CSCE 120: Learning To Code

Making Decisions
Hacktivity 4.2

Introduction

Prior to engaging in this hacktivity, you should have completed all of the pre-class activities as outlined in this module. At the start of class, you will be randomly assigned a partner to work with on the entirety of this hacktivity as a *peer programming* activity. Your instructor will inform you of your partner for this class.

One of you will be the driver and the other will take on the navigator role. Recall that a driver is in charge of the keyboard and computer while the navigator is in charge of the handout and directing the activity. However, you are *both* responsible for contributing and discussing solutions. If you were a driver/navigator in the prior activity, switch roles for this activity.

1 BOGO Sale

Let's warm up by writing a program to compute a total sales receipt for a BOGO Sale in which if a person buys one item at full price, they get a second item at half price. However, the discount is usually only applied to the *cheaper* item.

- Write code to prompt the user for two input values (the cost of each item).
- Write code to determine which value is less and apply a 50% discount to it and compute the total
- Display the total (as well as the amount of savings) to the user with a popup message
- Test your program with several values

2 Life & Taxes

In this exercise, you'll build a small app to help estimate tax payments. The 2014 tax brackets for a married couple filing jointly is presented in Table 1 (which uses the Adjusted Gross Income, AGI).

AGI is over	But not over	Tax
–	$18,150	10% of the AGI
$18,150	$73,800	$1,815 plus 15% of the AGI in excess of $18,150
$73,800	$148,850	$10,162.50 plus 25% of the AGI in excess of $73,800
$148,850	$225,850	$28,925 plus 28% of the AGI in excess of $148,850
$225,850	$405,100	$50,765 plus 33% of the AGI in excess of $225,850
$405,100	$457,600	$109,587.50 plus 35% of the AGI in excess of $405,100
$457,600	—	$127,962.50 plus 39.6% of the AGI in excess of $457,600

Table 1: 2014 Tax Brackets for Married Couples Filing Jointly

In addition, tax payers can take a tax credit (that offsets their tax liability) if they have qualifying children. The rules are as follows:

- If the AGI is $110,000 or more, they cannot claim a credit (the credit is $0)
- Each child is worth $1,000 credit, however at most $3,000 can be claimed
- The credit is not refundable: if the credit results in a negative tax liability, the tax liability is simply $0

As an example: suppose that a couple has $35,000 AGI and has two children. Their tax liability is

$$\$1,815 + 0.15 \times (\$35,000 - \$18,150) = \$4,342.50$$

However, the two children represent a $2,000 refund, so their total tax liability would be $2,342.50.

1. Download the Tax App from GitHub at the following URL: https://github.com/cbourke/TaxApp. We have provided a webpage interface (`taxes.html`) and incomplete JavaScript source file (`taxes.js`).
2. Evaluate the HTML page and try it out: it doesn't work (the result is always zero)
3. Open the JavaScript source file. We have provided code that pulls the AGI income and number of children from the webform and passes them to the function

`calculateTax(agiIncome, numChildren)`. Inside this function, you will need to implement logic to find the tax liability given these two values and store the result in the `totalTax` variable.

4. Run your code on the test cases in Table 2 and debug your code as necessary.

AGI	Number of Children	Tax Liability
$23,552.50	2	$625.30
$23,552.50	4	$0.00
$96,250.32	5	$12,775.08
$250,000.00	1	$58,734.50
$750,000.00	2	$243,752.90

Table 2: Test Cases for the Tax App

3 Reverse Engineering

When learning a new concept or language it is often easiest to look at an existing example and adapt it. This process sometimes involves *reverse engineering*: looking at technology or code, figuring out how it works and what each part does, and then adapting it to your own needs.

In this exercise you will practice some reverse engineering with your partner. Examine the code in the previous exercise with your partner and deduce the following.

1. Which line(s) of code were responsible for pulling data from the HTML webform?
2. How do those lines relate to elements in the HTML page (how was it that each piece of data was *identified*)?
3. Which line(s) in the HTML page were responsible for performing an action when the user clicked the button?
4. When the result is displayed to the user, new HTML elements are created and animated to "fade in." How is this accomplished?

3.1 Take Home

Go ahead and get yourself signed off for this hacktivity if you have completed up to this point. However, if you have time (or if you prefer, you can use this as a take home activity), complete the following grader program using what you learned from your reverse engineering.

A letter grade can be awarded based on the scale in Table 3. We have provided an HTML and JavaScript file, but even less code has been provided to you. Not only do you need

to implement the proper conditional statements, but you need to bring in the data from the web form and display the data in the webpage.

Percentage	Letter Grade
90 – 100%	A
80 – 89%	B
70 – 79%	C
$< 70\%$	F

Table 3: Grade Scale