Einstieg in die strukturierte Assembler-Programmierung

Name : _____	**Name : _____**
Vorname : _____	**Vorname : _____**
Matrikel-Nr : _____	**Matrikel-Nr : _____**

Problemstellung:

Ziel des Praktikumsversuchs ist die Erstellung eines Assemblerprogramms zur Berechnung der Primzahlen im Zahlenraum von 2 bis 1000. Hierzu soll auf das so genannte „Sieb des Eratosthenes“ zurückgegriffen werden.

Dabei werden aus der Folge der natürlichen Zahlen eines Intervalls von vorn beginnend nacheinander die Vielfachen der bisher ermittelten Primzahlen herausgestrichen, also die Vielfachen von 2, 3, 5, 7, 11 usw. Da z.B. die 9 durch 3 teilbar ist, wurde sie bereits früher gestrichen und wird daher hinter der 7 nicht mehr betrachtet. Als Ergebnis verbleiben zum Schluss alle Primzahlen des Intervalls.

Allgemeine Konzepterstellung (4. Woche – 6. Woche):

Begleitend zu den unten genannten Aufgaben soll ein zugehöriges Konzept (textuell und graphisch) erstellt werden, dass zu jedem Praktikumstermin vorzuzeigen und zu verfeinern ist. Das Konzept soll dabei die folgenden Punkte adressieren:

- Analyse der Aufgabenstellung
- Aufbau des Programms, z.B. als Java-Programm, Pseudocode, Ablaufdiagramm, etc.
- Entwicklung der Implementierungsstrategie, also z.B. welche Felder sollen verwendet werden? Von welchem Typ sollen die Elemente sein? Welche Instruktionen stehen zur Verfügung? Denken Sie hier auch an eine graphische Darstellung.

Hinweise zur Aufgabenbearbeitung (4. Woche – 6. Woche):

Zur Umsetzung der Aufgabenstellung ist es sinnvoll die Problemstellung in die folgenden Teilfunktionen zu unterteilen:

- **Sieb:** In dieser Teilfunktion werden die Primzahlen „ausgesiebt“, also von den Nicht-Primzahlen getrennt. Hierfür ist es sinnvoll ein Feld (Array) zu erzeugen, in dem jeder Eintrag eine Zahl von 2 bis 1000 repräsentiert. Dabei werden nicht die Zahlen selbst, sondern nur die Information im Feld selbst gespeichert, ob es sich um eine Primzahl handelt oder nicht; die Zahl selbst lässt sich über den Index (also die Speicheradresse) festlegen. Somit ist bspw. Feldelement 15 der Zahl 15 zugeordnet und enthält die Information, ob 15 eine Primzahl ist oder nicht.

  Hinweis: Das Herausstreichen der Vielfachen einer Zahl kann beim Quadrat dieser Zahl beginnen, da alle kleineren Faktoren schon vorher berücksichtigt wurden. So kann bspw. nachdem die 2,3,5 und 7 herausgestrichen wurden mit der 11 fortgefahren werden, da 9 keine Primzahl ist. Das Streichen kann bei 11*11 beginnen und mit 13*11, 15*11 etc. fortfahren. Der Ablauf endet, sobald die Obergrenze überschritten wurde

- **Abspeichern:** Diese Teilfunktion analysiert das gesiebte Feld und schreibt die Primzahlen in einen anderen Speicherbereich

Aufgabenstellung (4. Woche):

Im ersten Schritt wird die grundlegende Funktionalität des beschriebenen Algorithmus festgelegt. Hierzu sind folgende Punkte zu bearbeiten:

- Schreiben Sie eine `main.s` zunächst **ohne Assembler Code**, ausschließlich mit Kommentaren. Ziel der Kommentare soll eine Beschreibung dessen sein, was aus Ihrer Sicht der zukünftige Code zu leisten hat. Nutzen Sie dazu die weiter unten angegebene Unterteilung und das oben geforderten Konzept der Aufgabe in Java oder als Pseudocode.
- Erstellen Sie zudem eine **graphische Umsetzung**, in der der Aufbau sowie Zugriff und Inhalt auf die Elemente im Speicher dargestellt werden

Aufgabenstellung (5. Woche):

- Fügen Sie in Ihre `main.s` aus der vorherigen Woche assembler code für die von Ihnen identifizierten strukturierten Steuerbefehle ein. Diese Aufgabe

bezieht sich ausschließlich auf die Teilfunktion **Sieb** (siehe **Hinweise zur Aufgabenbearbeitung**)
- Beschreiben Sie auch den assembler code, der für den Abbruch der Schleifen und die Fallunterscheidungen benötigt wird
- Erzeugen Sie das Sieb für die spätere Auswertung
- Ergänzen Sie die genannten Aspekte entsprechend in ihrem Konzept
- Die Teilfunktion **Abspeichern** muss zu diesem Zeitpunkt noch nicht implementiert werden

Aufgabenstellung (6. Woche):

- Vervollständigen Sie aufbauend auf Ihren Ergebnissen aus den vorherigen beiden Wochen die Teilfunktion Sieb um die Datenbefehle, also die Umsetzung des Speicherzugriffs zum Streichen von Nicht-Primzahlen
- Ergänzen Sie Ihr Assemblerprogramm um die Teilfunktion **Abspeichern,** also das abschließende Schreiben der Primzahlen in einen anderen Speicherbereich (siehe **Hinweise zur Aufgabenbearbeitung**)
- Ergänzen Sie die genannten Aspekte entsprechend in ihrem Konzept
- Demonstrieren Sie die vollständige Funktionalität Ihres Assemblerprogramm zur Berechnung der Primzahlen von 2 bis 1000. Es ist zu zeigen, dass
    - Die Primzahlen fehlerfrei bestimmt und abgespeichert werden
    - Der Assemblercode sauber strukturiert und kommentiert ist
    - anhand der Sprunglabel für Schleifen und Fallunterscheidungen die Funktion und das Einsatzgebiet klar erkennbar sind
    - die Speicherbereiche wie in der Aufgabenstellung gefordert funtionieren (in Keil)

Praktikumsvorbereitung und –abnahme (4. Woche – 6. Woche):

- Zur **Vorbereitung** müssen wie gehabt die wesentlichen Teile der jeweiligen Aufgabe schon vor dem jeweiligen Praktikumstermin bearbeitet sein.

- Für die **Abnahme** müssen Sie
    - ihre Lösung in der `main.s` darstellen und Änderungen an der Lösung vornehmen können.
    - den Algorithmus in eigenen Worten beschreiben und anhand des Konzepts erklären können
    - die verwendeten Implementierungsstrategien insbesondere Aufbau des genutzten Speichers erklären können