Exercise 5: Getting started with Git

The aim of this lab is to familiarise yourself with Git and learn how to use it for projects.

The Basics of Git

Run through this basic introduction from Code School.

To become an expert at using GIT, run through the full tutorial from Atlassian.

GitHub

GitHub is an online application used for hosting GIT repositories. It can be used to store your code, collaborate with others, manage version control, and many other things.

Set up a GitHub account (if you haven't got one already). You can get a student account if you want to, although as of a few years ago GitHub allows anyone to make free, private repositories.

For practise, work with a partner to create a joint GitHub project. Link the project with your individual development environments, then try a series of commits, pushes, checkouts (forks) and merges, and then undo them.

Try opening a Pull Request on GitHub and get your partner to provide a code review for it. Push some changes addressing their feedback and then merge the PR in.

Integrate Git with VSCode

Run through this tutorial on using Git with VSCode.

Study break Challenge

Creating a Text-based Calculator

Create a new web page, calculator.html, based on your template page. Add a text-based input field, and a "Calculate" button. When the button is pressed, JavaScript (and jQuery) should parse the equation given in the input field and create a p element with the equation and the solution. You may assume for now that the formula contains at most three operations.

Note: Be sure to add a description of how to use this calculator!

You will need to use the regular expression methods to parse, or break down, the string into its components, and type conversion to extract the numbers. When doing the parsing, remember to respect the precedence of the arithmetic operators. For example, if the user enters:

15 + 4 x 3 should give the answer 27, not 57.

Note: Be sure to ignore any whitespaces.

A Stack-based Calculator

Repeat the above task for arbitrary length formulae using an array as a Stack data structure (that is, using the push and pop methods). See here if you are not familiar with stacks.

You will need three stacks:

1. Input stack; to hold the parts of the input as it is parsed
2. Pre-Order stack; to hold the components of the formula in preorder format
3. Answer stack; to accumulate the answer

You may assume that only natural numbers and no parentheses will be entered.

The algorithm proceeds as follows:

1. Pop the top item off the input stack. If it is a number push it onto the preorder stack. Otherwise find the first occurrence of the lowest precedence operator in the string, slice the string and push its two arguments onto the input stack (second argument first), and push the operator onto the preorder stack. Continue until the input stack is empty. For example:

input stack	preorder stack
1x2+3x4+5	empty
3x4+5, 1x2	+
3x4+5, 2, 1	+, x
3x4+5, 2	+, x, 1
3x4+5	+, x, 1, 2
5, 3x4	+, x, 1, 2, +
...	
empty	+, x, 1, 2, +, x, 3, 4, 5

2. Pop the top item off the preorder stack. If it is a number, push it onto the answer stack. If it is a numeric operator, pop two items off the answer stack, perform the operation, and push the answer onto the answer stack. Continue until the preorder stack is empty. For example:

preorder stack	answer stack
+, x, 1, 2, +, x, 3, 4, 5	empty
+, x, 1, 2, +, x, 3, 4	5
+, x, 1, 2, +, x, 3	5, 4
+, x, 1, 2, +, x	5, 4, 3
+, x, 1, 2, +	5, 12
+, x, 1, 2	17
...	
empty	19

To help debugging, write code that prints the contents of the stacks out in the browser window at each step, like the examples above.

Challenges:

- Add functionality that allows parenthesis to be parsed
- Allow negative real numbers to be given
- Support indices