Examples of delayed branching. A "delayed" branch means that the effect of the branch doesn't take place until after the next instruction. This next instruction is called the "delay slot". The assembly code you see below was generated by a compiler that filled the delay slot with a useful instruction rather than just a "nop".

Branch instructions ending with ",a" are "annulled" branches -- this means that the next instruction is nullified if the branch falls thru.

```
for (i = 0; i < 10; ++i)
  a[i] = i;

$3 = i
$4 = array element address
$5 = sentinel for end of array

        move $3, $0
        la   $4, a
        addi $5, $4, 40
L16:
        st   $3, 0($4)
        addi $4, $4, 4
        blt  $4, $5, L16
        addi $3, $3, 1

        li $v0, 10
        syscall
```

```
for (i = 0; i < n; ++i)
  a[i] = i;

$3 = i
$4 = array element address
$5 = sentinel for end of array

        lw     $5, n
        ble    $5, $0, L16
        move   $3, $0
        la     $4, a
        st     $3, 0($4)
L18:
        addi   $4, $4, 4
        addi   $3, $3, 1
        blt,a  $3, $5, L18
        st     $3, 0($4)

        li $v0, 10
        syscall
```

```
for (i = 0; i < 100; ++i)
  for (j = 0; j < 100; ++j)
    a[i][j] = 0;

$1 = offset to beginning of a row of the array
$2 = constant 40000, sentinel value for the end of the array
$3 = pointer to individual element of the array where we store value 0
$4 = sentinel value for the end of the row
$5 = base address of a
$6 = value to put into the array (which happens to be 0)

        move   $1, $0          # initialize offset to 0
        li     $2, 40000       # offset of end of array
        la     $5, a           # load base address
        move   $6, $0          # initialize value to put in array
        add    $3, $1, $5      # start pointing to beginning of row 0
L17:
        addi   $4, $3, 400     # point to end of row (sentinel)
        st     $6, 0($3)       # store value in array
L20:
        addi   $3, $3, 4       # increment pointer
        blt,a  $3, $4, L20     # continue inner loop if not at end of row
        st     $6, 0($3)       # store value in array

        addi   $1, $1, 400     # increment offset of where rows begin
        blt,a  $1, $2, L17     # continue outer loop if offset < 40000
        add    $3, $1, $5      # total address = offset + base

        li $v0, 10             # end of program
        syscall
```