Основы JavaScript.

Лабораторная работа № 6.

ЦЕЛЬ РАБОТЫ

Реализация калькулятора на основе JavaScript. Приобретение навыков доработки кода и использования готовых функций.

ПРОДОЛЖИТЕЛЬНОСТЬ

2 академических часа (1 занятие)

РЕЗУЛЬТАТ РАБОТЫ

Размещенный на Веб‐сервере и доступный по протоколу *http* документ рабочей версией калькулятора.

ДОПОЛНИТЕЛЬНЫЕ ТРЕБОВАНИЯ К РАБОТЕ

Результат лабораторной работы – веб-страница на которой представлены результаты работы JavaScript, в виде рабочего калькулятора.

В лабораторной работе дан шаблон, который необходимо доработать. Нужно написать стили, чтобы страница смотрелась "красиво", а также дописать недостающие части скрипта, чтобы калькулятор заработал. Для этого можно назначить обработчики событий клика на кнопки калькулятора, а также реализовать функцию evaluate, которая вычисляет значение арифметического выражения, записанного в обратной польской нотации (в файле со скриптом есть подробные пояснения). Калькулятор должен вычислять правильные арифметические выражения, состоящие из чисел, операторов +, -, \*, /, а также скобок. Проверку на правильность введённого пользователем выражения производить не нужно.

РЕКОМЕНДАЦИИ К СТРУКТУРЕ ПРОГРАММЫ

Функция evaluate принимает один аргумент – строку с арифметическим выражением, записанным в обратной польской нотации. Возвращаемое значение – результат вычисления выражения. Выражение может включать действительные числа и операторы +, -, \*, /.

Для самой функции необходимо использовать обратную польскую запись – форму записи математических и логических выражений, в которой операнды расположены перед знаками операций. Алгоритм данной записи следующий:

* Пока есть ещё символы для чтения:
  + Читаем очередной символ.
  + Если символ является числом или постфиксной функцией (например, ! — факториал), добавляем его к выходной строке.
  + Если символ является префиксной функцией (например, sin — синус), помещаем его в стек.
  + Если символ является открывающей скобкой, помещаем его в стек.
  + Если символ является закрывающей скобкой:
    - До тех пор, пока верхним элементом стека не станет открывающая скобка, выталкиваем элементы из стека в выходную строку. При этом открывающая скобка удаляется из стека, но в выходную строку не добавляется. Если стек закончился раньше, чем мы встретили открывающую скобку, это означает, что в выражении либо неверно поставлен разделитель, либо не согласованы скобки.
    - Если существуют разные виды скобок, появление непарной скобки также свидетельствует об ошибке. Если какие-то скобки одновременно являются функциями (например, [x] — целая часть), добавляем к выходной строке символ этой функции.
  + Если символ является бинарной операцией о1, тогда:
    - пока на вершине стека префиксная функция…
      * …ИЛИ операция на вершине стека приоритетнее или такого же уровня приоритета как o1
      * … ИЛИ операция на вершине стека левоассоциативная с приоритетом как у o1
      * … выталкиваем верхний элемент стека в выходную строку;
    - помещаем операцию o1 в стек.
* Когда входная строка закончилась, выталкиваем все символы из стека в выходную строку. В стеке должны были остаться только символы операций; если это не так, значит в выражении не согласованы скобки.

Функция clickHandler предназначена для обработки событий клика по кнопкам калькулятора. По нажатию на кнопки с классами digit, operation и bracket на экране (элемент с классом screen) должны появляться соответствующие нажатой кнопке символы. По нажатию на кнопку с классом clear содержимое экрана должно очищаться. По нажатию на кнопку с классом result на экране должен появиться результат вычисления введённого выражения с точностью до двух знаков после десятичного разделителя (точки).

Эту функцию можно реализовать с помощью механизма делегирования событий. Идея в том, что если у нас есть много элементов, события на которых нужно обрабатывать похожим образом, то вместо того, чтобы назначать обработчик каждому, мы ставим один обработчик на их общего предка.

СПРАВОЧНАЯ ИНФОРМАЦИЯ

|  |  |
| --- | --- |
| Браузер: документ, события, интерфейсы | <https://learn.javascript.ru/browser-environment> |
| Введение в DOM | <https://developer.mozilla.org/ru/docs/DOM/DOM_Reference/%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5> |
| DOM Standard | <https://dom.spec.whatwg.org/> |
| Теневой DOM | <https://learn.javascript.ru/shadow-dom> |
| Делегирование событий | <https://learn.javascript.ru/event-delegation> |
| Алгоритм сортировочной станции | <https://ru.wikipedia.org/wiki/Алгоритм_сортировочной_станции> |
| Обратная польская запись | <https://ru.wikipedia.org/wiki/Обратная_польская_запись#Вычисления_на_стеке> |

КОНТРОЛЬНЫЕ ВОПРОСЫ К ЛАБОРАТОРНОЙ РАБОТЕ

Для успешной защиты работы помимо соответствующего требованиям результата необходимо уверенно отвечать на нижеперечисленные и другие вопросы, а также на контрольные вопросы всех предыдущих лабораторных работ.

1. Что такое Обратная польская запись?
2. Какой алгоритм выполнения этой записи?
3. Что такое делегирование событий?
4. В каких моментах можно использовать делегирование событий?
5. Что такое Алгоритм сортировочной станции?

**В качестве ответа укажите ссылку на git-репозиторий с кодом сайта, а также ссылку на сайт, размещённый на хостинге.**