111 Makeup Project

1700 M 3/16 (FC)
1700 F 3/13 (XC)

Michael Hennessy, Computer & Information Science, UO
How the Makeup Project Works

The makeup project can be used to replace any score on projects 1-4, but not project 5.

If you already have full credit on projects 1-4, then there is nothing to be gained from doing the makeup project. Do not do this project if you already have full credit on p1-p4.

The makeup project is *optional* and *not* required. It can not be turned in for extra credit.

The sole purpose of the makeup project is to allow you to replace a single unsatisfactory score on projects 1-4.

You will find this project easier to complete if you first complete Project 5. Therefore, do Project 5 before working on this project.
Due: 1700 M 3/16 (FC)
1700 F 3/13 (XC)

Be sure you understand how the makeup project works, as explained on the previous page of this document.

This project is due no later than 5pm 3/16. It may be difficult to get project help on the due-date, as help hours on the due-date have long lines. Finish early, and do not wait until the last day.

This project may be turned in by 5pm Friday 3/13 for +5 XC points for Speedy Delivery.

1. [40 pts] 111/mu/dataStructures.js.

All of the following Command-Line JavaScript exercises. None of these functions will use prompt, alert or getElementById. You do not need to create a web page (.html) for this part of the project. Store the code for each exercise A-G in dataStructures.js in your mu folder.

A) Write a function arrMax that accepts an array of numbers as an argument and returns the largest number in the array.

arrMax([3, 7, 2, 6, 6, -100]) => 7

B) Write a function allOdd that accepts an array of numbers and returns true if every number is odd, false otherwise.

allOdd([2, 3, 5, 7]) => false
allOdd([281, 111, 381, 383]) => true
C) Write a function maxString that accepts an array of strings returns the longest string in the array.

maxString(["the", "albatross", "just", "ate", "your", "lunch"])) => "albatross"

D) Write a function maxWord that accepts a string representing a sentence, and returns the longest word in the sentence. You may use the String split method and the Array sort method to solve this problem, but are not required to do so.

maxWord("The albatross just ate your lunch") => "albatross"

E) Write a function maxWordLength that accepts a string representing a sentence, and returns the length of the longest word in the sentence.

maxWordLength("The albatross just ate your lunch") => 9

2. [60 pts] 111/mu/wilsons.html and wilsons.js.

United States currency notes now in production bear the following portraits: George Washington on the $1 bill, Thomas Jefferson on the $2 bill, Abraham Lincoln on the $5 bill, Alexander Hamilton on the $10 bill, Andrew Jackson on the $20 bill, Ulysses S. Grant on the $50 bill, and Benjamin Franklin on the $100 bill.

There are also several denominations of currency notes that are no longer produced. These include the $500 bill with the portrait of William McKinley, the $1,000 bill with a portrait of Grover Cleveland, the $5,000 bill with a portrait of James Madison, the $10,000 bill with a portrait of Salmon P. Chase, and the $100,000 currency note bearing a portrait of Woodrow Wilson.

A) Command-Line JavaScript: Write and test a series of simple functions named numOfWilsons, numOfChases, numOfMadisons, and so on, down to numOfNickels (numOfPennies is optional).

Each function accepts an argument (an amount in cents) and returns the corresponding number of bills or coins, so each is a classic, lego-block function.

numWilsons(5954000) => displays 5

After the above functions are written and tested, write a function named verbalizeCash that accepts an value in dollars and cents and returns a string showing the correct number of bills and coins:

verbalizeAmt(595.40) => "$595.40 = 5 Franklins 4 Jacksons 1
Hamilton 1 Lincoln 1 Quarter 1 Dime 1 Nickel

Note that 0 entities are not included, and single units are displayed as "1 Dime" and not as "1 Dimes".

B) client-side JavaScript.

When you have finished all the above functions, then create a web page named wilsons.html, that includes an image of the two sides of a $100K bill, a textbox, and a click a button. When clicked, the string returned by verbalizeCash is inserted into the web page.

All JavaScript goes in the .js file. No event handlers (onclick, onchange, onmouseover, etc.) will be in the HTML file.