LEARNING PROFILE FOR ASSIGNMENT#1, QUESTION#10

TaxReturn.java

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Name* | *:* | *Tyler Lucas* | *Due Date* | *:* | *N/A* |
| *Student ID* | *:* | *3305203* | *Submission Date* | *:* | *2017/05/09* |

# 1. Problem Statement

Modify the following program to the specifications given below:

1. Add a new status – SingleParent – where the tax is computed as a SINGLE but with a further reduction of $5000 per child.
2. Add a new tax condition – if the income is greater than $249,999 for SINGLE, then add a tax of 25% on income amount above $150,000; if the income is greater than $349,999 for MARRIED, then add a tax of 35% on income amount above $200,000.
3. Unknown status – if the status doesn’t belong to SINGLE or MARRIED or SINGLE\_PARENT, then compute a 33% tax on the income.

import java.util.Scanner;

public class TaxReturn

{

/\*\*

Constructs a TaxReturn object for a given income and

marital status, and computes the tax.

@param anIncome the taxpayer income

@param aStatus either SINGLE or MARRIED

\*/

public TaxReturn(double anIncome, int aStatus)

{

income = anIncome;

status = aStatus;

}

public double getTax()

{

double tax = 0;

if (status == SINGLE)

{

if (income <= SINGLE\_BRACKET1)

tax = RATE1 \* income;

else if (income <= SINGLE\_BRACKET2)

tax = RATE1 \* SINGLE\_BRACKET1

+ RATE2 \* (income - SINGLE\_BRACKET1);

else

tax = RATE1 \* SINGLE\_BRACKET1

+ RATE2 \* (SINGLE\_BRACKET2 - SINGLE\_BRACKET1)

+ RATE3 \* (income - SINGLE\_BRACKET2);

}

else

{

if (income <= MARRIED\_BRACKET1)

tax = RATE1 \* income;

else if (income <= MARRIED\_BRACKET2)

tax = RATE1 \* MARRIED\_BRACKET1

+ RATE2 \* (income - MARRIED\_BRACKET1);

else

tax = RATE1 \* MARRIED\_BRACKET1

+ RATE2 \* (MARRIED\_BRACKET2 - MARRIED\_BRACKET1)

+ RATE3 \* (income - MARRIED\_BRACKET2);

}

return tax;

}

public static final int SINGLE = 1;

public static final int MARRIED = 2;

private static final double RATE1 = 0.15;

private static final double RATE2 = 0.28;

private static final double RATE3 = 0.31;

private static final double SINGLE\_BRACKET1 = 21450;

private static final double SINGLE\_BRACKET2 = 51900;

private static final double MARRIED\_BRACKET1 = 35800;

private static final double MARRIED\_BRACKET2 = 86500;

private double income;

private int status;

public static void main(String[] args)

{

Scanner in = new Scanner(System.in);

System.out.print("Please enter your income: ");

double income = in.nextDouble();

System.out.print("Enter S (single) or M (married): ");

String input = in.next();

int status = 0;

if (input.equalsIgnoreCase("S"))

status = TaxReturn.SINGLE;

else if (input.equalsIgnoreCase("M"))

status = TaxReturn.MARRIED;

else

{

System.out.println("Bad input.");

return;

}

TaxReturn aTaxReturn = new TaxReturn(income, status);

System.out.println("The tax is " + aTaxReturn.getTax());

}

}

# 2. Description of the Code

Outputs temperature in Fahrenheit given an input temperature in Celsius.

# **3.** Errors and Warnings

Table 1: List of Errors and Warnings Encountered in the Program

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Errors / Warnings** | **Details** | **How I solved them** |
| 1 | CelsiusToFahrenheit class wasn’t found in CelsiusToFahrenheit project. | [v. 1.0] I had set the main class as "private". | I changed the class and main method from “private” to “public”. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 4. Sample Input and Output

[Version 1.2-1.3, input “0”]

Enter temperature in integer degrees Celsius: 0

0 degrees Celsius is equivalent to 32 degrees Fahrenheit.

[Version 1.2-1.3, input “60.5” (float type input)]

Enter temperature in integer degrees Celsius: 60.5

Exception in thread "main" java.util.InputMismatchException

at java.util.Scanner.throwFor(Scanner.java:864)

at java.util.Scanner.next(Scanner.java:1485)

at java.util.Scanner.nextInt(Scanner.java:2117)

at java.util.Scanner.nextInt(Scanner.java:2076)

at CelsiusToFahrenheit.main(CelsiusToFahrenheit.java:34)

C:\Users\tyblu\Documents\repos\comp268-random\CelsiusToFahrenheit\nbproject\build-impl.xml:1040: The following error occurred while executing this line:

C:\Users\tyblu\Documents\repos\comp268-random\CelsiusToFahrenheit\nbproject\build-impl.xml:805: Java returned: 1

# 5. Discussion

The first error, where a class couldn’t be found in the project, was caused by setting either or both the class and the main method to private. I first ran into this error when attempting the *HelloWorld* sample program[[1]](#footnote-1). Having read the class Style Guide in which it says “Create private fields with getters/setters rather than leaving fields public,” as well as Controlling Access to Members of a Class[[2]](#footnote-2) in which it says “Use private unless you have a good reason not to,” I mistakenly thought this applied to the main class and method as well, as I’m still not sure what the differences are between a class, method, and object. Searching online didn’t reveal a solution right away, as few experienced programmers would think that something this simple could go awry, but I eventually found the answer here: <https://goo.gl/P2OdMJ>. Of course, the next page in the textbook had the answer as well:

The word “public” in the first line of main() means that this routine can be called from outside the program. This is essential because the main() routine is called by the Java interpreter, which is something external to the program itself.[[3]](#footnote-3)

I’ll have to keep an eye out to

1. (Eck, 2014, p. 21) [↑](#footnote-ref-1)
2. (Oracle, 2015) [↑](#footnote-ref-2)
3. (Eck, 2014, p. 22) [↑](#footnote-ref-3)