

ALBERT-LUDWIGS-UNIVERSITÄT FREIBURG

BETRIEBSSYSTEME

Blatt 05



Students:

Julian Polzer jp390, David Janzen dj57

Tutor:

GRUPPE 7

1 Aufgabe

2 Aufgabe

1. Symboltabelle (st) bei start bds = 128:

st(x) = (var, int, 128)

st(y) = (var, int, 129)

st(z) = (const, int, 5)

2. Befehlsfolge ReTi

Die einzelnen Schritte:

1. x abspeichern und SP um eins senken
2. y abspeichern und SP um eins senken
3. z abspeichern und SP um eins senken
4. $z * y$ rechnen und SP um eins heben
5. 10 abspeichern und SP um eins senken
6. $10 + (z * y)$ rechnen und SP um eins heben
7. $x + (10 + (z * y))$ rechnen und SP um eins heben

Befehl	Schrittzugehörigkeit
SUBI SP 1	1
LOAD ACC 128	1
STOREIN SP ACC 1	1
SUBI SP 1	2
LOAD ACC 129	2
STOREIN SP ACC 1	2
SUBI SP 1	3
LOAD ACC (130 / 5)	3
STOREIN SP ACC 1	3
MOVE ACC IN1	4
LOAD ACC [SP + 2]	4
ADD ACC IN1	4
STOREIN SP ACC 2	4
LOAD ACC 10	5
STOREIN SP ACC 1	5
MOVE ACC IN1	6
LOAD ACC [SP - 2]	6
ADD ACC IN1	6
STOREIN SP ACC 1	6
ADDI SP 1	6
MOVE ACC IN1	7
LOAD ACC [SP - 2]	7
ADD ACC IN1	7
STOREIN SP ACC 1	7
ADDI SP 1	7

3. Um den Stack maximal zu benutzen klammert man wie folgt:

$(x_1 + (x_2 + (x_3 + (\dots + (x_n))))))_n$

Um den Stack minimal zu benutzen klammert man wie folgt:

$$({}_n(((x_1 + x_2) + x_3) .. + x_n))$$

3 Aufgabe

ReTi Code	explanation
1. LOAD ACC 10	x in den ACC laden
2. JUMP< +3	Falls $x < 0 \Rightarrow 3$ Befehle nach vorne springen
3. LOAD ACC 11	y in den ACC laden
4. JUMP< +3	Falls $y < 0 \Rightarrow 3$ Befehle nach vorne springen
5. LOAD ACC 10	x in den ACC laden
6. JUMP \geq +8	springen falls $y < 0$ und $x \geq 0$
7. LOAD ACC 11	y in den ACC laden
8. JUMP \geq +5	springen falls $x < 0$ und $y \geq 0$
9. LOAD ACC 10	x in den ACC laden
10. SUB ACC 11	x - y im ACC rechnen
11. JUMP \leq +2	in dem Fall ist $x \leq y$
12. JUMP +2	in diesem Fall ist $x > y$
13. LOADI ACC 1	Soll hier den return von 1 darstellen
14. LOADI ACC 0	Soll hier den return von 0 darstellen