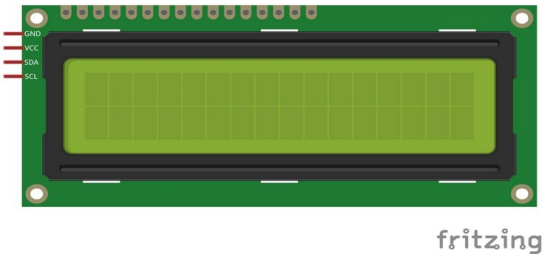
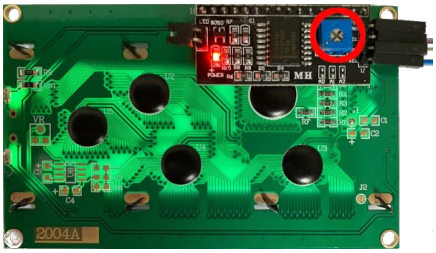
Taschenrechner Grundrechenarten

Im Seriellen Monitor werden Rechenaufgaben der Grundrechenarten eingegeben. Im Seriellen Monitor und auf dem LCD wird die Rechnung und das Ergebnis angezeigt.
Außerdem soll die Eingabe von Buchstaben und die Division durch 0 verhindert werden.

So sieht es aus:

Schließe das LCD an:

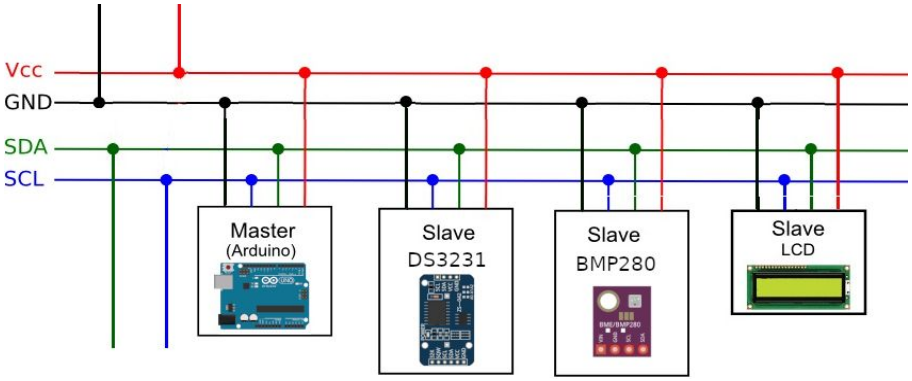




Auf der Rückseite befindet sich ein Potentiometer mit dem die Helligkeit eingestellt werden kann.

Normalerweise wäre eine komplexe Verkabelung zum Betrieb eines LCDs nötig. Der I²C-Bus regelt über einen eigenen Mikroprozessor die Kommunikation der Datenleitungen untereinander.
Es werden deshalb nur vier Anschlüsse benötigt.

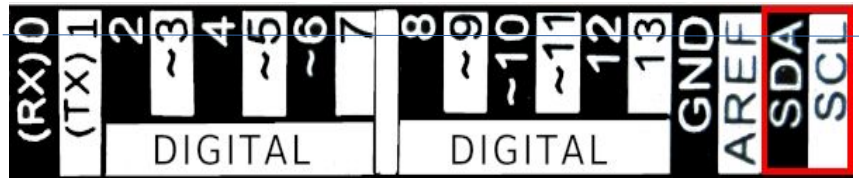
Der I²C-Bus (Inter Integrated Circuit) wurde ursprünglich von Philips entwickelt, er sollte die Kommunikation mit einem Master (dem Arduino) und den verschiedenen Bauelementen (den Slaves) ermöglichen.



Quelle: http://prometec.org/displays/the-i2c-bus (eigene Bearbeitung)

Der I²C-Bus kommt mit zwei Datenleitungen aus:

- die Taktleitung SCL (Serial Clock) → A5
- die Datenleitung SDA (Serial Data) → A4



Statt A4 (SDA) und A5 (SCL) kannst du auch die mit SCL und SDA beschrifteten Pins verwenden.

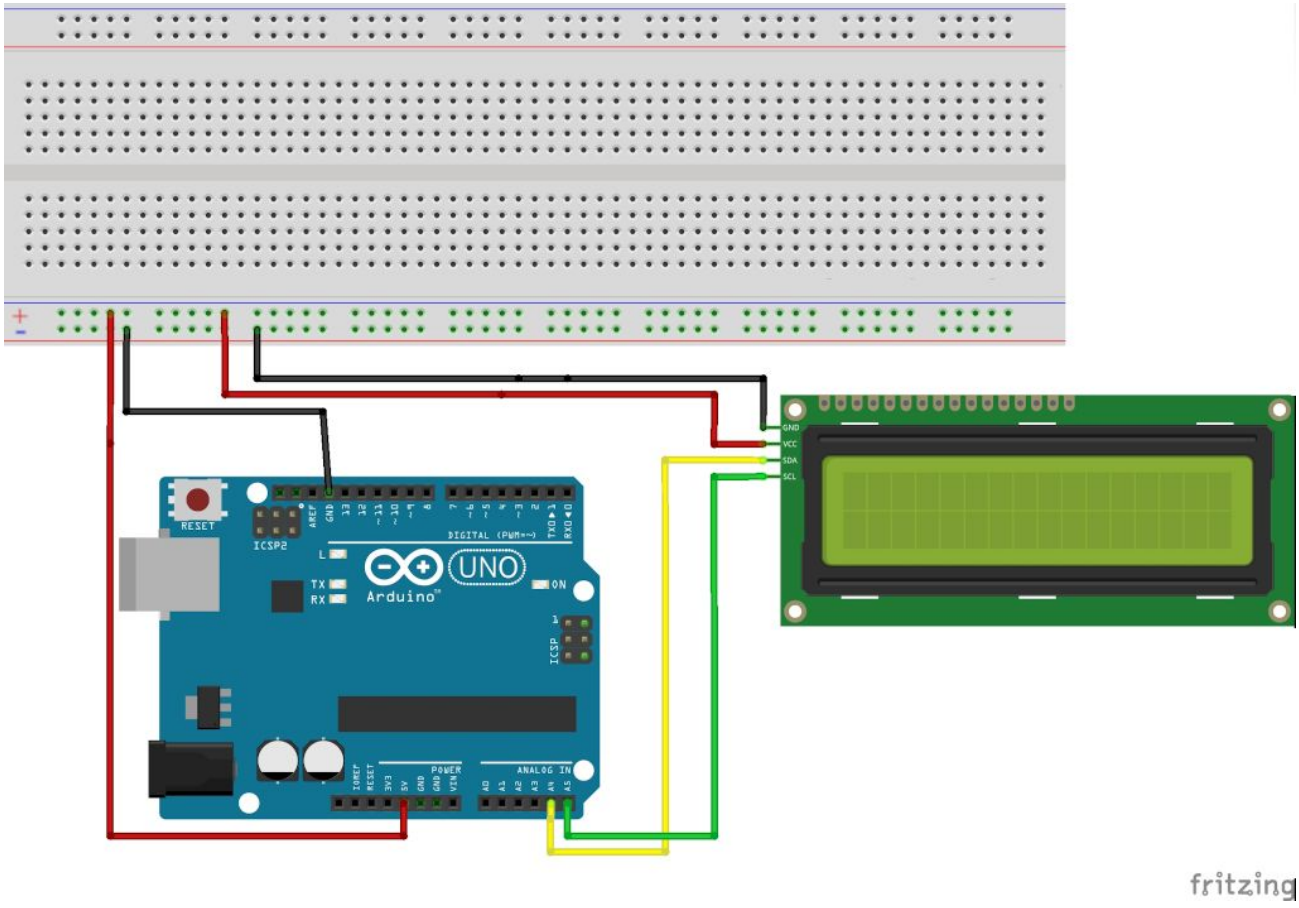
Benötigte Bibliothek:

Sketch → Bibliothek einbinden → Bibliotheken verwalten

Benötigte Bauteile:

- LCD 1602
- Leitungsdrähte

Baue die Schaltung auf.



Binde die benötigte Bibliothek ein und definiere die Variablen.

```
# include <LiquidCrystal_I2C.h>

// LCD Port festlegen
LiquidCrystal_I2C lcd(0x27, 20, 4);

String Eingabe;
String Rechnung;
long Ergebnis;

// Fehlervariablen für gefundenen Buchstaben oder Division durch 0
bool BuchstabeGefunden = false;
bool DivisionNull = false;
```

0x27 ist die ID des I^2C-Busses, 20 bestimmt die Anzahl der Zeichen pro Zeile, 4 nennt die Anzahl der Zeilen.

Wenn dir die ID des verwendeten Displays nicht bekannt ist, kannst du sie mit folgendem ➡Programm herausfinden.

Der setup-Teil. Beachte die Kommentare.

```
void setup()
{
  Serial.begin(9600);

  // LCD einschalten
  lcd.init();
  lcd.backlight();

  // Erklärung anzeigen Serieller Monitor
  Serial.println("Taschenrechner");
  Serial.println("_____________________");
  Serial.println("+\tAddition");
  Serial.println("-\tSubtraktion");
  Serial.println("*\tMultiplikation");
  Serial.println(":\tDivision");
  Serial.println("_____________________");

  // LCD
  lcd.setCursor(0, 0);
  lcd.print("+ Addition");
  lcd.setCursor(0, 1);
  lcd.print("- Subtraktion");
  lcd.setCursor(0, 2);
  lcd.print("* Multiplikation");
  lcd.setCursor(0, 3);
  lcd.print(": Division");
}
```

Der loop-Teil. Beachte die Kommentare:

```
void loop()
{
  String GleichZeichen = " = ";
  lcd.setCursor(0, 0);
  while (Serial.available() > 0)
  {
    // solange lesen, bis return \n = return eingegeben wurde
    Eingabe = Serial.readStringUntil("\n");

    // das letzte Zeichen ist return = \n → soll entfernt werden (-1)
    Eingabe = Eingabe.substring(0, Eingabe.length() - 1);

    lcd.clear();
    // Addition
    // "+" Zeichen suchen
    if (Eingabe.indexOf('+') > 0)
    {
      int PlusZeichen = Eingabe.indexOf('+');
      // erster Summand: von 0 bis zur Position des Pluszeichens
      String SummandEins = Eingabe.substring(0, PlusZeichen);

      // zweiter Summand: von der Position hinter dem Pluszeichen
      // bis zum letzten Zeichen
      String SummandZwei = Eingabe.substring(PlusZeichen + 1, Eingabe.length());

      // rechnen
      Ergebnis = SummandEins.toInt() + SummandZwei.toInt();

      // String Rechnung "zusammenbauen"
      Rechnung = SummandEins + " + " + SummandZwei + GleichZeichen + String(Ergebnis);
      BuchstabeGefunden = BuchstabeSuchen(SummandEins + SummandZwei);
    }

    // Subtraktion
    // "-" Zeichen suchen
    if (Eingabe.indexOf('-') > 0)
    {
      int MinusZeichen = Eingabe.indexOf('-');
      // Minuend: von 0 bis zur Position des Minuszeichens
      String Minuend = Eingabe.substring(0, MinusZeichen);

      // Subtrahend: von der Position hinter dem Minuszeichen
      // bis zum letzten Zeichen
      String Subtrahend = Eingabe.substring(MinusZeichen + 1, Eingabe.length());

      // rechnen
      Ergebnis = Minuend.toInt() - Subtrahend.toInt();

      // String Rechnung "zusammenbauen"
      Rechnung = Minuend + " - " + Subtrahend + GleichZeichen + String(Ergebnis);
```

```
    // nach Buchstaben suchen
    BuchstabeGefunden = BuchstabeSuchen(Subtrahend + Minuend);
  }

  // Multiplikation
  // "*" Zeichen suchen
  if (Eingabe.indexOf('*') > 0)
  {
    int MalZeichen = Eingabe.indexOf('*');
    // erster Faktor: von 0 bis zur Position des Malzeichens
    String FaktorEins = Eingabe.substring(0, MalZeichen);

    // zweiter Faktor: von der Position hinter dem Malzeichen
    // bis zum letzten Zeichen
    String FaktorZwei = Eingabe.substring(MalZeichen + 1, Eingabe.length());

    // rechnen
    Ergebnis = FaktorEins.toInt() * FaktorZwei.toInt();

    // String Rechnung "zusammenbauen"
    Rechnung = FaktorEins + " * " + FaktorZwei + GleichZeichen + String(Ergebnis);
  }

  // Division
  // "/" Zeichen suchen
  if (Eingabe.indexOf(':') > 0)
  {
    int GeteiltZeichen = Eingabe.indexOf(':');
    // Dividend: von 0 bis zur Position des Geteiltzeichens
    String Dividend = Eingabe.substring(0, GeteiltZeichen);

    // Divisor: von der Position hinter dem Geteiltzeichen
    // bis zum letzten Zeichen
    String Divisor = Eingabe.substring(GeteiltZeichen + 1, Eingabe.length());

    /* String Rechnung "zusammenbauen"
       Besonderheit: das Ergebnis einer Division kann eine Kommazahl sein
       → Datentyp float ->  Kommazahlen
       Ergebnis zeigt nur 2 Nachkommastellen
    */

    // Division durch 0 nicht möglich
    if (Divisor == "0")
    {
      DivisionNull = true;

      // String Rechnung ohne Ergebnis "zusammenbauen"
      Rechnung = Dividend + " : " + Divisor + GleichZeichen;
    }
```

```
      // Divisor != 0 -> rechnen
      else
      {
        float Quotient = Dividend.toFloat() / Divisor.toFloat();

        // String Rechnung "zusammenbauen"
        Rechnung = Dividend + " : " + Divisor + GleichZeichen + String(Quotient);
        BuchstabeGefunden = BuchstabeSuchen(Dividend + Divisor);

        // . durch , ersetzen
        Rechnung.replace(".", ",");
      }
    }

    // wenn kein Buchstabe gefunden wurde oder Divisor != 0 -> Rechnung ausgeben
    if (!BuchstabeGefunden && !DivisionNull) RechnungAusgeben();

    // Division durch 0 -> Meldung anzeigen
    if (DivisionNull)
    {
      Serial.println(Rechnung);
      Serial.println("Division durch 0 ist verboten ;-)");
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print(Rechnung);
      lcd.setCursor(0, 1);
      lcd.print("Divison durch 0");
      lcd.setCursor(0, 2);
      lcd.print("ist verboten ;-)");
    }

    // Buchstabe eingegeben -> Meldung anzeigen
    if (BuchstabeGefunden)
    {
      Rechnung = Rechnung.substring(0, Rechnung.indexOf('='));
      Serial.println(Rechnung);
      Serial.println("Buchstaben sind nicht erlaubt ;-)");
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print(Rechnung);
      lcd.setCursor(0, 1);
      lcd.print("Buchstaben sind ");
      lcd.setCursor(0, 2);
      lcd.print("nicht erlaubt ;-)");
    }

    // Eingaben löschen, Fehlervariablen auf false setzen
    Rechnung = "";
    Eingabe = "";
    DivisionNull = false;
    BuchstabeGefunden = false;
  }
}
```

Die Methode RechnungAusgaben() zeigt die Rechnung und das Ergebnis im Seriellen Monitor und auf dem LCD an.

Für die Anzeige auf dem LCD gibt es noch eine Besonderheit:
Wenn die Rechnung 20 Zeichen überschreitet, wird sie vor dem =-Zeichen umgebrochen.

```
void RechnungAusgeben()
{
  // Ausgabe Serieller Monitor
  Serial.println(Rechnung);

  // Ausgabe LCD
  lcd.clear();

  // Ergebnis zu String umwandeln, damit die Länge bestimmt werden kann
  String LaengeErgebnis = String(Ergebnis);

  lcd.setCursor(0, 0);
  int laenge = Rechnung.length();

  // wenn die Länge der Rechnung > 20
  if (laenge > 20)
  {
    // Position des =-Zeichen bestimmen
    int GleichZeichen = Rechnung.indexOf('=');

    // Rechnung bis =-Zeichen anzeigen
    lcd.print(Rechnung.substring(0, GleichZeichen + 1 ));

    // Ergebnis anzeigen
    lcd.setCursor(0, 1);
    lcd.print(Rechnung.substring(GleichZeichen + 2, Rechnung.length()));
  }
  // Rechnung passt in eine Zeile
  else lcd.print(Rechnung);
}
```

Jetzt fehlt nur noch die Funktion, mit deren Hilfe geprüft wird, ob in der Rechnung Buchstaben vorhanden sind.

```
bool BuchstabeSuchen(String Pruefen)
{
  bool BuchstabeGefunden = false;

  // wenn das Zeichen im zu prüfenden String ein Buchstabe (isAlpha) ist
  // -> BuchstabeGefunden = true
  for (int zaehler = 0; zaehler < Pruefen.length(); zaehler++)
  {
    if (isAlpha(Pruefen.charAt(zaehler)))
    {
      BuchstabeGefunden = true;

      // nicht weiter suchen Schleife verlassen
      break;
    }
  }

  return BuchstabeGefunden;
}
```

Das Programm soll jetzt um die Rechnung mit Dezimalzahlen erweitert werden.

- Der Arduino kennt leider nur den Datentyp float, der mit einfacher Genauigkeit rechnet. Deshalb werden lediglich zwei Nachkommastellen berechnet.
- Arduino verwendet die Punkt-Notation für Dezimalzahlen (2.5 statt 2,5). Die Eingabe soll aber mit Komma erfolgen.
- Wenn das Ergebnis eine natürlich Zahl ist, sollen die Nachkommastellen abgeschnitten werden.

Der Kopf des Programms:

```
# include <LiquidCrystal_I2C.h>

// LCD Port festlegen
LiquidCrystal_I2C lcd(0x27, 20, 4);

String Eingabe;
String Rechnung;
long Ergebnis;

// Fehlervariablen für gefundenen Buchstaben oder Division durch 0
bool BuchstabeGefunden = false;
bool DivisionNull = false;
```

Der setup-Teil:

```
void setup()
{
  Serial.begin(9600);

  // LCD einschalten
  lcd.init();
  lcd.backlight();

  // Erklärung anzeigen Serieller Monitor
  Serial.println("Taschenrechner");
  Serial.println("_____________________");
  Serial.println("+\tAddition");
  Serial.println("-\tSubtraktion");
  Serial.println("*\tMultiplikation");
  Serial.println(":\tDivision");
  Serial.println("_____________________");

  // LCD
  lcd.setCursor(0, 0);
  lcd.print("+ Addition");
  lcd.setCursor(0, 1);
  lcd.print("- Subtraktion");
  lcd.setCursor(0, 2);
  lcd.print("* Multiplikation");
  lcd.setCursor(0, 3);
  lcd.print(": Division");
}
```

Der geänderte loop-Teil. Beachte die Kommentare.

```
void loop()
{
  String GleichZeichen = " = ";
  lcd.setCursor(0, 0);
  while (Serial.available() > 0)
  {
    // solange lesen, bis return \n = return eingegeben wurde
    Eingabe = Serial.readStringUntil("\n");

    // das letzte Zeichen ist return = \n → soll entfernt werden (-1)
    Eingabe = Eingabe.substring(0, Eingabe.length() - 1);

    lcd.clear();
    // Addition
    // "+" Zeichen suchen
    if (Eingabe.indexOf('+') > 0)
    {
      int PlusZeichen = Eingabe.indexOf('+');

      // erster Summand: von 0 bis zur Position des Pluszeichens
      String SummandEins = Eingabe.substring(0, PlusZeichen);

      // Leerzeichen entfernen
      SummandEins.trim();

      // für die Rechnung , durch . ersetzen
      SummandEins.replace(",", ".");

      // zweiter Summand: von der Position hinter dem Pluszeichen
      // bis zum letzten Zeichen
      String SummandZwei = Eingabe.substring(PlusZeichen + 1, Eingabe.length());

      // Leerzeichen entfernen
      SummandZwei.trim();

      // für die Rechnung , durch . ersetzen
      SummandZwei.replace(",", ".");

      // rechnen
      Ergebnis = SummandEins.toFloat() + SummandZwei.toFloat();

      // prüfen, ob Ergebnis eine natürliche Zahl
      String StringPruefen = NullEntfernen(String(Ergebnis));

      // String Rechnung "zusammenbauen"
      Rechnung = SummandEins + " + " + SummandZwei + GleichZeichen +
String(StringPruefen);

      Rechnung.replace(".", ",");

      BuchstabeGefunden = BuchstabeSuchen(SummandEins + SummandZwei);
    }
```

```
// Subtraktion
// "-" Zeichen suchen
if (Eingabe.indexOf('-') > 0)
{
  int MinusZeichen = Eingabe.indexOf('-');

  // Minuend: von 0 bis zur Position des Minuszeichens
  String Minuend = Eingabe.substring(0, MinusZeichen);

  // Leerzeichen entfernen
  Minuend.trim();

  // für die Rechnung , durch . ersetzen
  Minuend.replace(",", ".");

  // Subtrahend: von der Position hinter dem Minuszeichen
  // bis zum letzten Zeichen
  String Subtrahend = Eingabe.substring(MinusZeichen + 1, Eingabe.length());

  // Leerzeichen entfernen
  Subtrahend.trim();

  // für die Rechnung , durch . ersetzen
  Subtrahend.replace(",", ".");

  // rechnen
  Ergebnis = Minuend.toFloat() - Subtrahend.toFloat();

  // prüfen, ob Ergebnis eine natürliche Zahl
  String StringPruefen = NullEntfernen(String(Ergebnis));

  // String Rechnung "zusammenbauen"
  Rechnung = Minuend + " - " + Subtrahend + GleichZeichen + String(StringPruefen);
  Rechnung.replace(".", ",");

  // nach Buchstaben suchen
  BuchstabeGefunden = BuchstabeSuchen(Subtrahend + Minuend);
}

// Multiplikation
// "*" Zeichen suchen
if (Eingabe.indexOf('*') > 0)
{
  int MalZeichen = Eingabe.indexOf('*');

  // erster Faktor: von 0 bis zur Position des Malzeichens
  String FaktorEins = Eingabe.substring(0, MalZeichen);

  // Leerzeichen entfernen
  FaktorEins.trim();

  // für die Rechnung , durch . ersetzen
  FaktorEins.replace(",", ".");

  // zweiter Faktor: von der Position hinter dem Malzeichen
```

```
      // bis zum letzten Zeichen
      String FaktorZwei = Eingabe.substring(MalZeichen + 1, Eingabe.length());

      // Leerzeichen entfernen
      FaktorZwei.trim();

      // für die Rechnung , durch . ersetzen
      FaktorZwei.replace(",", ".");

      // rechnen
      Ergebnis = FaktorEins.toFloat() * FaktorZwei.toFloat();

      // prüfen, ob Ergebnis eine natürliche Zahl
      String StringPruefen = NullEntfernen(String(Ergebnis));

      // String Rechnung "zusammenbauen"
      Rechnung = FaktorEins + " * " + FaktorZwei + GleichZeichen +
String(StringPruefen);
      Rechnung.replace(".", ",");
    }

    // Division
    // "/" Zeichen suchen
    if (Eingabe.indexOf(':') > 0)
    {
      int GeteiltZeichen = Eingabe.indexOf(':');
      // Dividend: von 0 bis zur Position des Geteiltzeichens
      String Dividend = Eingabe.substring(0, GeteiltZeichen);

      // Leerzeichen entfernen
      Dividend.trim();

      Dividend.replace(",", ".");

      // Divisor: von der Position hinter dem Geteiltzeichen
      // bis zum letzten Zeichen
      String Divisor = Eingabe.substring(GeteiltZeichen + 1, Eingabe.length());

      // Leerzeichen entfernen
      Divisor.trim();
      Divisor.replace(",", ".");

      /* String Rechnung "zusammenbauen"
         Besonderheit: das Ergebnis einer Division kann eine Kommazahl sein
         -> Datentyp float -> Kommazahlen
         Ergebnis zeigt nur 2 Nachkommastellen
      */

      // Division durch 0 nicht möglich
      if (Divisor == "0")
      {
        DivisionNull = true;

        // String Rechnung ohne Ergebnis "zusammenbauen"
        Rechnung = Dividend + " : " + Divisor + GleichZeichen;
      }
```

```
      // Divisor != 0 -> rechnen
      else
      {
        Ergebnis = Dividend.toFloat() / Divisor.toFloat();
        String StringPruefen = NullEntfernen(String(Ergebnis));

        // String Rechnung "zusammenbauen"
        Rechnung = Dividend + " : " + Divisor + GleichZeichen + String(StringPruefen);
        BuchstabeGefunden = BuchstabeSuchen(Dividend + Divisor);

        // . durch , ersetzen
        Rechnung.replace(".", ",");
      }
    }

    // wenn kein Buchstabe gefunden wurde oder Divisor != 0 -> Rechnung ausgeben
    if (!BuchstabeGefunden && !DivisionNull) RechnungAusgeben();

    // Division durch 0 -> Meldung anzeigen
    if (DivisionNull)
    {
      Serial.println(Rechnung);
      Serial.println("Division durch 0 ist verboten ;-)");
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print(Rechnung);
      lcd.setCursor(0, 1);
      lcd.print("Divison durch 0");
      lcd.setCursor(0, 2);
      lcd.print("ist verboten ;-)");
    }

    // Buchstabe eingegeben -> Meldung anzeigen
    if (BuchstabeGefunden)
    {
      Rechnung = Rechnung.substring(0, Rechnung.indexOf('='));
      Serial.println(Rechnung);
      Serial.println("Buchstaben sind nicht erlaubt ;-)");
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print(Rechnung);
      lcd.setCursor(0, 1);
      lcd.print("Buchstaben sind ");
      lcd.setCursor(0, 2);
      lcd.print("nicht erlaubt ;-)");
    }

    // Eingaben löschen, Fehlervariablen auf false setzen
    Rechnung = "";
    Eingabe = "";
    DivisionNull = false;
    BuchstabeGefunden = false;
  }
}
```

Die Methode RechnungAusgeben() und die Funktion BuchstabeSuchen() bleiben unverändert.

```
void RechnungAusgeben()
{
  // Ausgabe Serieller Monitor
  Serial.println(Rechnung);

  // Ausgabe LCD
  lcd.clear();

  // Ergebnis zu String umwandeln, damit die Länge bestimmt werden kann
  String LaengeErgebnis = String(Ergebnis);

  lcd.setCursor(0, 0);
  int laenge = Rechnung.length();

  // wenn die Länge der Rechnung > 20
  if (laenge > 20)
  {
    // Position des =-Zeichen bestimmen
    int GleichZeichen = Rechnung.indexOf('=');

    // Rechnung bis =-Zeichen anzeigen
    lcd.print(Rechnung.substring(0, GleichZeichen + 1 ));

    // Ergebnis anzeigen
    lcd.setCursor(0, 1);
    lcd.print(Rechnung.substring(GleichZeichen + 2, Rechnung.length()));
  }

  // Rechnung passt in eine Zeile
  else lcd.print(Rechnung);
}

bool BuchstabeSuchen(String Pruefen)
{
  bool BuchstabeGefunden = false;

  // wenn das Zeichen im zu prüfenden String ein Buchstabe (isAlpha) ist
  // -> BuchstabeGefunden = true
  for (int i = 0; i < Pruefen.length(); i++)
  {
    if (isAlpha(Pruefen.charAt(i)))
    {
      BuchstabeGefunden = true;

      // nicht weiter suchen Schleife verlassen
      break;
    }
  }

  return BuchstabeGefunden;
}
```

Die Funktion NullEntfernen() testet, ob das Ergebnis eine natürliche Zahl ist – die beiden Nackommenstellen sind 0 - und schneidet die Nachkommastellen ab.

```
String NullEntfernen(String PruefString)
{
  String TestString;
  PruefString.replace(" ", "");

  // wenn die letzten Ziffern 00 sind -> Ergebnis ist natürliche Zahl
  if (PruefString.indexOf(".00") > 0)
  {
    TestString = PruefString.substring(0, PruefString.indexOf(".00"));
  }
  else TestString = PruefString;

  return TestString;
}
```