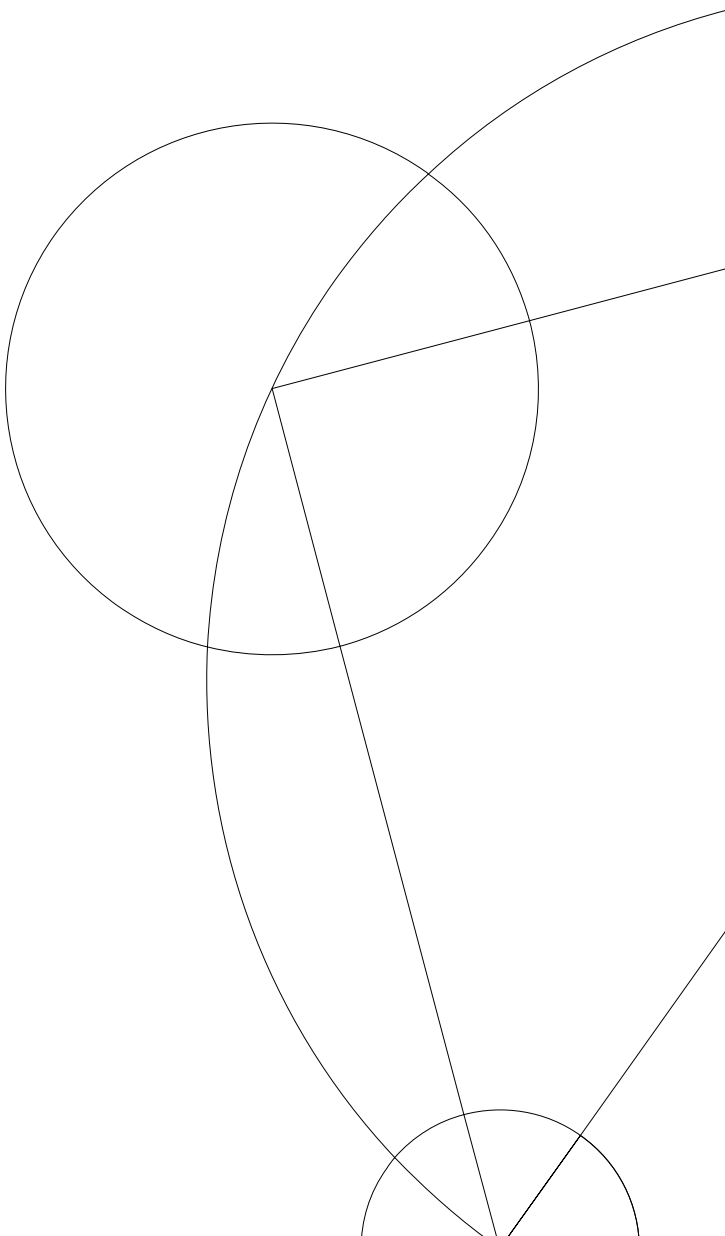
KØBENHAVNS UNIVERSITET

Assignment 3

Adit (hjg708)

IPS



17. maj 2023

Task 1

Intermediate code:

```
    t0 = v0
    t1 = v1

L1:
    t2 = t0
    t3 = t1
    t4 = 0
    if t3 == t4 then endlab else L2

L2:
    t5 = t2 / t3
    t6 = 1
    if t6 < t5 then L3 else endlab

L3:
    if t3 < t2 then L4 else L5

L4:
    t0 = t2 - t3
    goto L1

L5:
    t1 = t3 - t2
    goto L1

endlab:
```

MIPS:

```
    lw      $t0, v0
    lw      $t1, v1

L1:
    mv      $t2, $t0
    mv      $t3, $t1
    li      $t4, 0
    beq     $t3, $t4, endlab

L2:
    div     $t5, $t2, $t3
    li      $t6, 1
    sltu    $t7, $t6, $t5
    beqz    $t7, endlab

L3:
    slt     $t8, $t3, $t2
    beqz    $t8, L5

L4:
    sub     $t0, $t2, $t3
    j       L1

L5:
    sub     $t1, $t3, $t2
    j       L1

endlab:
```

Task 2

Make RISC-V pattern/replacement pairs as in the ”5 - Machine Code Generation”lecture for each of the following two IL instructions.

IL pattern	replacement
$z := x \geq y$	slt r_d, r_x, r_y xori $r_z, r_d, 1$
$w :=!z$	xori $r_w, r_z, 1$
$z := x \geq y$ $w :=!z$	slt r_z, r_x, r_y xori $r_w, r_z, 1$
$z := x \geq y$ $w :=!z^{last}$	slt r_w, r_x, r_y

Task 3

a)

The scan function applies a binary operation to each element of the input array x, accumulating the results in a new array b, which is then returned as the output of the function.

```
bool* scan(bool myop(bool, bool), bool ne, bool* x) = {
    int len = length(x);
    bool* y = malloc(len*4);
    int i = 1;
    y[0] = myop(ne, x[0])
    while(i < len) {
        bool tmp = myop(y[i-1], x[i]);
        y[i] = tmp;
        i = i + 1;
    }
    return y;
}
```

I tested my implementation of the scan function by using the same inputs as provided in the group project description: scan(plus, 0, {1,2,3,4}). I did a step-by-step calculation for that specifik, and in the end I ended up with the same output as the example, which is {1,3,6,10}

b)

```
lw   R_len, 0(R_x)
mv   R_y, R_HP

slli  R_tmp, R_len, 2
addi R_tmp, R_tmp, 4
add  R_hp, R_hp, R_tmp
sw   R_len, 0(R_y)

addi R_ix, R_x, 4
addi R_iy, R_y, 4
mv   R_i, zero

loop_beg:
  sub   R_tmp, R_i, R_len
  bgez  R_tmp, loop_end

  lw    R_tmp, 0(R_ix)
  addi  R_ix, R_ix, 4

  mv    R_clen, 8
  mv    R_c, R_hp
  add   R_hp, R_hp, R_clen
  sw    R_clen, 0(R_c)
  sw    R_ne, 4(R_c)
  sw    R_temp, 8(R_c)
  jal   myop

  sw    R_tmp, 0(R_iy)
  addi  R_iy, R_iy, 4
  addi  R_i, R_i, 1
  j     loop_beg

loop_end:
```

I used the MIPS code from slide 26 in 'Machine Code Generation'. Where I then implement the missing part with the 'myop' function.