LATHROP ENGINEERING

Name:

UNIT 2: BEGINNING JAVA

AP Computer Science A Unit Due Date: **September 28, 2018**

Welcome to the second unit of *AP Computer Science*! In this unit, we'll move beyond Jeroo to get into program *actual* Java programs. Here we will start from the very beginning and settle into the different challenges of programming at an easy pace. As we get started in this unit, our focus will be on the basic building blocks that make Java programs possible. In the end, the expectation is that you learn the following:

- How to use an IDE (BlueJ) to write, run, and troubleshoot your own programs using JAVA
- How to create a basic **class** to define a program and print "Hello World!"
- How to print text to the terminal (screen) in a variety of ways
- How to get input from a user
- How to do some simple mathematical calculations in your Java program

As we move through this unit, you are responsible for making adequate progress through the assignments, and for being done by the Unit Due Date (**September 28, 2018**). You are also responsible for completing each part before moving on to the next. Our unit is broken up into three main parts:

Part 1: **Beginning Java** *(20 pts) Approx. 3 days*		
The beginning of Java has to start with the classic "Hello World" program. In this first part to our unit, we'll do exactly that! Starting with the basics of class creation, the main method, and simple print statements, we'll be able to see how simple programs are built. We'll also get used to compiling our programs, troubleshooting and debugging, and running our programs.		▢ Basic Java & BlueJ Notes
		⊕ Complete "Hello World!"
		⊕ Complete Printing Tasks
		☆ Check-off from Mr. Benshoof

Part 2: **User Input** *(30 pts) Approx. 3 days*		
Once we can print things to the screen, the next logical step is to have the program ask for information from the user so that it can be used in the program's logic. Here we'll learn about the **Scanner** class, how to instantiate a Scanner object, and how to use an **if statement** to tell the program what to do!		▢ Scanner & If Notes
		⊕ Complete 6 Input Tasks
		☆ Check-off from Mr. Benshoof

Part 3: **Calculator** *(30 pts) Approx. 2 days*		
The last part of our unit will have you create your own calculator. To do this, we need to understand how to tell Java to do various math operations. Here we'll learn some of the math commands in Java, then brainstorm a group of complex equations (maybe from chemistry, physics, or engineering), and create a small calculator to do our math for us!		▢ Math in Java Notes
		⊕ Create Your Own Calculator
		▢ Write Log 2: *Beginning Java*
		☆ Check-off from Mr. Benshoof

(20 pts) Approx. 2 days

NOTE: Our IDE of choice – BlueJ – is already installed on all the computers in the lab. Please do not download and install a second copy.

Java begins here! In the first part of this introductory Java unit, we'll get to look at the basic building blocks of a Java program and of our chosen IDE: BlueJ. An IDE is an "Integrated Development Environment" that lets us write programs, compile programs, debug them when they don't work, and actually run them all from within the same application. IDEs make a programmer's life much simpler. While it's possible to write programs in a simple text editor like Notepad, IDEAs give us other tools to make the job of programming easier. To work through this introduction to Java, just follow the steps below!

1. Start by watching the three introductory videos on our website. These videos present the ideas of Java and also overview the BlueJ IDE. Take a full page of notes on the ideas presented in these videos.
2. Open BlueJ for yourself and create a new project called "Unit 1". All of your Unit 1 programs for each task should be saved within this one project to help keep things organized.
3. Create a new class called HelloWorld. Follow the information from the videos to create the HelloWorld program that simply prints the words "Hello World!" to the terminal. Your computer is now sentient!
4. Now, complete the first 5 Java Tasks below that all require printing text on the screen!
    a. JAVA TASK 1: Write a program that displays the numbers 1 to 8 on the same line using exactly ONE System.out.println statement with each number separated by a single space.
    b. JAVA TASK 2: Write a program that displays the numbers 1 to 8 on the same line using exactly FOUR System.out.print statements with each number separated by a single space.
    c. JAVA TASK 3: Write a program that displays the numbers 1 to 8 on the same line using exactly ONE System.out.printf statement with each number separated by a single space.
    d. JAVA TASK 4: Write an application that displays a box, an oval, an arrow, and a diamond using asterisks as shown on the right:
    e. JAVA TASK 5: Write a program that prints a 10x10 grid in the shape of a (solvable) maze. The maze should use asterisks (*) to create its walls.

```
********     ***       *         *
*      *    *   *     ***       * *
*      *   *     *   *****     *   *
*      *   *     *     *      *     *
*      *   *     *     *     *     *
*      *    *   *      *      *   *
********     ***       *       *
```

Part 1: Tasks	*5 points*	*4-3 points*	*2-1-0 points*
Basic Java & BlueJ Notes	+ Watch all presentations + You took a full page of notes on how basic Java programs are made + Your notes include details about BlueJ specifically	- Less than a full page of Java notes - No notes on using BlueJ specifically	- Very brief or no notes in your notebook
Hello World!	+ You completed the "Hello World" program	- You never got the program working fully	- You did not attempt the program
Java Tasks 1-5	+ You completed all 5 Java Tasks from this section	- You did not complete all 5 tasks	- You did not complete any tasks
Checkoff from Benshoof	+ Mr. Benshoof got to see your Java programs run successfully	N/A	N/A

(30 pts) Approx. 3 days

All good programs do more than just print things on the screen. If all we wanted was to simply print text to the screen, we could just watch a movie… maybe one with subtitles and no sound. In reality, we usually want our programs to have some kind of user input, and the easiest to work with is the keyboard. To let the user actually type things on the keyboard and pass that information along to the program itself, we need to start working with an object called a ***Scanner***. In this part of the unit, you'll learn about Scanners, how to create them, and how to use them to get information from the user.

1. Watch the three videos on Scanners and on the **if statement** control structure. Take a full page of good notes on each, making special note of how to instantiate (create) a Scanner, what methods it can use, and how if statements are formatted!
2. Now, complete the following Java Tasks below that all require using Scanners and/or if statements!
    a. JAVA TASK 6: Write a program that asks the user to enter two integers, obtains them from the user, and prints their sum, product, difference, and quotient (division).
    b. JAVA TASK 7: Write an application that asks the user to enter two integers, obtains them from the user, and then displays the larger number followed by the words "is larger". If the numbers are equal, print the message "These numbers are equal."
    c. JAVA TASK 8: Write an application that inputs three integers from the user and displays the sum, average, product, smallest, and largest of the numbers.
    d. JAVA TASK 9: Write an application that reads an integer from the user and prints whether or not the number is odd or even.
    e. JAVA TASK 10: Write a program that takes a decimal number (double) from the user and uses it as the radius of a circle. The program should then print the *diameter*, *area*, and *circumference* of a circle with that radius. Answers should be printed as a decimal, not just an integer.
    f. JAVA TASK 11: Write an application that inputs one number consisting of five digits from the user, separates the number into its individual digits and prints them separated by a space. For example, if the user typed "42339" the program should print
    "4 2 3 3 9".
    (A hint): When the user types their number, the computer will understand it as an integer… not as a word. So you need to figure out a way to break up a 5-digit number into it's individual digits. (hint: / and % would be useful here).

The 5th Operation: *MOD*

You probably thought you knew everything about algebra… but it turns out that $+, -, \div, and\ \times$ aren't the only operations in the world. There's a ***fifth*** operation called 'modulus' ("Mod" or "%" for short), and it returns *only the remainder* of a division problem!

EX: $7\%4 = 3$

Part 2: Tasks	*6-4 points*	*3-2 points*	*1-0 points*
Scanner, If, and % Notes	+ You took a full page of notes on Scanners, If Statements, and %	- You took less than a full page of notes	- Your notes are lacking or missing
	24-20 points (4 pts each)	*19-9 points*	*8-0 points*
Java Tasks 6-11	+ You completed all 6 Java Tasks from this section	- You did not complete all 6 tasks	- You did not complete most/any tasks
Checkoff from Benshoof	+ Mr. Benshoof got to see your Jeroo challenges work successfully!	N/A	N/A

(30 pts) Approx. 2 days

The final part of our unit has us look at how mathematics is done in Java, and what we can do with it! Certain functions are built naturally into the Java language like addition (+), subtraction (-), division (/), multiplication (*), and mod (%). Fancier things like exponents and square roots require the use of special methods built into a special class of methods called the ***Math*** class. Here, we'll learn how Java handles different mathematical operations and then use them to make a cool custom calculator for ourselves.

1. Watch the introductory videos about math in Java. To do special operations in Java like square roots, exponents, or absolute value, we have to use a special class called the **Math** class. As it turns out, the Math class is really easy to use and has all the functionality we could ever really want in an all-inclusive math resource. Take a full page of notes on the Math class and what we can do with it!
2. Now, think about your other classes. Pick one that has multiple challenging equations in it, and identify 3 different equations you have to use in that class. For example, if I was in Geometry/Trigonometry, I might want to use the Pythagorean Theorem, Law of Sines, and Law of Cosines as my three equations.
3. Once you've chosen your class and the corresponding equations, build a program that can do those calculations for you! Keep the following in mind:
    a. Your program should have a simple menu to let the user pick which equation to apply
    b. Your program can be simple in that it only needs to solve for one variable
    c. Your program should be user-friendly and easy to input numbers into
    d. Your program will need to be restarted every time you want to solve the equation another time
4. Finally, wrap up the unit with a 1-page writing assignment in your engineering notebook. Write a page describing what you think about these basic Java concepts. How did the tasks from this unit compare to those from Jeroo? What kinds of simple programs can you imagine making with what you already know? Who would use them? What questions do you still have about Java programming, and what are you excited for next?
5. Have Mr. Benshoof check your working calculator when you're finished, and check out other people's calculators to see what they did!

Part 3: Tasks	*5 points*	*4-3 points*	*2-1-0 points*
Math in Java Notes	+ You took a full page of notes on Java's Math class + Your notes include details on how to use a few of the methods in the Math class	- You took less than a full page of notes	- Your notes are lacking or missing
	15-10 points	*9-5 points*	*4-0 points*
Create Your Own Calculator	+ You created a functioning 3-equation calculator + Your calculator follows the parameters outlined above	- Your calculator solves 1 equation - Your calculator is missing some parts	- Your calculator is significantly lacking/missing
	10-8 points	*7-4 points*	*3-0 points*
Log 2: *Beginning Java*	+ You wrote a complete page in your engineering notebook + Your response addresses all the questions in the prompt provided	- You wrote less than a full page	- You wrote less than half a page
Checkoff from Benshoof	+ Mr. Benshoof got to see your calculator work successfully!	N/A	N/A