Übung zur Vorlesung Compiler 1: Grundlagen

Prof. Dr. Andreas Koch
Jens Huthmann, Florian Stock

TECHNISCHE UNIVERSITÄT DARMSTADT

Wintersemester 11/12
Aufgabenblatt 3

Abgabemodalitäten

Die Aufgabe sind in 2er Gruppen zu bearbeiten. Ihre Lösungen reichen Sie als PDF-Datei per E-Mail bis zum 26.01.2012 um 23:59 MEZ an der Adresse oc@esa.informatik.tu-darmstadt.de ein. Die E-Mail muss als Betreff "Compiler 1 Aufgabe 3" haben. Geben Sie in Ihrem Lösungsdokument den Namen und die Matrikelnummer aller Gruppenmitglieder an.

Aufgabe 3.1 Allgemeine Fragen — 9 Punkte

a) Beschreiben Sie die unterschiedliche Verwendung des Stacks und Heaps in der Speicherverwaltung

b) Schildern Sie die Unterschiede in der Verwendung des static bzw. dynamic links.

c) Was ist die Aufgabe des Routinenprotokolls und warum ist es wichtiger Bestandteil der Runtime-Spezifikation?

Aufgabe 3.2 Rechnung zur Stack-Maschine — 16 Punkte

Geben Sie für die folgenden Ausdrücke die zur Auswertung notwendigen Instruktionen in der richtigen Reihenfolge an. Gehen Sie analog zu den Folien 32 ff. des 4. Vorlesungsblocks vor. Optimieren Sie, falls möglich, in Richtung einer möglichst kurzen Instruktionssequenz. Erlaubt sind alle Operationen von Folie 33.

a) $x := 2 * a * b + 2 * a * c + 2 * b * c$

b) $y := 11 + a * (7 + a * (-5 + a * (-4 + a * 2)))$

Aufgabe 3.3 Statische Speicherverwaltung — 20 Punkte

Nachfolgend finden Sie die zur Definition eines Schachspiels notwendigen Datentypen. Geben Sie für dieses Programm die Speicherorganisation der Variable *state* gemäß Folie 40 des 4. Vorlesungsblocks an. Ihre Lösung sollte für jeden Wort-Index des Speichers, beginnend ab der fiktiven Anfangsadresse 0, dessen benutzerdefinierten- (Piece, Player...) und Basistyp (Integer, Boolean, Char) angeben. Markieren Sie auch welche Worte zu welchem Record bzw. Array gehören.

```
let
  type Player ~ record
    number : Integer
  end;
  type Piece ~ record
    symbol : Char
  end;
  type Square ~ record
    empty : Boolean,
    occupant : Piece,
    owner : Player
  end;
  type Board ~ record
    board : array 64 of Square
  end;
  type State ~ record
```

```
    board : Board,
    next : Player,
    moves : Integer
  end;
  var state : State
in
  ...
```

Listing 1: Beispiel zur statischen Speicherverwaltung

Aufgabe 3.4 Routinenprotokoll — 30 Punkte

Das nachfolgende Programm implementiert die notwendigen Funktionen zur Berechnung von $power(x, n) = x^n$. Zeichnen Sie hierfür stack frames, die Argumente und Resultate vor Eintritt in eine Funktion und nach Austritt aus einer Funktion zeigen für die Auswertung folgender Ausdrücke:

a) $power(2, 1)$

b) $power(3, 3)$

```
let
  func sqr (i : Integer) : Integer ~
    i * i;

  func even (i : Integer) : Boolean ~
    ((i // 2 ) = 0);

  func power (x : Integer, n : Integer) : Integer ~
    if n = 0 then
      1
    else if even(n) then
      sqr(power(x, n / 2))
    else
      sqr(power(x, n / 2)) * x
in
  ...
```

Listing 2: Beispielprogramm zum Routinenprotokoll

Plagiarismus

Der Fachbereich Informatik misst der Einhaltung der Grundregeln der wissenschaftlichen Ethik großen Wert bei. Zu diesen gehört auch die strikte Verfolgung von Plagiarismus. Weitere Infos unter www.informatik.tu-darmstadt.de/plagiarism