Betriebssysteme WS22/23 Blatt 6

 ${\it Malte Pullich, Daniel Augustin} \\ 01.12.2022$

Nummer 1

st(x) = (var, int, 128) st(y) = (var, int, 129)st(z) = (const, int, 2) 6+1 weil ihr eine der wenigen Gruppen seid die es korrekt gemacht haben = 7 /6 perfekt

```
1 \mid : v = 3
 2
  SUBI SP 1
                         Enlarging the SP
  LOADI ACC 3
                        Loading 3 into ACC
   STOREIN SP ACC 1
                          Saving 3 on SP
5
   LOADIN SP ACC 1
                         Loading SP into ACC
   ADDI SP 1
                        ; Deleting value 3 from SP
  STORE ACC 129
7
                       ; Storing in place for var y
   ; x = 15
   SUBI SP 1
   LOADI ACC 15
                         Loading 15 into acc
  STOREIN SP ACC 1
                         Saving 15 on SP
  LOADIN SP ACC 1
12
   ADDI SP 1
13
  STORE ACC 128
                       ; Storing in place for var x
   ; Saving value of x to SP
  SUBI SP 1
  LOAD ACC 128
                        ; Loading value for x
17
                        ; Saving value for x on SP
   STOREIN SP ACC 1
   ; Saving value of z o SP
19
   SUBI SP 1
20
21
   LOADI ACC 2
                       ; Loading 2 (const z) into ACC
   STOREIN SP ACC 1
                        ; Saving 2 on SP
   ; Saving value of y to SP
   SUBI SP 1
24
   LOAD ACC 129
                         Loading value for y
   STOREIN SP ACC 1
                      V; Saving value for y on SP
   ; Checking loop condition
28
   ; Computing z * y
   LOADIN SP ACC 2
                          Loading value of z from SP into ACC
   LOADIN SP IN2 1
                        ; Loading value of y from SP into IN2
   MUL ACC IN2
                         Computing z * y
   STOREIN SP ACC 2
                        ; Saving z * y on SP, overwriting z
   ADDI SP 1
                        ; Deleting value of y from SP
   ; Checking if x >= (z*y)
34
   LOADIN SP ACC 2
                         Loading value of x from SP into ACC
                          Loading the result of z * y into IN2
   LOADIN SP IN2 1
  SUB ACC IN2
                         Computing x - z * y
38 JUMP< 3
                         If x - z * y < 0 Jump to saving
```

```
39
                        ; 0/FALSE as result
40
   LOADI ACC 1
                        ; Saving 1 or TRUE as result
   JUMP 2
41
                        ; Skip saving 0/FALSE as result
   LOADI ACC 0
                        ; Saving 0 or FALSE as result
42
   STOREIN SP ACC 2
                        ; Saving the result of the statement
   ADDI SP 1
                        ; Cleaning the SP
44
   ; Checking if the statement was correct
   LOADIN SP ACC 1
46
47
   ADDI SP 1
                          Cleaning SP
   JUMP= 15
48
                         ; If statement result = 0/FALSE
49
                        ; jump to end of code
50
   ; Code inside the loop
   SUBI SP 1
51
   LOAD ACC 128
                          Loading the value of x
   STOREIN SP ACC 1
53
                          Saving x on stack
   SUBI SP 1
54
   LOADI ACC 3
55
                         ; Loading constant 3
   STOREIN SP ACC 1
                        ; Saving const 3 on stack
   LOADIN SP ACC 2
                          Loading value of x from stack
57
   LOADIN SP IN2 1
                        ; Loading vlaue of 3 from stack
58
59
   SUB ACC IN2
                        ; computing x - 3
60
   STOREIN SP ACC 2
                          Saving the result on the SP of x
61
   ADDI SP 1
                          Removing 3 from SP
62
   LOADIN SP ACC 1
                          Loading value for x from SP
63
   STORE ACC 128
                          Storing new value in x
   JUMP - 26
64
                         ; Jumping to the start of the Loop
```

jumpt nicht weig genug zurück, aber das ständig nachzändern ist auch etwas zu viel

ich glaub ihr hattet davor an die richtige Stelle gezeigt und nur jetzt ist es nicht mehr der Fall

Nummer 3 a

```
a = \&(p2.x);
//Speichert die Adresse 15 in die Variable von a (Adresse 10)
// Speichert den Wert 7 in der Adresse 15
p2.y = 4;
// Speichert den Wert 4 in der Adresse 16
p1 = (struct point *) malloc (size of (struct point));
//Speichert die Adresse 33 in die Variable von p1 (Adresse 8)
(*p1).y = *a;
```

die b) beantwortet ja iwie auch schon direkt die a)

Speichert den Wertes aus a (Wert 7) in die Adresse 34

Speichert die Adresse 33 aus p1 in Variable p3

p1 = &p2;

Speichert die Adresse 15 von p2 in die Variable p1

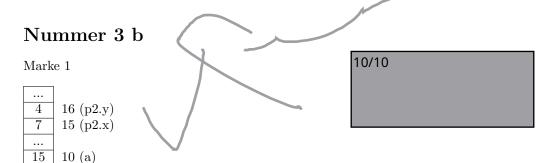
if ((*p1).y > 5)

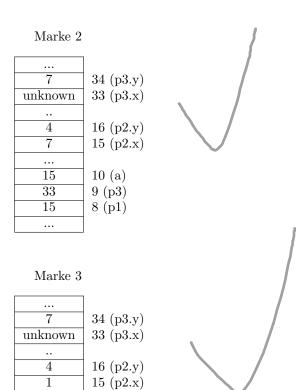
Auswerten ob der an der Adresse 16 (Wert 4) > 5 ist nur der else teil relevant

Speichert den Werts 1 in die von Variable a referenzierte Adresse 15

free(p3);

Ubergabe der in der Variable p3 referenzierten Adresse 33 an free()





Nummer 3 c

15

33

15

10 (a)

9 (p3) 8 (p1)

Die letzte Anfrage free
(p3) ist zulässig und die Adressen 33 und 34 werden freigegeben