```
/*

    Aufgabe 1: Begriffe

    Instruction Set: Der Maschinenbefehlssatz den ein Mikroprozessor
    beherrscht.

    Interrupt: Die Unterbrechung des gerade laufenden Prozesses
    bzw. Programms um auf eigetretendes Ereignis, wie z.B. eine
    Tastatureingabe zu reagieren.

    Prozess: Ein Prozess ist eine algorithmisch beschriebene
    Verarbeitung von gegebenen Daten zu einer definierten Ausgabe.

    Datei: Eine Datei ist eine Ansammlung von Daten, welche
    physikalisch unter einem Bezeichner auf einem Datenträger
    festgehalten sind.

    Systemaufruf: Eine Methode, um vom Betriebssystem
    bereitgestellte Funktionalitäten auszuführen oder auch vom
    Betriebssystem verwaltete Ressourcen zuzugreifen.

    Multitasking: Entweder die Möglichkeit für Benutzer mehrere
    Programme bzw. Prozesse gleichzeitig auszuführen, wobei das
    Betriebssystem hier jedem Prozess immer nur für gewisse
    Zeitintervalle den Zugriff erlaubt auf Hardwareressourcen,
    oder die tatsächliche parallele Ausführung von mehreren Prozessen
    auf einem System mit mehreren Prozessorkernen.

    Aufgabe 2: Kernel

    Unterschiede:

    Der elemantare Unterschied zwischen den beiden Kernelarten ist,
    dass beim monolithischen Kernel fast alle Funktionen des
    Betriebssystem im Kernel integriert sind. Bei dem Microkernel
    dagegen versucht man nur die essentiellen Funktionen im Kernel
    laufen zu lassen. Andere Funktionen werden in Module ausgelagert
    und über Schnittstellen ist die Kommunikation mit dem Kernel
    möglich.

    Daraus ergeben sich Vor- und Nachteile der verschiedenen
    Architekturen.

    Vorteile Monolithischer Kernel:

    - Schneller, weil weniger Kommunikationsbedarf und Kontextwechsel
    - Bei kleinen Systemen einfacher zu implementieren

    Nachteile Monolithischer Kernel:

    - Kann schnell komplex und unstrukturiert werden bei
    größeren Anforderungen
    - hohe Fehleranfälligkeit durch fehlenden Zugriffsschutz in
    den versch. Kernelkomponenten
    - Menge an Komponenten die kritische Fehler also Systemabstürze
    bewirken können ist größer

    Vorteile Microkernel:

    - Strukturierte und definierte Schnittstellen für Kommunikation
    ermöglichen schnellere Implementierung bei komplexen Anforderungen
    - Menge an systemkritischen Komponenten sehr gering also sind
    Systemabstürze selten
    - Zugriffsschutz durch getrennte Komponenten realisiert

    Nachteile Microkernel:

    - langsamer, weil erhöhter Kommunikationsbedarf zwischen Komponenten
    - Schnittstellen und Modularisierung für einfache Systeme evtl. zu
    komplex also überflüssige Entwicklungskosten

    3. Aufgabe: Begriffe
```

```
    Konzept:

    Tritt ein Interrupt auf so wird eine Interrupt Service Routine (ISR)
    ausgeführt um auf dieses Ereignis zu reagieren. Während der Ausführung
    kann theoretisch erneut ein Interrupt auftreten. Hierbei gibt es die
    Möglichkeit den Interrupt erstmal zu ignorieren und die laufende ISR
    erstmal zu beenden. Eine Alternative sind Nested Interrupts. Hierbei
    wird am Anfang der ISR dem Auslöser des Interrupts signalisiert, dass
    die Unterbrechung wahrgenommen wurde und dann werden Interrupts wieder
    aktiviert. Also das Interrupts enabled Bit auf True gesetzt. D.h.
    tritt während der Ausführung der ISR ein Interrupt auf, wird die ISR
    verlassen und der neue Interrupt bearbeitet. Es wird also immer nur
    die Bearbeitung gestartet und neue Interrupts solange verschachtelt
    bis alle abgearbeitet sind. Wenn man dabei nur nach der zeitlichen
    Reihenfolge geht, werden die Interrupts also quasi auf eine Art Stack
    gepusht und der oberste Interrupt auf dem Stack immer bearbeitet.
    Ist eine ISR fertig wird der Interrupt gepopt und kommt ein neuer
    Interrupt wird dieser gepusht und bearbeitet.

    Vorteile:

    - Geringe Verzögerungen, weil die ISR nur ganz kurz nicht
    unterbrochen werden kann
    - Der aktuellste Interrupt wird zuerst beendet (FIFO)
    - Interrupts gehen seltener verloren (nur wenn z.B. im blockierten
    Teil der ISR zwei Mal
    eine Eingabe per Tastatur erfolgt, anstatt der gesamten ISR)

    Nachteile:

    - Kompliziert zu implementieren
    - Wenn zu viele Interrupts auftreten, werden immer nur die
    aktuellsten bearbeitet und die ersten Interrupts haben eine
    große Verzögerung (FIFO)
*/

#include <stdio.h>
int main(int argc, const char * argv[])
{
    if (argc != 2)
    {
        perror("Programm muss mit Datei gestartet werden!");
        return 1;
    }
    //Datei öffnen
    FILE *testFile = fopen(argv[1], "r");
    if (testFile != NULL)
    {
        double sum = 0;
        if (fscanf(testFile, "%lf", &sum) == 1)
        {
            double comparedSum = 0, firstValue = 0, secondValue = 0;
            while (fscanf(testFile, "%lf %*c %lf",&firstValue, &secondValue) == 2) c
omparedSum += firstValue * secondValue;
            if (comparedSum == sum) puts("Korrekt.");
            else puts("Falsch.");
        }
        else
        {
            perror("Fehler. Datei hat falsches Format.");
            fclose(testFile);
            return 1;
        }
        fclose(testFile);
        return 0;
    }
    else
    {
        perror("Fehler. Datei konnte nicht gelesen werden");
        return 1;
    }
}
```