

Objectives

- More Java fundamentals
 - Static data types
 - Arithmetic operators
 - Relational operators
 - `java.lang` classes: `Math` and `String` class

Sept 15, 2021

Sprenkle - CSCI209

1

1

Review

- Why Java?
- How do you compile and run Java programs?
- How do you display output in Java?
- What are the modifiers for the `main` method?
 - What are the argument(s) to `main`?
 - How do you *call* the `main` method?
- How does Java compare to Python (so far)?
- Unix commands:
 - How do you make a directory?
 - How do you view the contents of a directory?
 - How do you go into a directory?

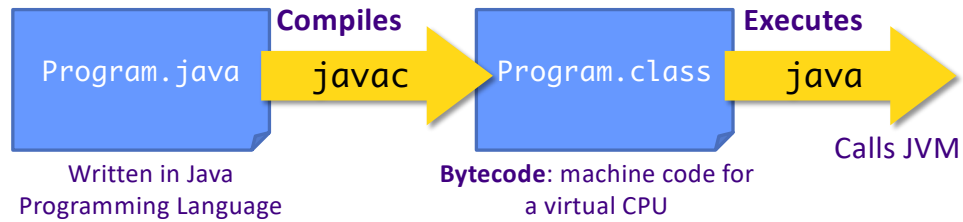
Sept 15, 2021

Sprenkle - CSCI209

2

2

Review: Compiling, Executing Java Programs



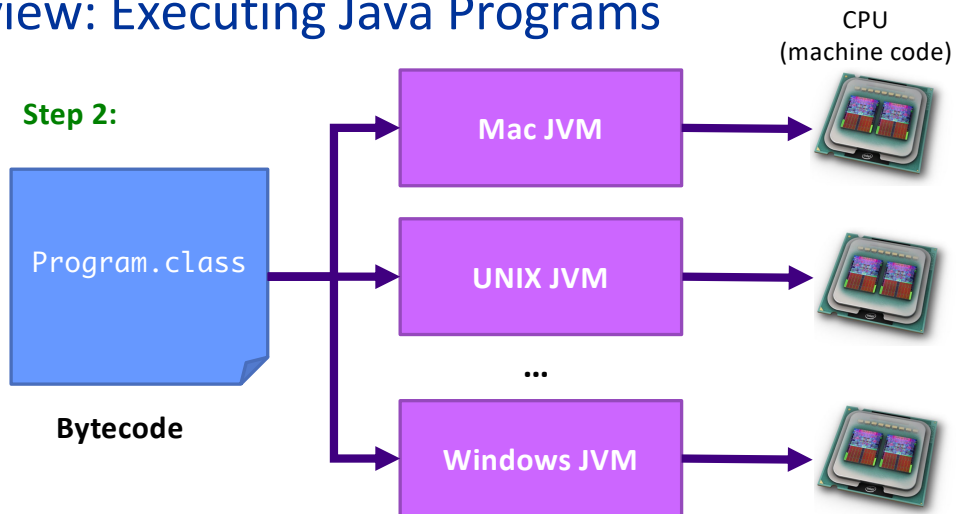
Sept 15, 2021

Sprenkle - CSCI209

3

3

Review: Executing Java Programs



- Same **bytecode** is executed on each platform
- Don't need to provide the source code

Sept 15, 2021

Sprenkle - CSCI209

4

4

Review: Example Java Program

```
/**
 * Our first Java class: displays Hello!
 * @author Sara Sprenkle
 */
public class Hello {
    public static void main(String[] args) {
        //print a message
        System.out.println("Hello!");
    }
}
```

main method is automatically called when you run

```
java Hello
```

5

Review: Benefits of Java

- Rapid development of programs
 - Large library of classes, including GUIs, Enterprise-level applications, Web applications
- Portability
 - Run program on multiple platforms without recompiling
- Compiled
 - Find some errors before execution!
 - Statically typed
 - Can give performance boost by doing optimizations

6

Review: Unix Commands

- How do you make a directory? `mkdir`
- How do you view the contents of a directory? `ls`
- How do you go into a directory? `cd dirname`

Sept 15, 2021

Sprenkle - CSCI209

7

7

Reflection: Assign 0

- How did it go?
- How long did it take?
- What tips/tricks did you learn/would you recommend?
- Why is part of the assignment debugging given code?
 - Why is that approach beneficial to you?

Sept 15, 2021

Sprenkle - CSCI209

8

8

Reflection: Assignment 0

- I am “overcommenting” the Java programs
 - I explain things that you don’t need to explain in your assignments (but you should certainly understand!)
- Expectations for your assignments
 - Comments
 - High-level comment at top of program
 - Mark authorship with @author at bottom of top comment
 - Explain blocks of code or difficult code
 - Comments that help you
 - Expectations will change as we learn more
 - Name classes and output files as I request

9

JAVA FUNDAMENTALS

10

Java keywords/reserved words

- Case-sensitive
- Can't be used for variable or class names
- Reserved words seen so far ...
 - **public**
 - **class**
 - **static**
 - **void**
- Exhaustive list
 - http://docs.oracle.com/javase/tutorial/java/nutsandbolts/_keywords.html

Sept 15, 2021

Sprenkle - CSCI209

11

11

Data Types

- Java is **strongly-typed**
 - Every variable must have a **declared type**
- All data in Java is an **object** – except for the **primitive data types**:

int	4 bytes (-2,147,483,648 -> 2,147,483,647)
short	2 bytes (-32,768 -> 32,767)
long	8 bytes (really big integers)
byte	1 byte (-128 -> 127)
float	4 bytes (floating point)
double	8 bytes (floating point)
char	2 bytes (Unicode representation), single quotes
boolean	true or false


Sept 15, 2021


Sprenkle - CSCI209

12

12

Variables

- Must be **declared** before used
 - Syntax: `<datatype> <name> [= value];`


Optional assignment
- Variable names typically start with *lowercase* letter
 - `_` (underscore) also a valid first character
 - Convention: Subsequent words are capitalized
 - Examples:  `myFile, firstCousinOnceRemoved`
 - Called “Camel Casing”


Sept 15, 2021


Sprenkle - CSCI209

13

13

Variable Examples

- Must be **declared** before used
 - Syntax: `<datatype> <name> [= value];`
- Examples:
 - `int x;`
 - `double pi = 3.14;`
 - `char exit = 'q';`
 - `boolean isValid = false;`


Camel Casing 

Note **must** use *single* quotes for **chars**

Sept 15, 2021

Sprenkle - CSCI209

14

14

Python Transition **Warning**

You can**not** redeclare a variable name in the same scope

- OK:

```
int x = 3;
x = -3;
```

- Not OK:

```
int x = 3;
int x = -3;
boolean x = true;
```

Compiler errors

Sept 15, 2021

Sprenkle - CSCI209

15

15

More Data Type-Related Information

- Result of integer division is an **int**
 - Same as Python 2, **not** Python 3
 - Example: $4/3 = ??$
- Casting
 - Similar to Python for primitive types
 - Example: $4/(\text{double}) 3$

TestScore.java

Sept 15, 2021

Sprenkle - CSCI209

16

16

Floats in Java

- Decimal literals are considered *doubles*
- This code won't compile:

```
float f = 3.14;
```

Compiler reads 3.14
as a *double*

- Compiler error message:

```
Float.java:15: error: incompatible types: possible
lossy conversion from double to float
    float f = 3.14;
              ^
1 error
```

- To fix code, add an **f** to specification of number or
declare as double

Sept 15, 2021

Sprenkle - CSCI209

Float.java

17

17

Escape Sequences

Same as Python:

- Combination of characters to represent something else
- Escape character: \

Meaning	Sequence
Newline character (carriage return)	\n
Tab	\t
Quote	\"
Backslash	\\

- In Java, you can print a ' without escaping
- What does the following display?

```
System.out.println("To print a \\, you must use  
\"\\\\\\\\\\\\\"");
```

EscapeCharacters.java

Sept 15, 2021

Sprenkle - CSCI209

18

18

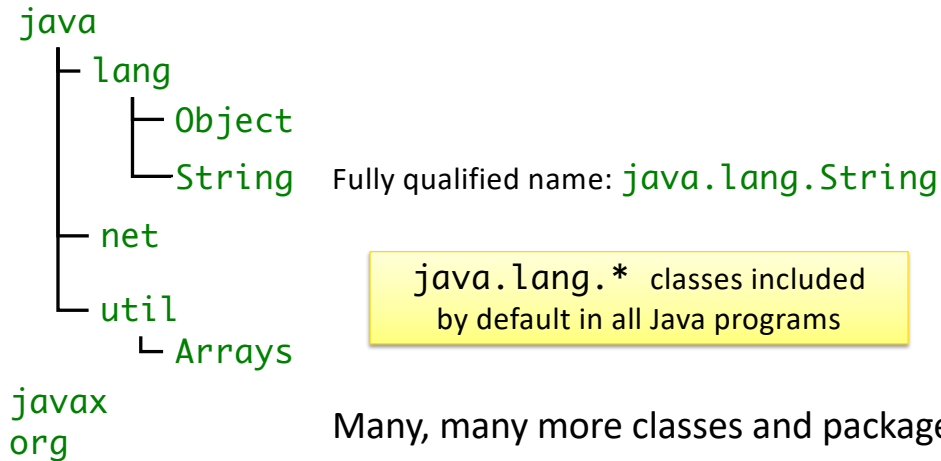
Arithmetic, Relational Operators

- Java has most of the same operators as Python:
 - Arithmetic operators: +, -, *, /, %
 - No power operator: **
 - Relational operators: ==, !=, <, >, <=, >=
 - Evaluate to a **boolean** value
 - Increment and decrement
 - += x, -= y, etc.
 - Additional shortcut for += 1, -=1: ++ , --

INTRODUCTION TO JAVA LIBRARIES

Java Libraries

- Organized into a hierarchy of **packages**



Sept 15, 2021

Sprenkle - CSCI209

21

21

Java API Documentation

<https://docs.oracle.com/en/java/javase/16/docs/api/index.html>

- API:** Application Programming Interface
 - What the class can do for YOU!
- Complete documentation of every class included with the JDK
 - Every method and variable contained in class
- Bookmark it!
 - Too many classes, methods to remember them all
 - Refer to it often

Sept 15, 2021

Sprenkle - CSCI209

22

22

java.lang.String class

- Similar functionality to Python but accessed differently
 - Mostly through *methods!*
- **Included by default** in every Java program
- Strings are represented by **double quotes**: ""
 - Single quotes represent **chars** only
- Examples:

```
String emptyString = "";
String niceGreeting = "Hello there.";
String badGreeting = "What do you want?";
```

Sept 15, 2021

Sprenkle - CSCI209

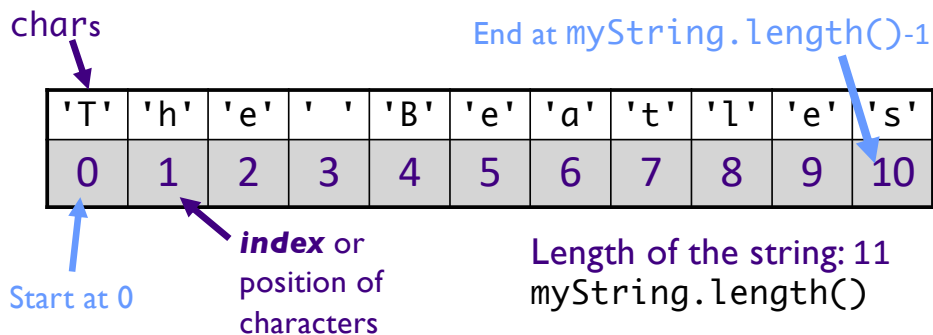
23

23

Strings

- A **char** at each position of String

```
String myString = "The Beatles";
```



Use **charAt** method to access chars

Sept 15, 2021

Sprenkle - CSCI209

24

24

String method: charAt

- A `String` is a collection of chars

```
String testString1 = "Demonstrate Strings";

char character1;
char character2 = testString1.charAt(3);
character1 = testString1.charAt(testString1.length()-2);

System.out.println(character1 + " " + character2);
```

Displays "g o"

Python Transition Gotcha: Can't use negative numbers for indices as in Python

Sept 15, 2021

Sprenkle - CSCI209

25

25

String methods: substring

- Like *slicing* in Python
- `String substring(int beginIndex)`
 - Returns a new `String` that is a substring of this string, from `beginIndex` to end of this string
- `String substring(int beginIndex, int endIndex)`
 - Returns a new `String` that is a substring of this string, from `beginIndex` to `endIndex-1`

```
String language = "Java!";
String subStr = language.substring(1);
String subStr2 = language.substring(2, 4);
```

subStr is "ava!"
subStr2 is "va"

Sept 15, 2021

Sprenkle - CSCI209

26

26

String Concatenation

- Use **+** operator to concatenate Strings

```
String niceGreeting = "Hello";
String firstName = "Clark";
String lastName = "Kent";
String blankSpace = " ";

String greeting = niceGreeting + ", " +
    blankSpace + firstName +
    blankSpace + lastName;

System.out.println(greeting);
```

Prints "Hello, Clark Kent"

Sept 15, 2021

Sprenkle - CSCI209

27

27

Review: String Concatenation

- If a **String** is concatenated with something that is not a **String**, the other variable is converted to a **String** automatically.

```
int totalPoints = 110;
int earnedPoints = 87;
double testScore = (double) earnedPoints/totalPoints;

System.out.println("Your score is " + testScore);
```

Converted to a String



Sept 15, 2021

Sprenkle - CSCI209

28

28

String Comparison: equals

- **boolean** equals(Object anObject)

- Compares this string to the specified object

```
String string1 = "Hello";
String string2 = "hello";
boolean test;
test = string1.equals(string2);
```

- **test** is false because the Strings contain different values

- Note that == does **not** do what you expect for Strings

- Compares that the objects are the same (like Python's is)

Sept 15, 2021

Sprenkle - CSCI209

Equals.java

29

29

String methods: and many more!

- **boolean** endsWith(String suffix)
- **boolean** startsWith(String prefix)
- **boolean** equalsIgnoreCase(String other)
- **int** length()
- String toLowerCase()
- String trim(): remove trailing and leading white space
- ...

See java.lang.String API for all available methods:

<https://docs.oracle.com/en/java/javase/16/docs/api/java.base/java/lang/String.html>

Sept 15, 2021

Sprenkle - CSCI209

30

30

To Do

- Textbook: Read “Java Data Types”, up to but not including List
- Assign 1
 - Part 0: Fixing compiler and logic errors from program
 - Part 1: Finding the file extension of a filename
 - Due Thursday at 11:59 p.m.