if and loops

- Need to do one of two things
  - if: do next instruction or skip to another (else)
  - loop: continue in loop or exit loop
Decision instructions in ARMv8

- Simple instructions:
  - `cbz X9, label`
    - Conditional branch on zero
    - If value in X9 equals zero, go to instruction at label
    - otherwise, execute next instruction in order
  - `cbnz X9, label`
    - Conditional branch if not zero
    - If value in X9 does not equal zero, go to instruction at label
    - otherwise, execute next instruction in order
if (a == b)    // a in X19, b in X20
    {c = 1}    // c in X21
// next instruction

sub X9, X19, X20
cbnz X9, endif
mov X21, #1
endif:    // next instruction
if - else

if (a == b) // a in X19, b in X20
  {c = 1} // c in X21
else
  {c = 0}

// next instruction
if – else in assembly

sub X9, X19, X20

cbnz X9, else

mov X21, 1

b endif

else: mov X21, 0

endif: # next instruction
What about other comparisons?

- a < b
- a <= b
- etc...
Conditional branches based on flags

- Flags set by instructions
  - set by arithmetic/logical instructions
  - indicate facts about previous result
  - four bits
    - negative(N) result was negative
    - zero (Z) result was zero
    - overflow(V) result had overflow
    - result(C) result had a carry out of most significant bit, or borrow into most significant bit
Conditional branches based on flags

- **Most commonly used with subs**
  - subtract and set flags
  - subs XZR, X9, X10
    - X9 – X10, set flags, can’t write to XZR
  - Condition branches on relation between X9, X10
    - b.lt branch if less than; b.le branch if less than or equal
    - b.gt, (b.ge) branch if greater than (or equal)
  - Unsigned conditions
    - b.lo, (b.ls) branch if lower (or same)
    - b.hi, (b.hs) branch if higher (or same)
Conditional branches based on flags

- Can test flags directly, when they are set by one of a limited set of operations, (append s to instruction to indicate that flags will be set, e.g. adds)
  - b.m branch on minus \((N = 1)\)
  - b.pl branch on plus \((N = 0)\)
  - b.vs branch if overflow \((V = 1)\)
  - b.vc branch if overflow clear \((V = 0)\)
if again

if(a < b)    // a in X19, b in X20
    {c = 1}    // c in X21

// next instruction

subs XZR, X19, X20
b.ge endif
mov X21, 1
endif:    # next instruction
while loop

while (condition)
{
    loop body
}

Each time at the top of the loop, check the condition. If true, continue the loop. At the end of the loop, go back to check the condition again.
while loop example - c

while (count < 10)
{
    // do something
    count ++;
}

// next instruction
while loop example - assembly

// Assume count is in register X10, 
// and register X19 contains 10

loop:     subs XZR, X10, X19
           b.?? loopend
           # do something
           add X10, X10, 1
           b loop

loopend:  
           # next instruction
while loop example - assembly

// Assume count is in register X10,
// and register X19 contains 10

loop:     subs XZR, X10, X19
b.ge loopend
# do something
add X10, X10, 1
b loop

loopend:
# next instruction
for loop example - c

```c
for(cnt = 5, cnt >= 0, cnt--)
    {
        // do something
    }

// next instruction
```
for loop example - assembly

```
mov     X10, #5

loop:   subs   XZR, X10, #0
        b.lt    loopend
        # do something
        sub     X10, X10, #1
        b     loop

loopend: # next instruction
```