Informatik III WS2004/2005

Übungsblatt 13

Institut Intelligente Systeme
Leiter: Prof. Dr. E. Lehmann
Dietmar Lippold

Besprechungstermin: 07.02.05 - 11.02.05

Aufgabe 1

In der Vorlesung wurde die objektorientierte Programmierung im Allgemeinen besprochen und mit Java eine objektorientierte Sprache vorgestellt.

1. Was sind die wesentlichen Konzepte der objektorientierten Programmierung (spezielle Eigenschaften objektorientierter Sprachen)?
2. Erläutern Sie die Begriffe *Klasse, Objekt, Instanz, Exemplar* sowie *Attribut, Variable, Nachricht, Methode, Prozedur, Funktion.*
3. Was versteht man unter *statischer Polymorphie* bzw. *Überladen* (*overloading*) und unter *dynamischer Polymorphie* bzw. *Überlagern* bzw. *Überschreiben* (*overriding*)?

Aufgabe 2

Bereits 1978 schrieben Brian Kernighan und Dennis Ritchie in ihrem Klassiker *The C Programming Language: "the first program to write is the same for all languages"*. Damit meinten sie natürlich das *Hello World*-Programm (ein Programm, das nur den Text *Hello World* ausgibt). Durch dieses einfache Beispiel–Programm kann man sich u.a. mit dem allgemeinen Aufbau eines Programms einer bestimmten Programmiersprache vertraut machen und kann die Schritte zur Ausführung des Programms kennen lernen.

1. Schreiben Sie das *Hello World*-Programm.
2. Übersetzen Sie das Programm durch den Aufruf des Compilers mit dem Kommando `javac HelloWorld.java` in Bytecode und beheben Sie eventuell vorhandene Fehler.
3. Führen Sie das *Hello World*-Programm durch das Kommando `java HelloWorld` aus. Das Kommando `java` startet die Java-Virtual-Machine (*JVM*) mit dem Bytecode-Interpreter. Die JVM lädt die angegebene Klasse und beginnt sofort mit der Ausführung von deren `main`-Methode.

Aufgabe 3

Untersuchen Sie die Arbeitsweise der folgenden Prozedur:

```
(define (mysterious x)
  (define (schleife x y)
    (if (null? x)
        y
        (let ((temp (cdr x)))
          (set-cdr! x y)
          (schleife temp x))))
  (schleife x '()))
```

1. Betrachten Sie den Prozeß für den Aufruf `(mysterious '(a b c d))`. Erstellen Sie ein Protokoll der Aufrufe der inneren Prozedur `schleife` mit den Werten der Argumente, zeichnen Sie für jeden Aufruf ein Box-Pointer-Diagramm der beiden Argumente und notieren Sie die Werte der in der Prozedur erzeugten und veränderten Variablen.
2. Was tut `mysterious`?
3. Handelt es sich um einen iterativer oder um einen rekursiven Prozeß? Ist die Prozedur destruktiv? Begründen Sie Ihre Antworten.
4. Erstellen Sie eine funktional äquivalente nicht–destruktive, rekursive Prozedur, aus der ein iterativer Prozeß erzeugt wird.
5. Geben Sie für die gegebene Prozedur und für Ihre Prozedur den Zeitaufwand und den Aufwand an neu erzeugten Cons-Zellen in O-Notation abhängig von der Listenlänge an.

Aufgabe 4 (schriftlich)

Ein einfacher unsymmetrischer Patternmatcher soll einen geschachtelten Ausdruck (Liste) mit einem Muster vergleichen. Falls das Muster eine Verallgemeinerung des Ausdrucks ist, soll als Wert `#t`, sonst `#f` geliefert werden.

```
> (match '(a (x y)) '(a (x y)))
#t
> (match '(a b c) '(a b))           > (match '(a b) '(a b c))
#f                                  #f
> (match '(a (x y) c) '(a ? c))     > (match '(a (x y z) c) '(a (x ? z) c))
#f                                  #t
> (match '(a c) '(a * c))           > (match '(a b c) '(a * d))
#t                                  #f
> (match '(a b (anna (eva ina) petra) c) '(a * c))
#t
```

1. Erstellen Sie einen Patternmatcher `(match ausdr pat)`, der im Pattern das Atom-Wildcard `?` zulässt. Es steht für genau ein Atom.
2. Erweitern Sie den Pattermatcher um den Segment-Wildcard `*`. Es steht für beliebig viele (auch geschachtelte) Elemente.

Aufgabe 5

Es soll die Menge aller Wörter (Folgen) der Länge k, die man aus einer Grundmenge (Alphabet) bilden kann, berechnet werden. Die Elemente des Alphabets sind als Symbole, die Wörter und die Grundmenge als Listen dargestellt.

```
> (k-woerter '(A B C) 3)
((A A A) (A A B) (A A C) (A B A) (A B B) (A B C) (A C A) (A C B) (A C C)
 (B A A) (B A B) (B A C) (B B A) (B B B) (B B C) (B C A) (B C B) (B C C)
 (C A A) (C A B) (C A C) (C B A) (C B B) (C B C) (C C A) (C C B) (C C C))
```

1. Erklären Sie in Worten, wie man aus der Menge aller Kombinationen der Länge k die Menge der Kombinationen der Länge $k + 1$ erzeugen kann.
2. Definieren Sie eine Prozedur `(k-woerter alpha k)`, die die Liste aller Wörter der Länge k aus der Grundmenge `alpha` zurückgibt.