3. Sprachkonzepte und ihre Übersetzungen

Übersetzung von Wertzuweisungen:

Funktion	Bedingung
codeR x p = codeL x p; ind *T*	x Variablenbez. vom Typ *T*
codeR c p = ldc *T* c	c Konstante v. Typ *T*
codeR (e1 = e2) p = codeR e1 p; codeR e2 p; equ *T*	Typ(e1)=Typ(e2)=*T*
codeR (e1 / e2) p = codeR e1 p; codeR e2 p; div *N*	Typ(e1)=Typ(e2)=*N*
codeR(-e) p = codeR e p; neg *N*	Typ(e)=*N*
code(x := e) p = codeL x p; codeR e p; sto *T*	Typ(e)=*T, x* Variablenbez.
codeL x p = ?? Übungsaufgabe	*??*

3. Sprachkonzepte und ihre Übersetzungen

Beispiel Übersetzung:

x,y,z integer-Variablen, p(x)=5, p(y)=6, p(z)=7

code(x := (y +(y*z))) p

= codeL x p; codeR(y + (y * z)) p; sto i

= ldc a 5; codeR(y) p; codeR(y*z); add i; sto i

= ldc a 5; codeL y p; ind i;codeR(y*z);add i; sto i

= ldc a 5; ldc a 6;ind i; codeR(y) p; codeR(z) p; mul i;add i; sto i

= ldc a 5; ldc a 6; ind i; ldc a 6; ind i; codeR(z) p; mul I, add i; sto i

= ldc a 5; ldc a 6; ind i; ldc a 6; ind i, ldc a 7; ind i; mul i, add i; sto i

code(b:= (((x-y) > 0)) or (x=z)))

= ?? Übungsaufgabe

int x,y,z; bool b, p(b)=3, p(x)=4

p(y)=5, p(z)=6

3. Sprachkonzepte und ihre Übersetzungen

P-Befehl bei **Kontrollstrukturen**:

if e **then** st1 **else** st2 **fi** bzw. **if** e **then** st **fi**

while e **do** st **od**

repeat st **until** e

Befehl	Bedeutung	Beding.
ujp q	PC := q (unbedingter Sprung)	q Є[0,codemax]
fjp q	If STORE[SP] = false then PC := q fi; SP := SP -1; (bedingter Sprung)	(b) q Є[0,codemax]

3. Sprachkonzepte und ihre Übersetzungen

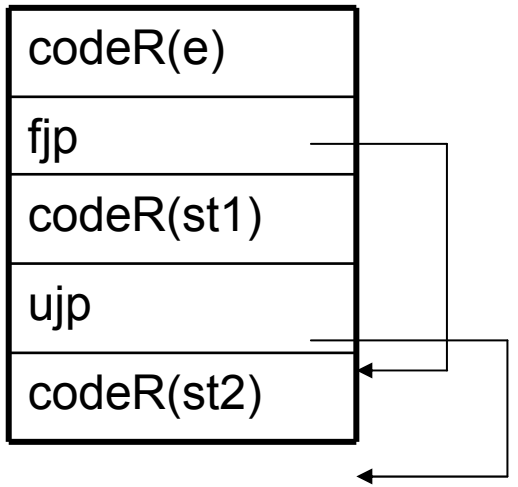
Übersetzung bei Kontrollstrukturen:

code(if e then st1 else st2 fi) p

= codeR e p; fjp *m1;* code st1 p; ujp *m2*; *m1:* code st2 p; *m2:*

m1:, m2: Marken: markieren Befehle oder Ende des Schemas

Ziel eines Sprungbefehls



code(if e then st fi) p

= ?? Übungsaufgabe

3. Sprachkonzepte und ihre Übersetzungen

Übersetzung bei Kontrollstrukturen:

code(while e do st od) p

= *m1:* codeR e p; fjp *m2;* code st p; ujp *m1*; *m2:*

code(repeat st until e) p

= *m1:* code st p; codeR e p; fjp *m1*;

code(**while** a > b **do** c:= c+1; a:= a-b **od**) p

= ?? Übungsaufgabe (mit p(a)=5, p(b)=6, p(c)=7)

code(**for** i:=1 **to** 10 **do** st **od**) = ?? Übungsaufgabe (mit p(i)=5)

3. Sprachkonzepte und ihre Übersetzungen

P-Befehl bei case:

case e **of**

0: st0;

1: st1;

…

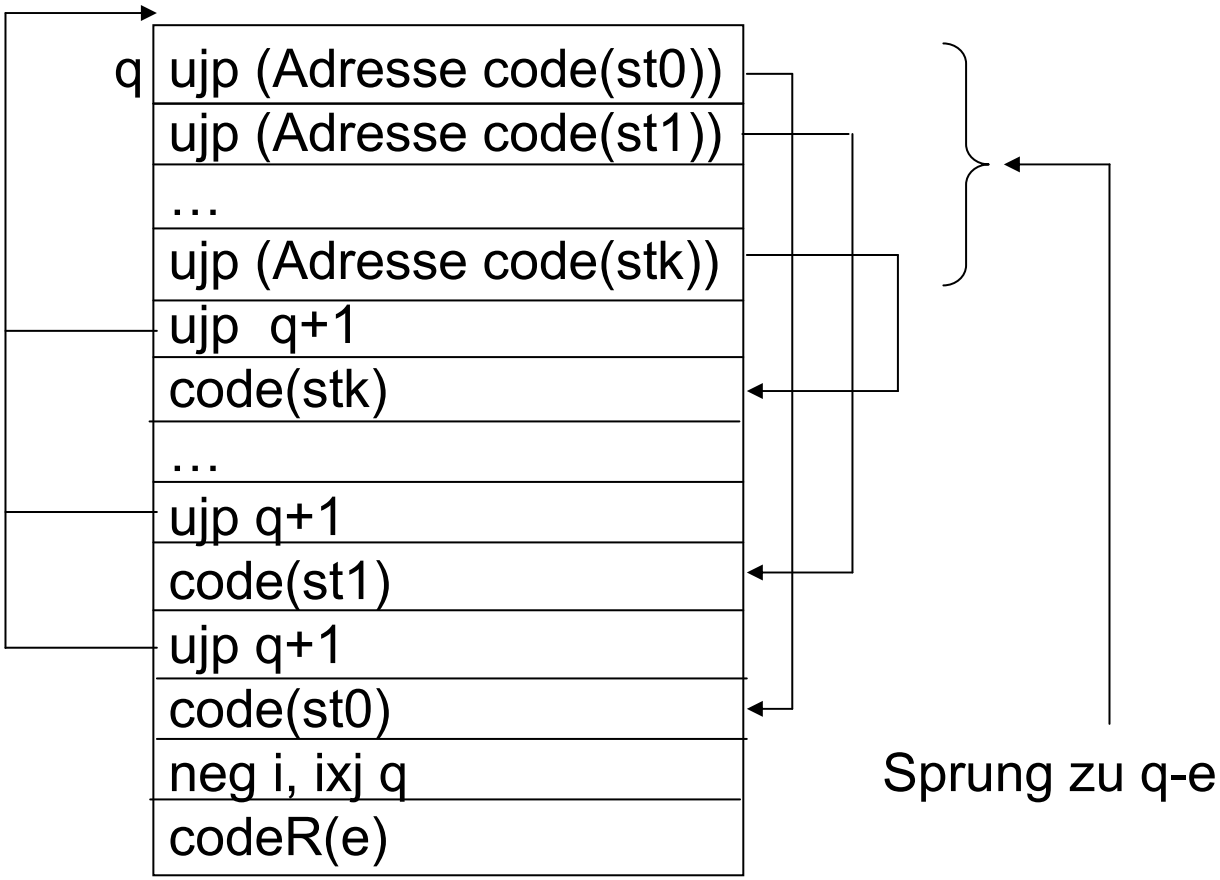
k: stk;

end

Befehl	Bedeutung	Beding.
ixj q	PC := STORE[SP] + q; SP := SP-1; (indizierter Sprung)	(i)

3. Sprachkonzepte und ihre Übersetzungen

Codegenerierung (Skizze) bei case:



3. Sprachkonzepte und ihre Übersetzungen

Übersetzung des case – mittels Marken:

```
codeR e p; neg i; ixj q;          // q ist dabei abhängig von der
mo: code(st0) p; ujp mz;                    // Größe und
m1: code(st1) p; ujp mz;          // Anzahl der case-Fälle
 … ;
mk: code(stk) p; ujp mz;
ujp mk;…; ujp m1; ujp m0;
mz:
```

3. Sprachkonzepte und ihre Übersetzungen

Übersetzung von **Variablen**:

Übersetzungszeit: Überführung Pascal-Programm in P-Programm

Laufzeit: Ausführung P-Programm mit Eingabedaten

Informationen

statisch:	zur Übersetzungszeit aus Programmcode ersichtlich/erzeugbar
dynamisch:	erst zur Laufzeit durch Ausführung verfügbar

3. Sprachkonzepte und ihre Übersetzungen

Übersetzung von Variablen:

hier:

- Speicherzelle: nimmt einen einfachen Typ (integer, real, bool, char, Zeiger-, Aufzählungs-, Mengentyp) auf
- Wortlänge der P-Maschine nicht festgelegt

Reale Übersetzer:

- „packen“ mehrere kleine Werte in ein Wort
- hohe Genauigkeit durch reelle Zahlen mit langer Mantisse

Speicherbelegung:

- pro Variable einfachen Typs eine Adresse im Kellerspeicher (fortlaufend; ab Adresse 5)
- Auffassen als relative Adresse (ab Basis 0)
- $p(n_i) = i+5, i \geq 0$

3. Sprachkonzepte und ihre Übersetzungen

Übersetzung von **statischen Feldern**:

var f: **array**[1..5,1..10] **of integer**;
var feld: **array**[u1..o1,…,uk..ok] **of integer**;

Speicherbelegung:

- zeilenweise Ablage
- Funktion p: Adresse der ersten belegten Zelle
- Funktion gr: liefert Größe eines Typs
  gr(T)=1, wenn T integer, real, char, …
  gr(T)= $\prod$(oi – ui +1) für i=1 bis k wenn T obiges Feld

3. Sprachkonzepte und ihre Übersetzungen

Übersetzung von statischen Feldern:

- Adresse einer Variablen ni: p(ni)= 5 + ∑ gr(Tj), j=1 bis i-1
- Alle Größen, p, gr zur Übersetzungszeit bekannt
- Speichern des Wertes 0 in erste Zelle eines Feldes f:

  ldc a p(f);

  ldc i 0;

  sto i;
- Speichern des Wertes 0 in f[i,j], i,j erst zur Laufzeit bekannt??

  Lösung:

  aktuelle Werte i^ und j^ mithilfe von p(i) und p(j) laden,

  Zieladresse: (i^ – u1) * (o2 – u2 +1) + j^ – u2 + p(f)

3. Sprachkonzepte und ihre Übersetzungen

hilfreiche Befehle:

Befehl	Bedeutung	Beding.	Ergebn.
ixa *q*	STORE[SP-1] := STORE[SP-1] + STORE[SP]*q SP:= SP-1;	(a, i)	(*a*)
inc *T q*	STORE[SP] := STORE[SP] + q	(*T*),Typ(q)=*i*	(*T*)
dec *T q*	STORE[SP] := STORE[SP] - q	(*T*),Typ(q)=*i*	(*T*)
chk *p q*	**if**(STORE[SP]< p) or (STORE[SP]>q) **then** error(„value out of range“) fi;	(a,*T*)	

codeL c[i1,..ik] p = ldc a p(c); codeI [i1,..,ik] g p //g Komponentengröße

codeI [i1,..,ik] g p = codeR i1 p; chk u1 o1; ixa g*d(1);
…;
codeR ik p; chk uk ok; ixa g*d(k);
dec a g*d;

3. Sprachkonzepte und ihre Übersetzungen

Beispiel:

```
var i,j:integer;                                  //p(i)=5, p(j)=6
    feld: array[1..5,1..10] of integer;           //p(feld)=7
```

Spanne d1: 5-1+1=5, d2=10-1+1=10, d(1)=10, d(2)=1

Komponentengröße g = 1(Integer)

d = u1*d2+u2 = 1*10+1=11

code(feld[i+1,j]:=0) p = ldc a 7; ldc a 5; ind i; ldc i 1; add i;
chk 1 5; ixa 10;
ldc a 6; ind i; chk 1 10; ixa 1;
dec a 11;
ldc i 0;sto i;

Allgem.: 1. d(1) = d2*d3*…*dk; d(2) = d3*d4*…*dk ,…, d(k) ist immer 1

2. d = u1*d2*..*dk+ u2*d3*..*dk+uk-1*dk+uk