LU01.A01 - myAlgebraCalculator

Rahmenbedingungen

- Sozialform: Einzelarbeit
- Hilfsmittel: Openbooks
- Zeit: 70 Minuten
- Erwartetes Resultat: JavaScript-File, dass die nachfolgend genannten Taschenrechner-Operationen durchführen kann.
    1. plus
    2. minus
    3. mal
    4. geteilt
    5. quadrat
    6. wurzel
    7. potenz

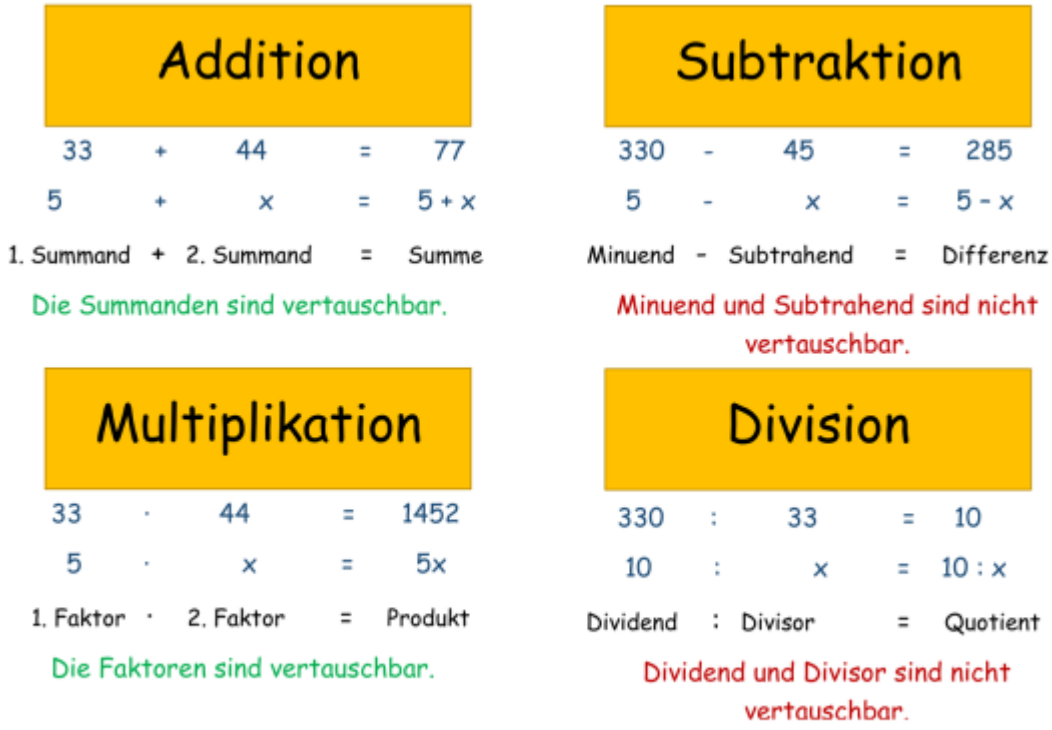
Hinweise

Wichtig bei der Umsetzung Ihrer Lösung ist, dass sie nach best-practise programmieren. Konkret heisst das:

- Ihr Script, sowie die verwendeten Subroutinen (Funktionen, Methoden) müssen ei-nen Header haben: Autor, Datum, Angaben zur Input-, und Output-Parameter, eine kurze Beschreibung zum Verhalten der Funktion.
- Bei fehlenden Parametern wird die eingebende Person entsprechend über das HTML-Formular oder der Kommandozeile informiert.
- Verwenden Sie die unbedingt die passenden Datentypen (primitive oder komplexe).
- Verwenden Sie keine JS-Bibliotheken, sondern programmieren Sie alle relevanten Funktionen selbst. Letztendlich ist das das eigentliche Ziel dieser Übung.
- Aus Gründen der Datenkapselung und der umsichtigen Programmierung realisieren sie Ihre Funktionen ausschliesslich mit Inputparameter und Return-Values. Es wird also möglichst nicht auf globale Werte innerhalb der Methoden zugegriffen.
- Die für die Berechnung benötigten Werte können über Commandline, als Variable oder über ein simples HTML-Formular eingegeben werden.
- Verschwenden Sie keine Zeit in eine *schöne* Oberflächengestaltung, weil es aktuell um Programmierung, und nicht um Gestaltung geht.

Auftrag

Programmieren Sie einen Rechner *myAlgebraCalculator.js*, der nicht nur die Grundrechenarten durchführt, sondern auch quadrieren, potenzieren und die Wurzeln ziehen kann. Testen Sie anschliessend Ihre Lösung auf Funktion, indem Sie die Funktionen ausführen und das Ergebnis auf der Kommandozeile ausgeben lassen.



Der Rechner soll die nachfolgenden Funktionen realisieren:

Teilauftrag 1: Plus

```
// Autor:
// Datum:
// Beschreibung: Zwei eingegebene summand1 und summand2 werden als Summe zurückgegeben.
// Hinweis: Die beiden Summande sind vertrauschbar.
function plus(summand1, summand2) {
  ....
  return summe
}
```

Teilauftrag 2: Minus

```
// Autor:
// Datum:
// Beschreibung: Der eingegebene minuend wird von subrahend abgezogen und die differenz zurückgegeben
// Hinweis: Minuend und Subrahend sind nicht vertauschbar.
function minus(minuend, subrahend ){
  ....
  return differenz
```

```
}
```

Teilauftrag 3: Mal

```
// Autor:
// Datum:
// Beschreibung: Zwei eingegebene Zahlen faktor1 und faktor2 werden
multipliziert und als produkt zurückgegeben.
// Hinweis: faktor1 und faktor2 sind vertauschbar.
function mal(faktor1, faktor2) {
  ....
  return produkt
}
```

Teilauftrag 4: Geteilt

```
// Autor:
// Datum:
// Beschreibung: Der eingegebene dividend wird durch den divisor geteilt und
als quotient zurückgegeben.
// Hinweis: dividend und divisor sind nicht vertauschbar.
// Hinweis: Divisor darf nicht 0 sein, d.h. definitionsgemäss darf nicht
durch 9 geteilt werden.
function geteilt(dividend, divisior) {
  ....
  return quotient
}
```

Teilauftrag 5: Quadrat

```
// Autor:
// Datum:
// Beschreibung: Eine Zahl basis soll mit sich selbst multipliziert und das
Ergebnis als produkt zurückgegeben werden.
// Hinweis: basis und exponent sind nicht vertauschbar.
function quadrat(basis){
  ....
  return produkt
}
```

Teilauftrag 6: Wurzel

```
// Autor:
// Datum:
// Beschreibung: Aus einer Zahl basis soll die Quadratwurzuel gezogen
werden.
```

```
// Hinweis: Die Operation wurzel wird mit **Math.sqrt** realisiert.
function wurzel(basis){
  ....
  return produkt
}
```

Teilauftrag 7: Potenz

```
// Autor:
// Datum:
// Beschreibung: zu einer zahl soll die Pozent gebildet werden. Dies
erledigen wir in einer Schleife (for oder while) mit dem exponenten im
Schleifenkopf, wobei die basis mit sich selbst entsprechend oft
multipliziert wird.
// Hinweis: Potenz und Exponent sind nicht vertauschbar
// Advanced: Erledigen Sie die Aufgaben in 2 Versionen, mit einer for- und
mit einer while-Schleife.
function potenz( basis, exponent) {
  ....
  return produkt
}
```

Lösungen

LU01.L01

 Volkan Demir

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/de/modul/m307/learningunits/lu01/aufgaben/01?rev=1772098986

Last update: **2026/02/26 10:43**