Hinweise für den Prüfling

Auswahlzeit: 30 Minuten

Bearbeitungszeit (insgesamt): 240 Minuten

Auswahlverfahren

Es gibt drei Aufgabengruppen A, B und C, aus denen jeweils ein Vorschlag zu bearbeiten ist. Die vorliegenden Vorschläge aus den Gruppen A (Objektorientierte Modellierung) und B (Konzepte und Anwendungen der theoretischen Informatik) sind Pflichtvorschläge.

Wählen Sie von den zwei vorliegenden Vorschlägen der Gruppe C (Datenbanken) einen zur Bearbeitung aus. Der nicht ausgewählte Vorschlag muss am Ende der Auswahlzeit der Aufsicht führenden Lehrkraft zurückgegeben werden.

Erlaubte Hilfsmittel

1. ein Wörterbuch der deutschen Rechtschreibung
2. eine Liste der fachspezifischen Operatoren

Sonstige Hinweise

ohne PC-Nutzung

In jedem Fall vom Prüfling auszufüllen

Name: _____ Vorname: _____

Prüferin/Prüfer: _____ Datum: _____

Kommentare

Bevor ein Compiler Programmcode aus einer höheren Programmiersprache wie Java, Delphi oder Python in Maschinenbefehle übersetzt, müssen Zeilen- und Blockkommentare entfernt werden. Bei Zeilenkommentaren wird nur der Beginn gekennzeichnet und der Kommentar erstreckt sich bis zum Zeilenende. Blockkommentare sind Kommentare, deren Beginn und Ende gekennzeichnet sind und die sich über mehrere Zeilen erstrecken können.

Material 1 zeigt eine Methode in Python, in der sowohl Block- als auch Zeilenkommentare vorkommen. In der Programmiersprache Python werden Zeilenkommentare durch die Raute # eingeleitet, Blockkommentare beginnen und enden mit drei doppelten Anführungszeichen """.

Aufgaben

1. In Material 2 ist das Zustandsdiagramm eines endlichen Automaten gegeben. ε steht hier für das leere Wort, also „keine Ausgabe“, und X für ein beliebiges Zeichen mit Ausnahme des Anführungszeichens ".

   Analysieren Sie den Automaten und erläutern Sie seine Funktionsweise.

   (5 BE)

2.1 Entwerfen Sie das Zustandsdiagramm eines erkennenden endlichen Automaten, der überprüft, ob in einem gegebenen Python-Programmcode mindestens ein vollständiger Blockkommentar enthalten ist.

(4 BE)

2.2 Entwerfen Sie eine Grammatik für die Sprache, die der in Aufgabe 2.1 zu entwerfende Automat erkennt.

(4 BE)

2.3 Ordnen Sie Ihre Grammatik aus Aufgabe 2.2 in die Chomsky-Hierarchie ein.

(2 BE)

3. Geben Sie eine allgemeine Definition für erkennende endliche Automaten an und nennen Sie die Unterschiede zwischen diesen und dem in Material 2 gegebenen Automatentypen.

   (3 BE)

4. Ein Programmierer ärgert sich immer wieder darüber, dass die Programmiersprachen, mit denen er arbeitet, keine geschachtelten Blockkommentare erlauben. Dies würde ihm ermöglichen, bei der Fehlersuche in seinen Programmen ganze Bereiche auszukommentieren, auch wenn dort bereits Blockkommentare vorhanden sind.

4.1 Geben Sie an, zu welcher Ausgabe der in Material 2 gegebene Automat bei folgender Eingabe kommt und erläutern Sie das Ergebnis.

""" """Das ist ein Kommentar""" im Kommentar"""

(3 BE)

4.2 Der Programmierer entwickelt die eigene Programmiersprache Schako, in der beliebig tief geschachtelte Blockkommentare möglich sind. Blockkommentare beginnen mit """ und enden mit /"".

Entscheiden Sie, ob ein endlicher Automat geschachtelte Blockkommentare in Schako löschen kann.

(3 BE)

4.3 Blockkommentare in Schako-Programmen sollen von einer Turingmaschine gelöscht werden. Die Turingmaschine erhält Schako-Programme als Eingabe auf dem Band. Nach Beendigung des Vorgangs soll sich der kommentarfreie Programmcode nur noch getrennt durch Leerzeichen ␣ auf dem Band befinden und der Schreib/Lese-Kopf auf dem ersten Zeichen des Programmcodes stehen. Sie können davon ausgehen, dass zu jedem Blockkommentaranfang auch genau ein Blockkommentarende vorhanden ist und dass das Bandleerzeichen * im gesamten Schako-Programm nicht vorkommt.

Entwerfen Sie eine kommentierte Turingmaschine, die den beschriebenen Anforderungen gerecht wird. Verwenden Sie dabei das Zeichen x für alle Zeichen außer ␣," und /.

(6 BE)

Material 1

```
"""Die Methode "it_mult" berechnet iterativ das Produkt zweier Zahlen m und n und
gibt das Ergebnis zurück."""

def it_mult(m,n):

    if (n < 0) and (m >= 0):  # falls n negativ und m positiv ist
        a = m
        m = n
        n = a

    elif (m < 0) and (n < 0): # falls beide Faktoren negativ sind
        m = -m
        n = -n

    produkt = 0

# Produkt wird bei jedem Schleifendurchlauf um m erhöht

    for i in range(1, n+1):
        produkt += m

    return produkt
```

Material 2