Review

Spring 2017
CS 102

Software

- Operating System: the program that manages a computer's resources
- Program: a sequence of instructions that performs some task
  - Performing an instruction is called "executing" an instruction

2

Compilation

- Translator:
  - translates a program from one language to another.
- Machine language:
  - the ones and zeros that a computer understands.
  - A low level language.
- Compiler: The translation software invoked at compile time
  - a translator which typically translates a high-level language into a low-level one
  - Java is a high-level language
  - Java's compiler translates Java code into bytecode
  - Bytecode is like machine language, but is not tied to a specific machine
- A Java bytecode interpreter is used to execute the bytecode
  - Called a Java Virtual Machine (JVM) is the software invoked at runtime.

3

Terminology

- Abstraction
  - Taking away the complexity of a problem by representation with simpler classes. We "abstract away" from the complexity with a simpler representation.
- Encapsulation or Information Hiding
  - Not revealing how the method does it's work. Data members are private; accessor and mutator methods (if they exist) are public.
- Modularity
  - Dividing code into smaller pieces (modules), each one of which is easier to code.

4

OOP Terminology

- OOP (Object-Oriented Programming) languages:
  - Encapsulate code inside the class' methods
  - Use additional methods for modularity
- A (primitive) type is the basic unit of storage in Java
  - A type is a template for a variable
- A class is composed of types (or other classes) as well as methods
  - A class is a template for an object
- All variables must have their type declared before they are used. Creating an object from a class is called instantiating the object, using the new keyword.

5

Problem solving steps

- Analysis
  - What needs to be done?
- Design
  - How is it going to be done?
- Implementation
  - Solution to problem with correct inputs.
- Testing
  - Does it work correctly?

6

Readable programs

- Comments are English text
  - Single lines have a // before them in a Java file
  - /* */ or /** */ are multiline comments
- Very long lines should be broken into smaller ones.
- Blank lines make a program easier to read
- Indentation helps humans identify which code is within {}'s
- Keywords have special meanings in Java; can't be used for identifier names
  - Examples: int, double, class, static, public

7

Identifiers

- Identifiers: programmer-defined names
  - For classes, variables, methods, etc.
  - Cannot be a keyword
  - Must start with a letter (or _ or $)
  - Can contain numbers also (but not as the first character)
- Good identifiers: radius, width, position
- Bad identifiers: x, y, q, the_really_really_long_variable_name_hi_mom Identifiers like susan and edward for numbers. Names should reflect variable purpose

8

Computer bugs

- A bug is an error in the program, at compile time or runtime
- To debug is to remove bugs (errors)

9

Java classes

- The class keyword is used to start a class declaration
  - Can be made public
  - Classes start with "public class ClassName"

inheritance ↓ keywords

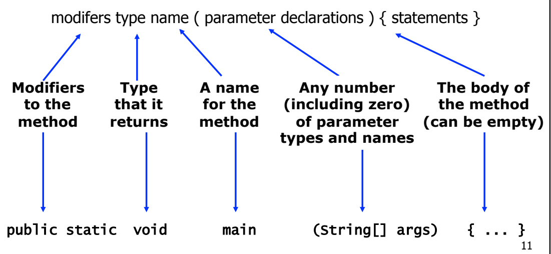
- Purpose of classes:

1. A class can be a library of static methods
2. A class can be a "template" for objects
   - Just as a type is a "template" for a variable

10

Java methods

- All methods have the following syntax:


modifers type name (parameter declarations) { statements }
Modifiers to the method
Type that it returns
A name for the method
Any number (including zero) of parameter types and names
The body of the method (can be empty)
public static
void
main
(String[] args)
{ ... }

11

Program execution

- Java starts executing a program at the beginning of the main() method
- Braces { } are used to specify where a method begins and ends
- A statement ends when a semicolon is encountered
  - A statement can span multiple lines

12

Misc Information

- A literal character string is a sequence of characters enclosed by double quotes
- System is the Java class that allows you to access parts of the computer system
  - System.in: access to the keyboard
  - System.out: access to the monitor
- Period is used for selection: Math.round
  - Given String s, select a method via: s.substring()
- An exception is when Java "panics"
  - It means something is wrong during run time

13

Escape sequences

- Java provides escape sequences for printing special characters

\n	newline
\t	tab
\\	backslash
\"	double quote
\ '	single quote

14

Primitive variable types

- Java has 8 (or so) primitive types:

float	**real numbers**
double	
boolean	**two values: true and false**
char	**a single character inside 's**
byte	**integer numbers**
short	
int	
long	

15

Constant names vs. literal values

- Which is easier to enter:
  - Math.PI
  - 3.141592653589793
- Entering a constant reduces chances of errors
- It allows for easily finding and changing the constant later on
- Constants are usually declared final so changes can't be made in a program

16

References and variables

- A primitive **variable** is an actual spot in memory that holds a (primitive type) value
- A variable **reference** is a memory address that points to another spot in memory where the object is stored.
- Variables defined in a class but outside a method are initialized to a default value (global to class)
- Variables defined in a method are **not** initialized to a default value (local to the method)

17

Math

- Standard operators: + - * /
- Note that / can be either integer division or floating-point division
- % computes the remainder (aka modulus)
- Can provide numbers in decimal or scientific notation

18

Expressions

- Evaluating an expression yields a result and a type
  - Example: 4/3 yields 1 of type int
  - Example: 3.5*2.0 yields 7.0 of type double
- Binary operator has two operands
  - Example: 3+4, 6*3, etc.
  - Left one is evaluated first
- Unary operator has one operand
  - Example: -3, etc.
- Operators have precedence
  - For example, * and / are evaluated before + and -

19

Operators

- Assignment: = pronounced ("gets")
- Increment (++) and decrement (--)
- Consider:

```
int i = 5;                    int i = 5;
System.out.println (i++);     System.out.println (++i);
System.out.println (i);       System.out.println (i);
```

- There are 4 ways to add 1 to an int:

```
i = i + 1;
i += 1;
i++;
++i;
```

compound operators

20

Casting

- Casting converts one type to another
- Example:

```
int x = 1;
System.out.println ((double) x);

double d = 3.4;
System.out.println ((int) d);
```

21

Scanner class

- Creating one:

```
Scanner stdin = new Scanner (System.in)
```

- Methods:
  - public int nextInt()
  - public short nextShort()
  - public long nextLong()
  - public double nextDouble()
  - public float nextFloat()
  - public String next()
  - public String nextLine()
  - public boolean hasNext()

22

References

- An object variable is really a reference to that object
- null represents an object variable that points to nothing
- Once there is no pointer to an object, Java automatically deletes that object
  - Called **garbage collection**
- A final object variable:
  - Only the reference (where it points in memory) is final
  - The values in the object can change via member methods
- We use constructors to create objects

23

Strings

- A String is a **sequence of characters**
- The **+** operator concatenates two Strings
- The **+=** operator appends a String
- First character has index 0
- A String can never be modified once created!

24

String methods

- length()
- substring()
- indexOf()
- lastIndexOf()
- charAt()
- trim()
- valueOf()

Logical expressions

- Logical expression has value either true or false
- Java has the boolean type with values true or false

Logical operators

- Three primary logical operators: **and (&&), or (||), not (!)**
- An **&&** operation is only true when both parts are true
- An **||** operation is true when either (or both) parts are true
- A **!** operation negates the value of the expression
- **!** operator is unary
- If the first boolean expression in an **&&** statement is false (or if the first boolean expression in an **||** is true), then the rest of the expression is skipped. This is called *short circuiting:*

```
if ((x > 0) && (3 / x == 1)) //the second part is not executed if the first part returns false
```

Equality

- Two equality operators: == and !=
- When comparing objects, == compares the references, not the objects themselves
- Use the .equals() method to test for object equality

Ordering

- Relational operators: ==, !=, <, >, <=, and >=. These only work on primitive types!
- Relational operators include the equality operators and the ordering operators
- For characters, ordering is based on the Unicode numbers of the characters

If statements

- An if statement has the form: if (expression) action
- An if-else statement has the form: if (expression) action1 else action2
- An if-else-if statement is used when there are many tasks to do, depending on the logical expressions

Switches

- A switch statement can be more readable than an if-else-if block
- Should always put either break at the end of each case of a switch, or a comment such as
  // FALLING THRU
- The **default** case means any case not matched by any of the previous cases

31

Exceptions

- try...catch blocks can be used to keep your code from crashing during execution.

  For example, you could put a try block around code that could cause an exception:

```
int[] arr = new int[9];
try {
    for (int i = 0; i <= arr.length; i++) {
        System.out.println(arr[i]);
    }
 } catch (??? ) { }   // exception generated in for loop?
```

32

Generating random numbers

- (int)(Math.random() * 12) + 1;
- The (int) in the expression above is called a **cast** operation. It is needed because the random method returns a double.
- The random method at the top of this slide is static...how do we know this without looking in the java api?

33