Homework 01 - Age Calculator

Problem Description

You've now been introduced writing a program in Java. This means you can write simple, but useful programs! Given your newfound skills, you've been hired by a company to calculate the age of customers! You've been provided with a Java class, `AgeCalculator` that you need to fill in to the appropriate specifications.

Solution Description

For this assignment, you won't be expected to write your own class. Instead, we'll provide the class and you simply have to fill in the `main` method. Keep in mind that what you print out should match the defined output *exactly*. Deviation from what the output should be, even by a character or two, will result in lost points.

To start, save the following code in a file called `AgeCalculator.java`:

```
public class AgeCalculator {
    public static void main(String[] args) {
        // Do not modify the following line.
        int birthYear = Integer.parseInt(args[0]);
        // Part 1:

        // Part 2:

    }
}
```

Part 1: Calculate the age

Inside of the main method, create a variable called `age` that can hold integers. Assign to `age` the age of the subject. Assume that someone's age is the difference between 2019 and their birth year.

Part 2: Report the age

Once you calculate the age, it's time to print it out. Output should be in the form:

`This person is [age] years old!`

Testing your app

In order to test your program, you can run `java AgeCalculator [birth year]` with some example year. For example,

`java AgeCalculator 1998` prints out `This person is 21 years old!`

`java AgeCalculator 2001` prints out `This person is 18 years old!` ____

Allowed Imports

To prevent trivialization of the assignment, you are *not* allowed to import any classes or packages.

Feature Restrictions

There are a few features and methods in Java that overly simplify the concepts we are trying to teach. For that reason, do not use any of the following in your final submission:

- var (the reserved keyword)
- System.exit

Collaboration

Collaboration Statement

To ensure that you acknowledge a collaboration and give credit where credit is due, **we require that you place a collaboration statement as a comment at the top of at least one java file that you submit**. That collaboration statement should say either:

I worked on the homework assignment alone, using only course materials.

or

In order to help learn course concepts, I worked on the homework with [give the names of the people you worked with], discussed homework topics and issues with [provide names of people], and/or consulted related material that can be found at [cite any other materials not provided as course materials for CS 1331 that assisted your learning].

Recall that comments are special lines in Java that begin with `//`.

Turn-In Procedure

Submission

To submit, upload the files listed below to the corresponding assignment on Gradescope:

- AgeCalculator.java

Make sure you see the message stating "HW01 submitted successfully". From this point, Gradescope will run a basic autograder on your submission as discussed in the next section.

You can submit as many times as you want before the deadline, so feel free to resubmit as you make substantial progress on the homework. We will only grade your last submission: be sure to **submit every file each time you resubmit**.

Gradescope Autograder

For each submission, you will be able to see the results of a few basic test cases on your code. Each test typically corresponds to a rubric item, and the score returned represents the performance of your code on those rubric items only. If you fail a test, you can look at the output to determine what went wrong and resubmit once you have fixed the issue.

The Gradescope tests serve two main purposes:

1) Prevent upload mistakes (e.g. forgetting checkstyle, non-compiling code)
2) Provide basic formatting and usage validation

In other words, the test cases on Gradescope are by no means comprehensive. Be sure to thoroughly test your code by considering edge cases and writing your own test files. You also should avoid using Gradescope to compile, run, or checkstyle your code; you can do that locally on your machine.

Other portions of your assignment can also be graded by a TA once the submission deadline has passed, so the output on Gradescope may not necessarily reflect your grade for the assignment.

Important Notes (Don't Skip)

- Non-compiling files will receive a 0 for all associated rubric items
- Do not submit `.class` files.
- Test your code in addition to the basic checks on Gradescope
- Submit every file each time you resubmit
- Read the "Allowed Imports" and "Restricted Features" to avoid losing points
- Check on Piazza for all official clarifications