Student Name:

**Project 7: Computing Income Tax with Arrays**

CSC 201-M6

Problem Description:

The United States federal personal income tax is calculated based on filing status and taxable income. There are three filing statuses: single filers, married joint filers, and head of household filers. The tax rates vary every year. Table below shows the expected rates for the year 2013.

|  |  |  |  |
| --- | --- | --- | --- |
| Rate | Single Filers | Married Joint Filers | Head of Household Filers |
| 10% | $0 to $8,925 | $0 to $17,850 | $0 to $12,750 |
| 15% | $8,925 to $36,250 | $17,850 to $72,500 | $12,750 to $48,600 |
| 25% | $36,250 to $87,850 | $72,500 to $146,400 | $48,600 to $125,450 |
| 28% | $87,850 to $183,250 | $146,400 to $223,050 | $125,450 to $203,150 |
| 33% | $183,250 to $398,350 | $223,050 to $398,350 | $203,150 to $398,350 |
| 35% | $398,350 to $400,000 | $398,350 to $450,000 | $398,350 to $425,000 |
| 39.6% | $400,000 and up | $450,000 and up | $425,000 and up |

Specifications:

You are to write a program to compute personal income tax. Your program will need to do the following:

1. Create and initialize a 7 row, 4 column array.
   1. Column 1 is the Tax Rate for the Tier.
   2. Column 2 is the maximum income for the Tier for Single Filers.
   3. Column 3 is the maximum income for the Tier for Married Joint Filers.
   4. Column 4 is the maximum income for the Tier for Head of Household Filers.
2. You may prepopulate the values in Columns 2 – 4.
3. Ask the user to enter the Tax Rate for each of the 7 tiers.
4. Ask the user for a Filing Status and a Taxable Amount.
5. Search the array to find the appropriate starting point for Tax Calculation.
6. Calculate the Tax.
7. Display the tax using the following template:

The tax on {Taxable Amount} for a {Single|Married|Head of Household} filer is {Tax}.

{Taxable Amount} and {Tax} are to be displayed as currency and have 2 decimal places.

1. Steps 4 – 7 are to be repeated until the user indicates they wish to stop by entering -1 for the Filing Status

Analysis:

(Describe the purpose, processing, input and output in your own words.)

Purpose of the project is to let the user calculate tax based on taxable income and filing status.

**Design:**

**(Describe the major steps for solving the problem.)**

The major steps as implemented are as below:

1. Create a default tax table using 2 – dimensional array
2. Allow users to modify tax rates
3. Get filing status from the user using a Scanner class object
4. Get the taxable income from user using a Scanner class object
5. Create a helper function to determine where to stop and start tax calculation
6. Create a method with proper algorithm to calculate tax
7. Print output to the console

I always start coding with the most basic approach: ignore user inputs!

Initially, I developed algorithm for single filing status and single tax group.

int status=1;

At this point I do not implement any error validation or user input.

Next I target only one column to use tax rates properly.

My first trial methods are as below; they have bugs but show the thought process.

You could see that I am not using all of the columns in the 2D array, just trying to finalize my code for the Single filing status group.

public static int incomeBracket(double yourIncome)

{

//initialize tax

double tax=0;

//find tax bracket

int count=0;

int status=1;

for (int i=0; i<taxBrackets.length; i++)

{

if (yourIncome>taxBrackets[i][status])

{

count++ ;

}

}

return count;

}

public static double calculateTax(double yourIncome)

{

//get the incomebracket first

double tax=0;

double status=1;

int incomeTaxBracket;

incomeTaxBracket=incomeBracket(yourIncome);

if (incomeTaxBracket==0)

{

//System.out.println(yourIncome);

//System.out.println(taxBrackets[0][0]);

tax = tax+yourIncome\*taxBrackets[0][0]/100;

return tax;

}

else

{

for (int i=1; i<= incomeTaxBracket-1; i++)

{

tax = tax+(((taxBrackets[i][1]-taxBrackets[i-1][1])\*taxBrackets[i][0])/100);

System.out.println(tax+"1");

}

tax=tax+taxBrackets[0][1]\*taxBrackets[0][0]/100;

tax=tax+((yourIncome-taxBrackets[0][incomeTaxBracket])\*taxBrackets[incomeTaxBracket][0])/100;

System.out.println(tax+"2");

return tax;

}

}

This phase includes several runs to check for outputs which were compared against output from previously coded Project2.

Once the algorithm is finalized and verified, I have implemented error validations for various inputs.

The algorithm development was nearly completed in class where I first decided to use a helper class to figure out where to stop tax calculations. Once this class is defined, the major algorithm development became much easier. I have tried to follow the divide and conquer concept of program development with project 7.

Only when I am completely happy with the program outputs, I have implemented code for the user to edit default tax rates.

Lastly, I have improved the output and formatted numbers according to specifications.

Also, please note that most program development steps are also documented as comments inside the program.

**Please go to the next page.**

**Testing: (Describe how you test this program)**

I started testing the program in very early stage.

A very early stage development output looked like this (incomplete, wrong result):

50000

Please enter your taxable amount

50000

50000.0

Exception occurred.

Project7.main({ });

10.0

Please enter your filing status

1 = Single Filers,

2 = Married Joint Filers,

3 = Head of Household

1

Please enter your taxable amount

50000

50000.0

-----------------

13028.75

As I caught error in my implementation, I kept updating the code till my outputs matched with Project2 outputs.

Before submitting the code, I realized that once the used inputs the tax rates, they do not go back to default rates. So, before starting the next tax calculation fresh I have implement a method that resets all of the tax rates to the default values.

Several output files are also attached as .txt file for reference.

How to submit your assignment

1. Login Blackboard
2. Click on Assignments on the left
3. Click on Week 10 Work folder
4. Read the instruction there and submit the following items:

* Your jar file with source code. The jar file without the source code will not be graded. Please use the steps given on Project 1 Instructions to create your jar file. Rename your jar file as YourName\_Project2. Suppose your name is Susan Boyd, you should rename your jar file as SusanBoyd\_Project2. Files with wrong name will not be graded.
* This document with answers for analysis, design and testing. Rename this document as Project2\_Yourname. Suppose your name is Susan Boyd, you should rename this document as Project2\_SusanBoyd. Files with wrong name will not be graded. This document is worth 10 points and the comments in your program is worth 10 points. Working code is worth 30 points.