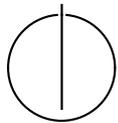


Name: Vorname: Matr.-Nr.:



TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK



Lehrstuhl für Sprachen und Beschreibungsstrukturen
Praktikum Grundlagen der Programmierung
Prof. Dr. Helmut Seidl

WS 2007/2008
08. Dezember 2007

Zwischenklausur zu Einführung in die Informatik I

Hinweis: In dieser Zwischenklausur können Sie insgesamt 30 Punkte erreichen. Zum Bestehen der gesamten Prüfung (Zwischen- und Endklausur) benötigen Sie mindestens 40 von insgesamt 100 Punkten.

Aufgabe 1 Syntax-Baum

(4 Punkte)

Zeichnen Sie den Syntax-Baum für das folgende MiniJava-Programm. Benutzen Sie dazu die Vorgabe auf der nächsten Seite.

Hinweis: Die Grammatik aus der Vorlesung befindet sich im Anhang.

```
int a,b;
a = read();
b = - a;
if (a >= 0 && b <= 0) {
    a = b / (a + 1);
}
```

```
int a, b; a = read(); b = -a; if ( a >= 0 && b <= 0 ) { a = b / ( a + 1 ) ; }
```

Aufgabe 2 Übersetzung von MiniJava nach MiniJVM

(4 Punkte)

Gegeben sei folgendes MiniJava-Programm:

```
int n, i, prod;
n = read();
i = 2;
prod = 1;
while (i <= n) {
    prod = prod * i;
    i = i + 1;
}
write(prod);
```

Übersetzen Sie das angegebene MiniJava-Programm in ein MiniJVM-Programm.

Hinweis: Die Übersetzungsregeln aus der Vorlesung befinden sich im Anhang.

Aufgabe 3 Arrays

(3+2=5 Punkte)

Schreiben Sie eine statische Methode, die ein `String-Array` `input` und ein `int-Array` `indices` als Argumente erhält. Sind die beiden Arrays nicht gleich lang, soll `null` zurückgegeben werden. Sind sie gleich lang, soll folgendes getan werden:

- a) Es soll überprüft werden, ob in dem Array `indices` jede Zahl von 0 bis `indices.length - 1` **genau** einmal vorkommt. Ist dies nicht der Fall, soll `null` zurückgegeben werden. Ansonsten wird mit b) fortgefahren.
- b) Es soll ein neues Array `result` erzeugt und zurückgegeben werden, in welchem an der Position `indices[i]` der String an der Position `i` des Arrays `input` steht.

Beispiel:

input:

"Eingabe"	"ist"	"Das"	"die"
-----------	-------	-------	-------

indices:

3	1	0	2
---	---	---	---

result:

"Das"	"ist"	"die"	"Eingabe"
-------	-------	-------	-----------

Aufgabe 4 Mengen

(1 + 2 + 2 + 2 = 7 Punkte)

Ziel dieser Aufgabe ist es, eine Klasse Menge zur Repräsentation von Mengen von Strings zu entwickeln. Implementieren Sie die Klasse Menge mit Hilfe einer **verketteten Liste** (Klasse List), in der jedes Element der repräsentierten Menge **genau einmal** vorkommt. Die Klasse Menge soll folgende Methoden bereitstellen:

- a) Einen Konstruktor zur Erzeugung einer **leeren** Menge.
- b) Eine Methode **boolean** istElement(String s), die überprüft, ob der String s in der Menge enthalten ist.
- c) Eine Methode **void** hinzufuegen(String s), die den String s zu der Menge hinzufügt.
- d) Eine Methode Menge vereinige(Menge m), die die Vereinigung der Menge **this** und m zurückliefert.

Die Klasse List:

```
public class List {
    public String info;
    public List next;

    public List(String s, List n) {
        info = s;
        next = n;
    }
    public List(String s) {
        this(s, null);
    }
}
```

Aufgabe 5 Parkhaus

(2+1+2+3+2=10 Punkte)

Ziel dieser Aufgabe ist es, ein Parkhaus zu simulieren.

- a) Implementieren Sie die Klasse `Auto` mit den privaten Attributen `marke` und `kennzeichen`, sowie den beiden Methoden `String getMarke()` und `String getKennzeichen()` und einem sinnvollen Konstruktor.
- b) Implementieren Sie die Klasse `Parkhaus`, die ein Parkhaus mit 20 Parkplätzen repräsentiert.
- c) Erweitern Sie die Klasse `Parkhaus` um die Methode `int parken(Auto a)`, bei der das `Auto a` den ersten freien Parkplatz des Parkhauses belegt. Die Methode soll dabei die Nummer des belegten Parkplatzes zurückliefern. Wenn das Parkhaus voll ist, soll `-1` zurückgegeben werden.
- d) Erweitern Sie die Klasse `Parkhaus` um die Methode `int suchen(String k)`, die mit Hilfe des Kennzeichens ein `Auto` im Parkhaus sucht. Ist das `Auto` mit dem Kennzeichen `k` im Parkhaus, soll die Methode die Nummer des Parkplatzes ausgeben, ansonsten `-1`.
- e) Erweitern Sie die Klasse `Parkhaus` um die Methode `Auto wegfahren(String k)`, das den vom `Auto` mit dem Kennzeichen `k` belegten Parkplatz wieder freigibt und das `Auto` zurückgibt. Falls das `Auto` mit Kennzeichen `k` nicht im Parkhaus ist, soll `null` zurückgegeben werden.

Anhang

Die Grammatik von MiniJava (aus der Vorlesung):

```
<program> ::= <decl>* <stmt>*
<decl> ::= <type> <name> ( , <name> )* ;
<type> ::= int
<stmt> ::= ; | { <stmt>* } |
          <name> = <expr>; | <name> = read(); | write( <expr> ); |
          if ( <cond> ) <stmt> |
          if ( <cond> ) <stmt> else <stmt> |
          while ( <cond> ) <stmt>
<expr> ::= <number> | <name> | ( <expr> ) |
          <unop> <expr> | <expr> <binop> <expr>
<unop> ::= -
<binop> ::= - | + | * | / | %
<cond> ::= true | false | ( <cond> ) |
          <expr> <comp> <expr> |
          <bunop> <cond> | <cond> <bbinop> <cond>
<comp> ::= == | != | <= | < | >= | >
<bunop> ::= !
<bbinop> ::= && | ||
```

<name> und <number> für Bezeichner und Zahlen werden nicht weiter verfeinert.

Anhang

Regeln aus der Vorlesung zur Übersetzung von MiniJava nach MiniJVM:

prog	=	ALLOC n Übersetzung von ss HALT — sofern prog aus einer Deklaration von n Variablen, gefolgt von der Statement-Folge ss besteht.
x	=	LOAD i — sofern x die Variable mit Adresse i ist.
c	=	CONST c — sofern c eine Konstante ist.
expr ₁ + expr ₂	=	Übersetzung von expr ₁ Übersetzung von expr ₂ ADD — Analog für die anderen Operatoren (SUB, MUL, DIV, MOD, AND, OR, LESS, LEQ, EQ, NEQ, GT, GEQ)
-expr	=	Übersetzung von expr NEG — Analog für NOT
x = expr;	=	Übersetzung von expr STORE i — sofern x die Variable mit Adresse i ist.
x = read();	=	READ STORE i — sofern x die Variable mit Adresse i ist.
write(expr);	=	Übersetzung von expr WRITE
if (cond) stmt	=	Übersetzung von cond FJUMP A Übersetzung von stmt A: ...
if (cond) stmt ₁ else stmt ₂	=	Übersetzung von cond FJUMP A Übersetzung von stmt ₁ JUMP B A: Übersetzung von stmt ₂ B: ...
while (cond) stmt	=	A: Übersetzung von cond FJUMP B Übersetzung von stmt JUMP A B: ...
stmt ₁ ... stmt _k	=	Übersetzung von stmt ₁ ... Übersetzung von stmt _k