```
if (<boolean expression 1>) {
   <statement 1>
}
else if (<boolean expression 2>) {
   <statement 2>
}
else if (<boolean expression 3>) {
   <statement 3>
}
else {
   <statement 4>
}
```

if-else ladder template



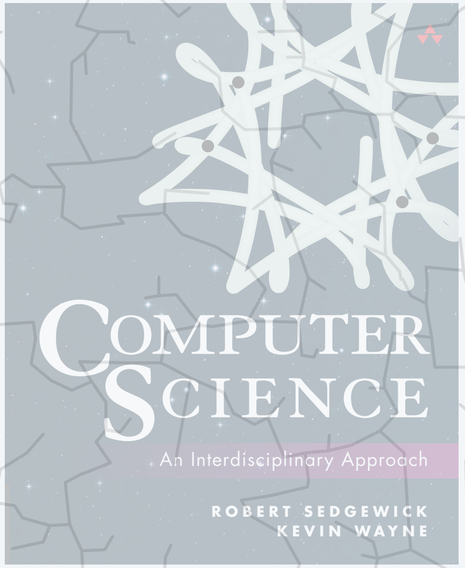
if-else ladder flowchart

Examples of multiway selection in practice

task	Java code
classify flow regime based on Reynolds number (ratio of inertial to viscous forces)	`if (reynolds <= 2000.0) { System.out.println("laminar flow"); } else if (reynolds >= 3500.0) { System.out.println("turbulent flow"); } else { System.out.println("transitional flow"); }`
compute sign function $sign(x) = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ +1 & \text{if } x > 0 \end{cases}$	`double sign; if (x < 0.0) sign = -1.0; else if (x > 0.0) sign = +1.0; else sign = 0.0;`

exactly one of 3 alternatives executes

exactly one of 3 alternatives executes



https://introcs.cs.princeton.edu

1.3 CONDITIONALS

- *booleans*
- *if statements*
- *if–else statements*
- *nested conditionals*
- *case study: text-to-speech (years)*

How to speak a year in English

Goal. Convert a numeric year ($1 - 9999$) into the way it is typically spoken in English.

- Split the year into two two-digit numbers; say each pair aloud.
- Special cases:
  - year ends in 000: say *thousand* for last three digits
  - year ends in 00 (but not 000): say *hundred* for last two digits
  - year ends in 01 to 09: say *oh* followed by the single digit
  - year begins with 00: skip the first two digits

year	spoken
2024	*twenty twenty-four*
1776	*seventeen seventy-six*
2000	*two thousand*
1700	*seventeen hundred*
1901	*nineteen oh one*
0026	*twenty-six*
12345	*invalid year*

Domain-specific synthesis

Main idea. Concatenate pre-recorded audio clips to speak the output.

speaking the year 1901

word(s)	audio clip(s)
1–99	1.wav, 2.wav, 3.wav, ...
hundred	hundred.wav
thousand	thousand.wav
oh	oh.wav

audio vocabulary

Applications.

- Talking clocks and timers.
- Train schedule announcements.
- Interactive telephone voice response systems.

Note. Output is limited by the available recorded words.

Live coding

```
public class SayYear {
   public static void main(String[] args) {
      int year = Integer.parseInt(args[0]);
      int firstTwoDigits = year / 100;
      int lastTwoDigits  = year % 100;

      if (year % 1000 == 0) {
         int firstDigit = year / 1000;
         StdAudio.play(firstDigit + ".wav");
         StdAudio.play("thousand.wav");
      }

      else {
         if (firstTwoDigits > 0)
            StdAudio.play(firstTwoDigits + ".wav");

         if (lastTwoDigits == 0)
            StdAudio.play("hundred.wav");

         else {
            if (lastTwoDigits < 10)
               StdAudio.play("oh.wav");

            StdAudio.play(lastTwoDigits + ".wav");
         }
      }
   }
}
```

assumption: `year` *is between* 1 *and* 9999

compute the first two and last two digits of the year

if the year ends in 000,
say the first digit, then "thousand"

say the first two digits (*skip* 00)

if the year ends in 00 (*but not* 000),
say "hundred"

if the year ends in 01 *to* 09, *say "oh"*

say the last two digits

Testing

Principle. Supply inputs that exercise every possible execution path in the program. ← *ensures that every line of code is executed by at least one test*

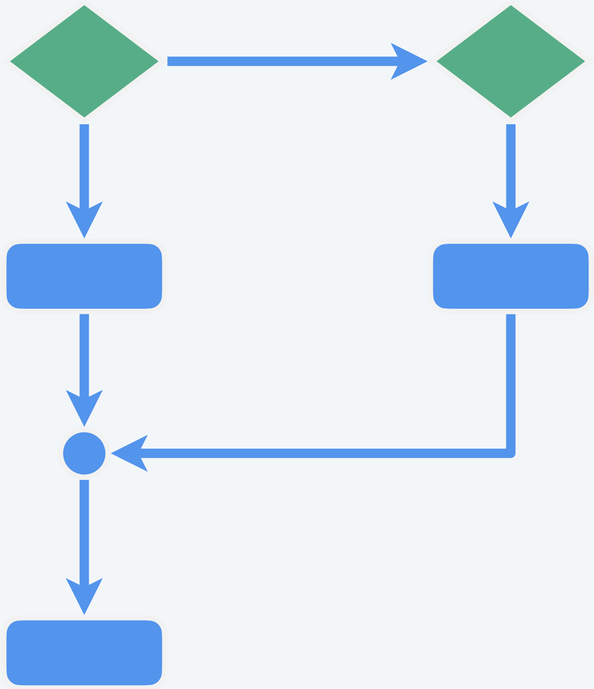
```
~/cos126/conditionals> java-introcs SayYear 2024   ← typical case
[speaks "twenty twenty-four"]

~/cos126/conditionals> java-introcs SayYear 1776   ← typical case
[speaks "seventeen seventy-six"]

~/cos126/conditionals> java-introcs SayYear 2000   ← year ends in 01 to 09
[speaks "two thousand"]

~/cos126/conditionals> java-introcs SayYear 1700   ← year ends in 000
[speaks "seventeen hundred"]

~/cos126/conditionals> java-introcs SayYear 1901   ← year ends in 00 (but not 000)
[speaks "nineteen oh one"]

~/cos126/conditionals> java-introcs SayYear 26     ← year begins with 00
[speaks "twenty-six"]
```

Summary

One-way selection. An `if` statement.

Binary selection. An `if-else` statement.

Multiway selection. A ladder of nested `if-else` statements.



control flow with conditionals

Credits

media	source	license
Decision Making	nextlevelscoaching.com	non-commercial use
Scientific Calculator	Fornax at Wikimedia	CC BY-SA 3.0
!false Meme	Tees Design	
Coin Toss	clipground.com	CC BY 4.0
Types of Triangles	Adobe Stock	education license
Bugs	Adobe Stock	education license
Russian Nesting Dolls	Adobe Stock	education license
Rock, Paper, Scissors	Adobe Stock	education license
Watering Can	Katerina Kamprani	
Digital Clock	Chrkl at Wikimedia	CC BY 3.0
Live Coding Icon	Adobe Stock	education license
Code Testing Icon	Adobe Stock	education license
The Year in English	Woodward English	