COMP 10-IDI

Assignment 10: Forms, Input/Output

Due Tuesday Apr 7 at 11:00PM

---

1. Two Problems, Two Versions Each

For this project you will design and program pages that read user input, process that information, and present user output. You will use two types of input/output: pop-up boxes and form elements. There are two parts to this assignment.

☞ **Working Directory**: Do your work in a directory called hw10 in your website. It must be called hw10 in lower case letters.

2. Part I: Flash Cards

For this problem, you will write an arithmetic quiz program based on a starting framework. First, copy this file:

```
cp /comp/10IDI/files/hw10/* .
```

This command copies flashcards.html into your directory. Try the page, then read the source code.

Explanation: Random Numbers

Flashcards.html shows how to use random numbers in a javascript program. The dave-reed.com site has some javascript that has a function called RandomInt that gives you a random integer in a range of values.

2.1. Version P: Pop-Ups

Copy flashcards.html to a new file called flash-popup.html and modify that copy to do this part.

For version P of the flashcards program, add code to this page so it tells the user if the answer is correct or not. If the input is correct, the program adds 2 points to the score, congratulates the user, and reports the user's score.

If the answer is incorrect, the program tells the user whether the input is too high or too low and then gives the user another chance. If the second try is correct, the program adds 1 to the score, congratulates the user, and reports the score. The message should read something like: *Correct! Your score is 7/12*.

If the user's second input is not correct, the program tells the user the correct answer and adds nothing to the score.

In addition, put two new buttons on the page, one marked *new game*, and one marked *score*. When the user presses the *new game* button, the program uses a confirm pop-up to ask: "Start over? Are you sure?". If the user presses the Ok button on the confirm pop-up, the program resets the score.

When a user presses the *score* button, the program uses an alert pop-up to report the current score without changing it.

Make sure this page is called flash-popup.html and be sure to include comments and clear indenting.

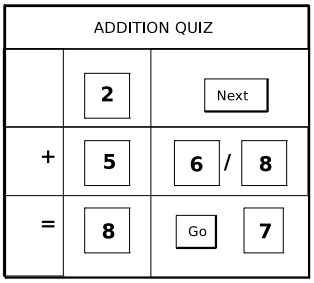
Explanation: confirm()

We use the `alert()` function to tell the user something, and we use `prompt()` to ask the user for input. Javascript has one more pop-up window: `confirm()`. The `confirm()` function is designed to ask users yes/no questions such as "Are you sure you want to quit?" . The `confirm()` function takes a question as an argument and returns true or false depending on which button the user presses. For example:

```
if ( confirm("Are you sure?") == true ){
    total = 0;
}
```

3. Version F: Form Input/Output

Make a copy of flashcards.html called flash-form.html and add code to make the page look like:



This version displays an addition problem in the top two rows of the second column (the ones showing 2 and 5) and expects the user to type in an answer in the box next to the equals sign. The lower right box (the one with 7) is the place the program displays the correct answer if the user does not get the correct answer. The middle row displays the current score. The first number is the number of correct answers, and the second number is the total number of questions asked.

This version does not give second chances. When the user presses the Next button, the program fills in the two numbers in the addition problem and clears the two boxes on the bottom row.

The user then enters an answer in the box next to the

equals sign and then presses the Go button. The program checks the user input. If the user input is correct, the program puts an "!" in the lower right box. If the user input is not correct, the program puts the correct answer in the lower right box. In both cases, the program updates the score.

4. Part II: Snack Shop

For this project you will write a webpage to act as the computerized cash register in a doughnut shop. This time, you will write a single program that uses form elements as well as pop-up windows.

4.1. Details

Make a file called snackshop.html that presents:

This digital cash register computes individual sales as the day goes on, and the register also keeps track of the total sales for the day.

At the start of the day, the user presses the 'Clear' button to reset the total of all sales to zero. When that button is pressed, the program uses `confirm` to ask if the user really wants to "Clear register total?" . If the user confirms the question, the day total is set to zero.

When a new customer arrives, the user presses the 'New' button. This clears all the boxes shown on the screen. The user can then enter numbers in the boxes for the number of coffees and the number of doughnuts. The other boxes must be made "readonly"; Just include the word `readonly` as an attribute in the element. When the user presses the 'Done' button, the program computes and displays the cost of the coffees, the cost of the doughnuts, and then the total price including 5% tax.

At any time, the user can press the "Sales" button to see the total sales for the day. The program uses an alert window to display the total sales for the day.

5. Turning in Your Work

Turn in your work using Moodle and also leave your work on your website in the hw10 directory.

To turn in your work using Moodle, you must turn in plain text html files. To do so:

a. Copy with ssh or fugu the files to your desktop
b. Submit those files using moodle

6. Extra Credit

Humans sometimes enter bad data. For two points of extra credit on each part, you can include code to identify non-numeric input and inform the user and give the user another chance to enter valid data. There are two ways to do this. One way is to use a `while` loop described in chapter 13 in the textbook. Another way is to write a function that calls itself (a *recursive* function.)