**Programming I**

**8/22/19**

OOP (object-orientated programming)

**Class**

Structure for encapsulating data and operations (methods) into one unit

Defines a template, or a model, for creating and manipulating objects

**Objects**

Data created using the class and its methods

An object is an instance of the class

Creating an object is called instantiation

1. **Write the source code using a text editor**

Save in a .java file

1. **Compile the source code**

Compiler makes a .class file

1. **Execute the application**

An IDE consist of a program editor, a compiler, and a run-time environment, integrated via a GUI

IF we just want to try out some java code without writing a complete application, we can use jshell

Command line is where you can find jshell

REPL = Read, Evaluate, Print, loop

// is a single line comment

/\* and \*/ are multiple line comments

Source file must be saved with the same name as the class name

**Compiler Errors**

Found by Compiler usually caused by incorrect syntax or spelling

**Run time Errors**

Reported by the JVM usually caused by incorrect use of prewritten classes or invalid data

**Logic**

Found by testing the program, incorrect program design or incorrect execution of design

Typical program structure: Input Data, perform some processing, output the results

Data may be different, but the instructions can be the same

**Programs**

All programs consist of at least one class

Identifiers are used to name classes, variables, and methods

They must start with a “java” letter

A-Z a-z $ \_ and Unicode letters

Can contain essentially any number of Java letters and digits but no spaces

Case sensitive

Number1 and number1 are different

Cannot be keywords or reserved word

Cannot start with numbers

**Identifiers**

For all data assign a identifier and a data type

Data types can tell a compiler how much memory to allocate how to store data and perform operations

Compilers monitor use of data

Java has eight primitive data types

**Byte short int long float double char Boolean**

(byte – long = integers) (float and double = decimals) (Boolean = true or false) (char = characters)

Variables can only hold one value at a time

**8/26/19**

**Declaring Variables**

Syntax = datatype identifier, or, datatype id.1, id.2, id.3

**Naming Convention for variables**

Camelcase

Ex. taxRate, testGrade, ect.

**Valid assignments**

Float salesTax = .05f;

Double taxRate = salesTax;

**Invalid assignment**

Double salesTax = .05;

Float taxRate = salesTax;

**String**

String is a class not a basic data type

String variables are objects

String literal: text contained within double quotes

**Escape keys**

Newline = \n

Tab = \t

Double quotes = \”

Single quote = \’

Backslash = \\

**Cosntants**

If you don’t want the value of a data variable to change you can declare it as a constant by adding **final** to the beginning of the decleration

Constants are all uppercase

Ex. Final double TAX\_RATE

**Arithmetic operators**

+ addition

-Subtraction

\* multiplication

/ division

% Modulus (remainder after division)

Follows order of operations

Modulus occurs with \* and /

**Explicit Type casting**

(Datatype)(Expression)

Int total = 100;

Int count = 30;

Double average = (double) (total/count)

Under Scores can be used in numbers in place of commas

**Short Cut Operators**

Count++; = count + 1

Count --; = count – 1

+= a =+ 3; a = a +3

Replace the plus with different functions for different effects

**Input**

Input using Scanner class

Provides methods for reading all the basic types and string

Scanner is in java.util

Scanner parses input into something sequences of characters called tokens

By default, tokens are separated by standard white space characters (tabs, spaces, newline, ect.)

**Scanner Constructor**

Creates a scanner object for reading from source. If source is System.in this instantiates a scanner object for reading from the java console

nextDataType () Returns the next token in the input stream can be any of the base data types. Replace data type with a base data type such as Boolean.

8/28/19

Scanner does not hava a nextChar method

To read a single character, read the input as a String, then extract the first character of the String into a char variable

**Using System.out**

Println = prints something and then goes to the next line

Print = prints something but stays on the same line

**Java packages**

Java.lang = basic functions

Java.awt = graphics for drawings aand using color

Java.swing = user-interface components

Java.text = classer for formatting numbers

**String concatenation operators**

+ = appends a string to another string

+= ex. S3 = s3 + ‘!’

Length() = returns the number of characters in a string

toUppercase = makes all characters upper case

toLowerCase = makes all characters lower case

indexOf = returns the index of the first occurrence of a ,string or a character, or -1 if not found

substring = returns a substring of a string beginning at the character at startIndex and ending at the character at endIndex -1

**The random class**

Random = generates a pseudorandom number (appearing to be random but is mathematically calculated)

Randon() creates a random number

nextInt = returns a random integer ranging from 0 up to but not including number

**The math class**

Stores E and Pi

Ex. System.out.println(Math.PI);

Abs = Absolute Value

Log = returns the natural log

Sqrt = returns the square root

Pow = raises the value to a power

Round = rounds a number

9/4/19

**Exit method**

Terminates a java program, a value of 0 indicates a normal termination. Any other value indicates an abnormal termination.

Min and Max can be used using the Math function

**Wrapper Classes**

Wrap the value of a primitive data type

Useful for converting strings into an int or double

Wrapper class is the upper case version of a primitive class ex. (int = Integer)

Autoboxing is the automatic conversion between a primitive type and a wrapper class

Unboxing is t he automatic conversion between a wrapper object and a primitive data type when a wrapper object is used where a primitive data type is expected

Parse int turns a string into an int

Value of coverts a string to an Integer

**Four types of flow control**

Sequential processing = executes in order

Method call = jump to code in a method then returns

Selection = Choose code to execute based on data value

Looping or iteration = repeat operations for multiple data values

**4 Cornerstones of computational thinking**

**Decomposition**

Breaking down a complex problem or system into smaller parts that are more manageable

**Pattern recognition**

Involves finding patterns can be used to help solve problems more efficiently

**Algorithm**

Is a plan a set of step by step instructions to solve a problem Sometimes written as a flow chart or as pseudocode

9/9/19

Absent

9/11/19

Nested ifs are if statements inside of nested if statements

Threshold values == how close should the values be, to be considered equal

If the abs value of the difference between the two numbers is less than the threshold value, the we will consider the values to be equal

The big decimal class can help with rounding issues

.equals is a way to compare Strings: it is case sensitive

Equalsignorecase is not case sensitive

Compareto compares the values of strings

The conditional operator (?:)

Condition ? True equal : False equal

Ex. Int absValue = ( a > 0 ? a : -a);

Switch statements can be used instead of an if/else/if

9/16/19

**Loops**

Executes a set of instructions repeatedly while some condition is true

Loops are used for accumulation, counting items, finding a average, finding a max or min, animation.

Can be used to check user input input

**While loop**

The condition is evaluated, if it is the loop is executed than revaluated. It keeps looping till false, then processes the result.

**Do While Loop**

The do while loop body is executed then the condition is checked. If true, the loop is repeated if false the loop is not repeated. It’s a while loop that always runs once first.

Syntax:

Do {

body

} while (condition);+3252

**For Loop**

Intialization, condition, and update all are on the same line

Syntax:

For (initialization; loop condition; loop update){

Output

}

9/18/19

Hasnext can be used to check inputs

Hasnext is used best inside a loop

9/23/19

Git hub and git

9/30/19

Methods

Modifier return-type methodname (parameter-list){

Body of method

}

Methods are usually named starting with a verb then an adj. ect.

findSum

calculateTotal

computeMax

setX and getX

11/11/19

Quiz II Study guide

1. Methods
2. Arrays
3. Wrapper Classes
4. ArrayLists
5. Error handling
6. Inheritance
7. Constructor
8. Polymorphism

Example Calculator GUI

Use grid

|  |  |  |  |
| --- | --- | --- | --- |
| Label | | | |
| 7 | 8 | 9 | \* |
| 4 | 5 | 6 | / |
| 1 | 2 | 3 | - |
| . | 0 | = | + |

**Programming II**

**OOP**

1. Java
2. Object
3. Grammar
4. Inheritence
5. AWT (Abstract Window Talkie)
6. Applet
7. API(Application Programming Interface)

First language was machine language

Next comes Assembly to improve machine language but limits on memory restricted language

With memory advancements came HLLs (High level languages): Java, C++, Pascal, ect.

HLL are compiled to assembly which computer can turn into binary

Java files are .java and when compiled they become .class files

Java has 3 properties = Syntax, JVM, and API

**Types of HLLs**

Procedural code is one block of text usually run top to bottom and is hard to understand when read

Structural code used multiple blocks or paragraphs to help with understanding and efficacy but can be hard to integrate with many people

OOP increases efficiency for multiple programmers

OOP allows for: 1. Meeting program requirements, 2. Working together efficiently, 3. Allows for reuse

To do this OOP has 3 properties: Encapsulation, Inheritance, and Abstraction.

IDE = Integrated Development Environment

3 Templates of Java:

1. Input is in the source
2. Input is given by a user
3. Input to source is a file
4. Output is given to another file

**2/3/20**

Object can be anything because they can be characterized by their attribute and their action

**2/10/20**

The 3 advantages of encapsulation = Modularity, easy to reuse, info hiding

IMPORTANT \*\*\*

Midterm on 3/11 12:30 – 1:30

Midterm review on 3/9

**3/2/20**

**Types of Operates**

Assignment (=)

Arithmatic ( +, -, /, \*, and %)

Bitwise(&, |, ^ , ~, >> , <<) = converts to binary

Comparison(>, <, >=, <=, ==, !=)

Conditional(&&, ||, : ?) : ? = Ternary = y= 3<2 : 2 ? 1 ; y = 1 : true and ? false

Decrement and Increment( ++, --)

Logic(!, !=)

**3/9/20**

**Midterm Review**

Object is everything

Attribute = field

Action = method

Keyword = Class = Blueprint

3 properties of OOP = Encapsulation, Inheritance, Abstraction

3 advantages of Encapsulation = Info Hiding, Modularity, and Easy plugin

3 properties of java = Syntax, JVM, and API

.java –(Compile)--> .class –(JVM)--> run code

Keywords new, and . used to make a new instance of a class

Keywords Private, public, default, protected, and this.

Getters and setters

Constructors 🡪 Default and Para-metered Constructors

this ()

static

OOP Business Model = Main 🡨🡪 UI 🡨🡪 🡨🡪 CRUD 🡨🡪Database

Types of Variables = Local(Initialized), Instance(Non static Field), Class(Static field), parameter

Identifier = naming variables rule

Reference = Uppercase (User defined datatype)

Primitive = 8 primitive data types

Byte 1 0 **To find range: (2^ # of bits / 2)** **: Plus side is - 1 (Bits is number of Bytes \* 8)**

Short 2 0

Int 4 0

Long 8 0L

Float 4 0.0f

Double 8 0.0d

Char 1 ‘\u0000’ = null character

Boolean 1 false **Range: True or False**

Types of operators (**Look at the notes above)**

Control statements = If Else, if else if else, switch case, for, extended for