S21 Challenge: Language Basics

Description: Write a Java application using the NetBeans IDE that utilizes primitive data types, operators, expressions, statements, blocks, and control flow.

Purpose: This application provides experience working with the basic language features of Java. It is important when working with a new language to understand how it handles its data types, operators, expressions, statements, blocks, and control flow. A good thing to always do with a new language is to build test applications where you experiment with these language features to make sure you understand how they work.

Requirements:

Project Name: <Pawprint>LanguageBasicsS21

For the Project Name, follow the same naming scheme used in the previous challenges. The Project Name is to be comprised of your pawprint with the first letter capitalized followed by LanguageBasicsS21. For example, if the pawprint is **abcxyz9** the project is to be named **Abcxyz9LanguageBasicsS21**.

Write the code necessary to do the following inside the main() method of the application's Main Class created from NetBeans by default.

Declare the following variables and set their initial values as indicated:

- Declare a char, named **`c1`** that has an initial value of 'c'
- Declare a char, named **`c2`** that has an initial value of 99
- Declare a short, named **`qualityScore`** with an initial value of 61
- Declare a float, named **`miles`** that has an initial value of 150
- Declare a float, named **`milesPerGallon`** that has an initial value of 27.5
- Declare a float, named **`gasPrice`** that has an initial value of 2.34
- Declare a boolean, named **`sunny`** that has an initial value of false
- Declare a boolean, named **`warm`** that has an initial value of false
- Declare an int, named **`hour`** with an initial value of the current hour from system clock
- Declare a double, named **grade** that has an initial random value between 0.0 to 4.0. Make sure you use a random number generator to generate this value.
- Create a String, named **`greeting`** that has an initial value of "Hi"
- Create a String, named **`myPawPrint`** that has an initial value of "*your own Pawprint*"

S21 Challenge: Language Basics

Using the variables declared and initialized above, do the following: Where this document says "display" it means output a line to the standard out. Each displayed item is to be on a separate line. **Note:** System.out.println() puts a newline character at the end of the line. However, System.out.printf() must have a "\n" at the end of the string to put a newline at the end, so the next element is displayed on a separate line.

Using char variables **`c1`** and **`c2`** compare them to see if they are the same. If they are the same, display a line that displays the two characters with the string **" and "** between them followed by **" are the same."** If they are not the same display a line that displays the two characters with the string **" and "** between them followed by **" are NOT the same."** Here are examples of what the two different displayed lines would look like:

a and a are the same.
k and p are NOT the same

If **`qualityScore`** is greater than or equal to 0 and less than or equal to 60 display **"The quality is bad."** Otherwise display **"Good quality."**

Declare a variable called **`gasFee`** that is assigned to the **`miles`** divided by the **`milesPerGallon`** `then` times the **`gasPrice`**. The **`gasFee`** variable must be of the same type as the type that results from the calculation. The formula here is: **`gasFee = (miles / milesPerGallon) * gasPrice.`** The output should look like the following (actual value will be different): **Total gas fee = 12.349**

Note: The output of **gasFee** should only be displayed up to **three** decimal places.

Using **`sunny`** and **`warm`** display these specific outputs: **"Go swimming."** if **`sunny`** and **`warm`** are both true and **"Go hiking."** if **`sunny`** is false and **`warm`** is true. For any other possibilities, display a message saying **"Stay home and code."**

Use switch/case and the variable **`hour`** to display **"The current time is <hour> in the MORNING."** if **`hour`** is equal to 5-10, **"The current time is <hour> in the AFTERNOON."** if **`hour`** is equal to 11-16, **"The current time is <hour> in the EVENING."** if **`hour`** is equal to 17-22, **"The current time is <hour> in the NIGHT."** if **`hour`** is equal to 23-4, and **" You have the wrong time."** if **`hour`** is something other than 0-23.

S21 Challenge: Language Basics

Note: Make sure you replace <hour> with the actual hour collected from the system clock.

Note: The hour values will be different since they are dynamic and based on the system clock. Make sure you test your switch statement for all possible values, if there is a bug with end points (numbers you would not normally expect), points will be deducted.

Note: Printing 1 or 2 digits for the hour is widely accepted. However, 1 digit is used on most watches produced in the industry, so 1 digit is required for the output of hour when it is less than 10.

Use whatever conditional statement you prefer to evaluating the variable grade. If **grade** is in [0.00, 0.70) then display **"The student's GPA grade is an F in the class."**, If **grade** is in [0.70, 1.70) then display **"The student's GPA grade is a D- to D+ in the class."** If **grade** is in [1.70, 2.70) then display **"The student's GPA grade is a C- to C+ in the class."** If **grade** is in [2.70, 3.70) then display **"The student's GPA grade is a B- to B+ in the class."** If **grade** is equal to or larger than 3.70 then display **"The student's GPA grade is an A- to A+ in the class."**

Note: Make sure you test the last case here, either by hard coding the grade, or you could also make the grade from 0.0 to 4.9 in the random number generator in order to test the last case here.

Note: Your code should handle the "in between" cases. Therefore, if the grade generated is 2.69867, the grade should still be a C+. There are a few ways to implement this, but check the edge cases and make sure your code works.

Note: Every time you run the program, the output of your conditional statement for the variable grade will be different since you are randomly generating this number using a random number generator.

S21 Challenge: Language Basics

Using a "for" loop count from 2 to 10 (inclusive) using an integer variable **`count`**. Inside the loop display each value of **`count,`** that is divisible by 3, with a line that is **"`Count:` "** followed by the value of **`count`** as in:

`Count`: 3
`Count`: 6
…

Declare an int variable **`countDown`** with an initial value of 10. Using a "while" loop that continues while **`countDown`** is **greater than 0**, display the value of **`countDown`** in a line with **"Count Down:"** followd by the value of **`countDown.`** (Example: **Count Down: 10**) After the countdown line is displayed, decrease the value of **`countDown`** by 1 using the post-decrement operator. When **`countDown`** reaches 0, output the following message: "Houston, we have a lift off!".

Using the invokeMe() method from Challenge 2: Hello World, re-develop the method to display a line that contains the **`greeting`** string, followed by comma and space ", " then the **`myPawPrint`** string, followed by " **and today's date is** ", and last but not least, call the invokeMe() method. Consider the example below:

Hello, my pawprint is wergelesn and today's date is 08/30/2020 5:06PM

Note: You do not need to import the invokeMe() method, just simply copy and paste it into the new program and then refractor the code to accomplish the tasks for this challenge.

Note: The requirements in this document are the minimum requirements for this challenge. Going above and beyond the rest of the class is encourage and often rewarded with bonus points. For this challenge, you can build, implement, and experiment with more of Java's language basics. How? You can create additional states and/or behaviors (fields or methods), implement more calculations, look at more built in functionality in the JDK, compare differences between C and Java (or other languages), and many many more. The work described here is different from the bonus opportunity provided below. There are many ways to get bonus points and if you are ever wondering what is counted or what is not counted, contact Professor Wergeles and ask. Most of the time, if you go above and beyond your colleagues, then bonus points can be awarded. If you ever

implement above and beyond work make sure the grader knows by putting a comment on your submission and if you want to ensure we see it, send an email to the course email. Going above and beyond the course is encouraged and will help you learn the material better, which will help you get an awesome job!

Note: Make sure you get the minimum requirements completed first before you attempt any "bonus" or above and beyond work. Once the minimum requirements are done, submit a copy of your work on Canvas, that way you know you have everything for the minimum requirements submitted before the due date. Then if you have extra time, work on going above and beyond the course for the extra reward. Then you can submit another copy of your work on Canvas. Multiple submissions are allowed and encouraged. The main thing is do not make the scope of this project too large where you cannot get anything done. Work progressively, get the a minimal viable product built, then if you have time, work on bonus work. Otherwise, if you make the scope too large, you may not get anything done. This cannot be used as an excuse later for not completing assignments, you have been warned.

Run your application and make sure everything works as expected. ZIP the project directory and submit it on Canvas following the directions posted on Canvas in Module 1.

Things to Submit on Canvas:

1. The zip file created after you export your project. Follow the tutorial on canvas.
2. Submit screenshots of your application running for proof with the system clock, pawprint, and submitted before the due date. The system clock must contain the full date and time.
    - The more screenshots you take the better. Take screenshots of your application running, the output, and take screenshots of ALL of your code. You can combine these to reduce the number of screenshots submitted.
    - This may help you in the event that something goes wrong. You will be surprised on how many times this helped students and how many times grades were penalized because students did not have adequate proof due to lack of screenshots.

S21 Challenge: Language Basics

Bonus Opportunity:

- Instead of putting everything in the main method, create a class and use proper object-oriented programming techniques and practices.
- Putting everything in the main is the minimum requirements and should be done first to make sure you have the challenge completed by the due date and time. However, if you have the minimum requirements completed, then you can give the bonus opportunity a try.
    - I recommend you actually submit the minimum requirements on Canvas, then try the bonus opportunity, if you get it completed, just do another submission on Canvas. On your end, it will look like you only submitted the current version, but on our end, we will see all submissions.
- You must have getters and setters for each of the fields using the "this" keyword, then how you create the rest of the class is up to you.
    - You could have a constructor to set some or all of the fields.
    - You could have a no-arg constructor and set the fields with the setter methods.
    - You could have methods for each of the calculations in the requirements.
    - You could have a method to get the current date from the system clock.
    - You could have one large print method for all of the fields, or you could have several print methods for each individual print statement in the requirements.
    - Then you create the object in the main, set the states (set the fields/variables using the constructor or setter methods), invoke the behaviors (call the methods to perform the calculations and conditional statements), and then invoke the print method(s) to show the results (call the print methods to display the results to the console as described in the requirements).
    - These are just ideas and possible recommendations, how you set it up is up to you, just use proper programming practices discussed during class.
    - If you have questions or would like clarification on what is allowed and what is not allowed, or what will count and what will not count, contact Professor Wergeles by showing up to office hours or the developer night and have a quick discussion on the ideas you want to implement.
- Do not make this too difficult. It could be easy to make this complicated and create too many classes. Start out simple, then if you have time, you can make the code better and cleaner at each version.
- Submit the bonus opportunity the same as the regular assignment, then let the TAs know you completed the bonus by leaving a comment on Canvas. To ensure we know you implemented bonus work, you can also send an email to the course email which is the official channel of communication. However, TAs tend to like the comments on Canvas which is easy to see while grading.