Technische Universität München
Fakultät für Informatik
Prof. Dr. Seidl

WS 2004/2005
Übungsblatt 2
4. November 2004

Übungen zu Einführung in die Informatik I

Aufgabe 5 Umwandlung zwischen Binär und Dezimal

Schreiben Sie ein Programm, das:

a) eine zur Basis 10 dargestellen Zahl einliest und deren Darstellung zur Basis 2 ausgibt. Zum Beispiel sollte für dreizehn 1101 ausgegeben werden.

b) eine zur Basis 2 dargestellten Zahl einliest und deren Darstellung zur Basis 10 ausgibt.

Aufgabe 6 Sieb des Eratosthenes

Folgendes Verfahren zur Primzahlen-Gewinnung ist als **Sieb des Eratosthenes** bekannt:
(*Nach Eratosthenes von Kyrene, hellenistischer Gelehrter, um 276-195 v. Chr.*)

Aus der Menge der natürlichen Zahlen von 2 bis n werden alle Nicht-Primzahlen gestrichen. Beginne mit der Zahl 2 und streiche alle echten Vielfachen von 2. Streiche nun sukzessiv alle echten Vielfachen der jeweils nächsthöheren noch nicht gestrichenen Zahl. Nach Streichung aller Vielfachen enthält die Restmenge nur noch Primzahlen.

Implementieren Sie das oben beschriebene Verfahren in Java mit Hilfe eines Arrays. Ihr Programm soll nach Festlegung einer Obergrenze durch den Benutzer alle Primzahlen bis zu dieser Grenze bestimmen und ausgeben.

Aufgabe 7 Grammatiken

Welche Sprachen werden von den folgenden Grammatiken beschrieben?

a) S ::= aSa | bSb | c

b) S ::= aSd | aAd
A ::= bAc | bc

Bitte wenden!

Aufgabe 8 Syntax-Baum

Zeichnen Sie für die beiden MiniJava-Programme (i) und (ii) je einen Syntax-Baum.

Aufgabe 9 Kontrollfluss-Diagramm

Konstruieren Sie für die beiden Programme (i) und (ii) jeweils ein Kontrollfluss-Diagramm.

(i)

```
int a,b;
a = read();
b = 29;
if( 1 < a && (a*3) < b ) {
    write(1);
}
else
    write(0);
```

(ii)

```
int x, r, n;
r = 1;
n = 1;
x = read();
while( n < x ) {
    if( n % 2 == 0 )
        r = r * n;
    else
        r = r * (-n);
    n = n + 1;
    write(r);
}
```

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>	::=	&& | ||