HW 3.4 Branch If (C to ARM)

a → r0, b → r1

```
if ((a<-2) && (b!=1))
    {
        b=b+9;
    }
```

两个都需要成立

⇓

```
    cmp r0, -2
    bge , L2
L0:
    cmp r1, 1
    beq   L2
L1: add r1, r1, 9
L2:
```

HW 3.5 Branch if (ARM to C)

```
  cmp r0, -6
  ble .L1
.L0:
  cmp r1, -9
  bne .L2
.L1:
  add r0,r0,8
.L2:
```

成立直接a=a+8
说明中间是 or "||"

如果不成立跳到 end
说明应该是 bne 的 negation
"=="

```
if (a <= 6) || (b == -9)){
   a=a+8;
}
```

HW 3.6 Branch If (c to ARM)

```
if (((a≥9)||(b==5)) && (c>8))
{
    c = c-1;
}
```

```
    cmp r0, 9
    bge L1
.L0:
    cmp r1, 5
    bne .L2
.L1:
    cmp r2, 8
    ble .L3
.L2:
    sub r2, r2, 1
.L3:
```

Hw 3.7

```
if (a>=-5)
    { a=a+8;
  }
  else
        b=4;
  }

Cmp  r0, -5
ble  .L1
.L0: add r0, r0, 8
     b    .L2
.L1: mov r1, 4
.L2:
```

HW 3.8

```
if (a != 5) && (b >= 5)
    { b = b - 3; }
  else  c = c - 1;
  }

cmp  r0, 5
  beq    .L2
.L0:
   cmp  r1, 5
 blt     .L2
.L1:
      sub  r1, r1, 3
  b        .L3
.L2:  sub  r2, r2, 7
```

'L3:

HW 3.9

```
if (a>=5)
  { a=a-4;}
else if (b>=-2)
  { b=1;}
else {
      c=2;}
}
```

```
cmp r0,5
blt  .L1

.L0:
    sub r0,r0,4
    b   .L4
.L1:

    cmp r1,-2
    blt   .L3

.L2:
    mov r1,1
       b    .L4
.L3:
    mov r2,~~2~~

.L4:
```

HW 3.10 Converting ARM to C.

```
cmp r0, r1
 ble .Lelse
  mov r3, r2
  b. Lend
.Lelse:
  add r3, r0, r1
 .Lend
```

```
int branch (int r0, int r1, int r2)

  { int r3=0;
    if (r0>r1)
      { r3 = r2;
      }
    else
      { r3=r0+r1; }
```

return r3;

HW 3.11

```
  cmp r0, r1
  bgt .Lelse
  add r4, r0, r1
  cmp r4, r2
  ble .Lelse
  mov r3, r2
  b .Lend

.Lelse:
    add r3, r0, r1

.Lend:
```

C code:

```
int branch(int r0, int r1, int r2)
{ int r4=0;
  int r3=0;
  if (r0≤r1)
    { r4=r0+r1;
      if (r4≤r2)
        {
          r3=r0+r1;
        }
      else
        { r3=r2;
        }
    }
  else
  {
    r3=r0+r1;
    }
  return r3;
```

HW 3.12

```
 lsr r4, r2, #17
 cmp r4, #1
  bne .Litelse
    and r3, r0, r1
   b .Lend

.Litelse
   cmp r0, #0
   bge .Lelse
   mov r3, r1
   b .Lend
```

```
.Lelse:
    add r3, r0, r1
    add r3, r3, r2
.Lend
```

C code:

```
int branch(int r0, int r1, int r2)
{  int r3=0;
   int r4 = (r2>>7);
   if(r4==1)
    {
      r3= r0 ~~&~~ r1;
    }
   else
    {
```

```
    if (r0 < 0)
    {
        r3 = r1;
    }
  else
        {
        r3 = r0+r1;
        r3 = r3+r2;
        }
  }
      return r3;
}
```

Hw 3.13 unoptimized loop tracing

```
mov r3, 0x8f

.L1:
  and r4, r3, 0x04
  cmp r4, 0x04
   beq .L2
   lsr r3, r3, 1
   b .L1
.L2
  mov r0, r3
```

r0 final value?

lsr called how many times?

① r3 = 0x89; = 10001001

② r4 = r3 & 0x04

$$
\begin{array}{r}
= 10001001 \\
\& \ 00000100 \\
\hline
00000000
\end{array}
$$

= 0x00

③ cmp r4, 0x04 ≠

④ lsr r3, r3, 1

10001001 >> 1

$\rightarrow$ 01000100 = 0x44 (r3)

⑤ b .L4

⑥ r4 = r3 & 0x04

```
  01000100
& 00000100
----------
  00000100
```

r4 $\rightarrow$ 0x04

⑦ r4 = 0x04

⑧ r0 = r3 = 0x44

[HW 3.1]

x = array[4]

```
mov r2, 4
mul r2, r2, 4
ldr r6, [r0, r2)
```