ECE 252
Intro to Computer Engineering

Week 09 Discussion

Attendance via TopHat

Course code: **265393**

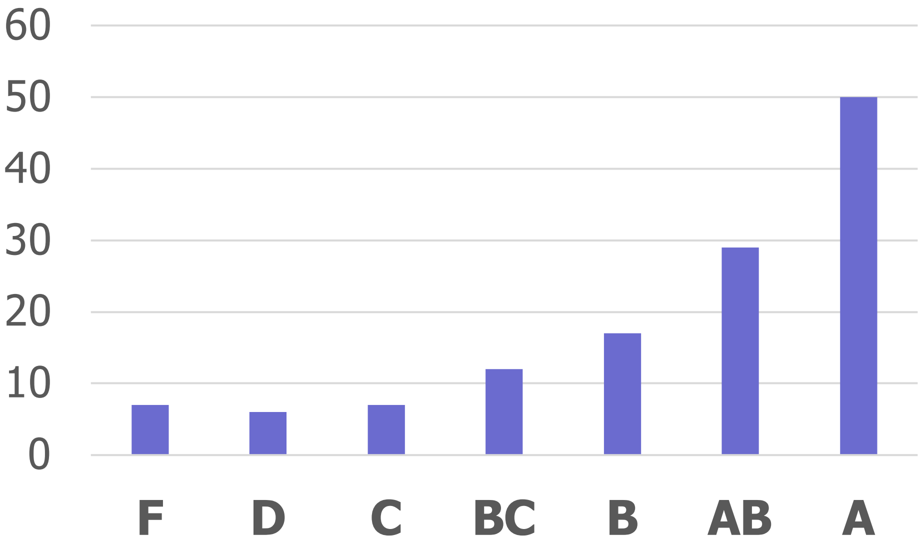
Attendance code:

5351

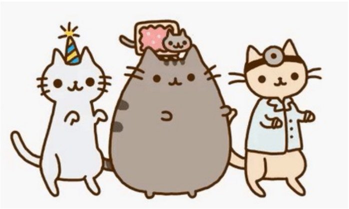
Assessment A3 – Fall 2024

- 50 / 128 students scored an A
- Average score: 87.22

A	93
AB	88
B	83
BC	78
C	70
D	60



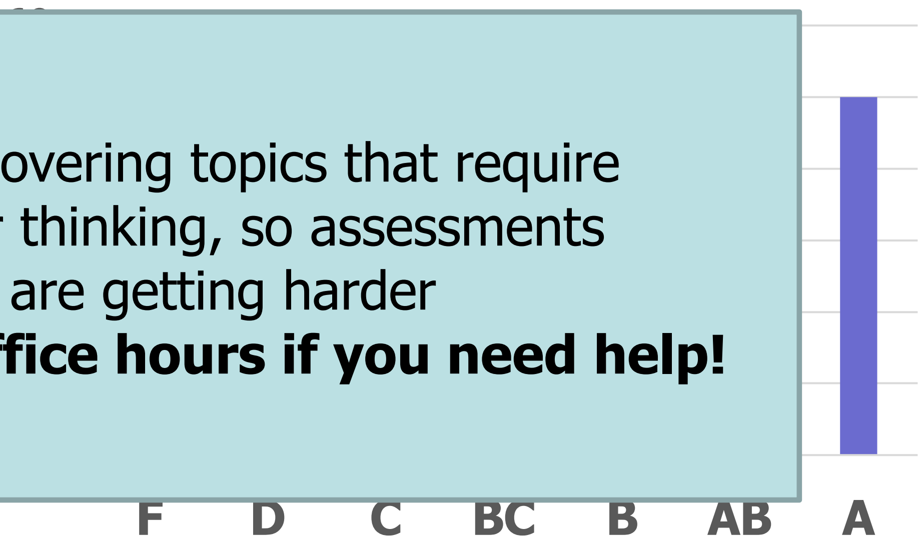
Course code: **265393**
Attendance code: **5351**



Assessment A3 – Fall 2024

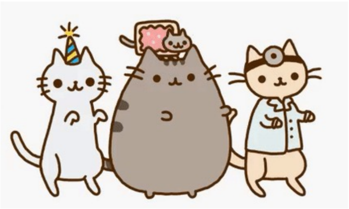
- 50 / 128 students scored an A
- Average score: 87.22

A	93
AB	[illegible]
B	[illegible]
BC	[illegible]
C	[illegible]
D	60



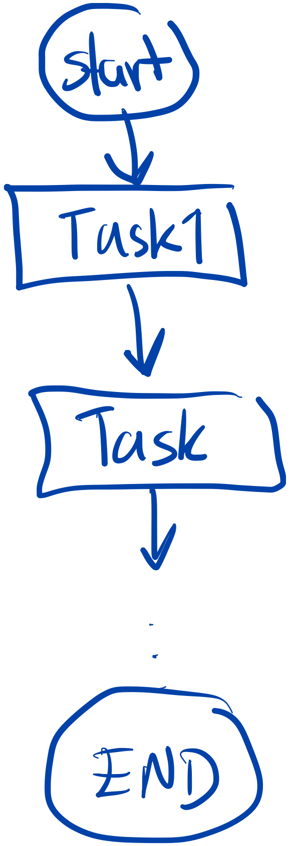
We're covering topics that require deeper thinking, so assessments are getting harder
come to office hours if you need help!

Course code: **265393**
Attendance code: **5351**

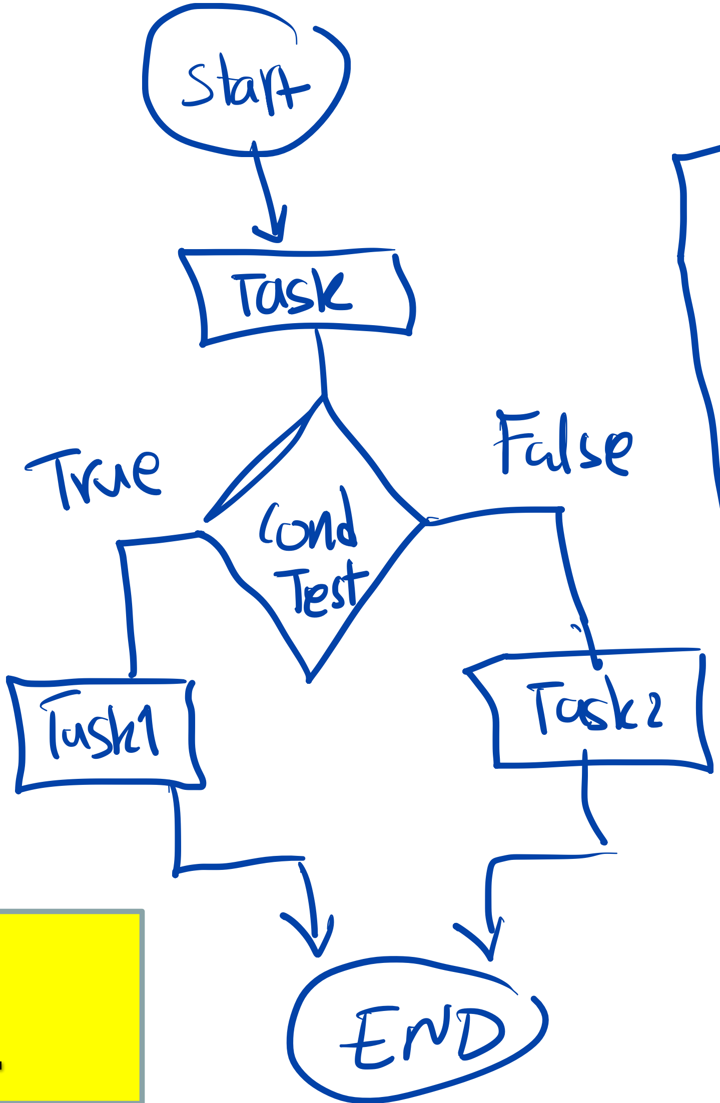


Structured Programming

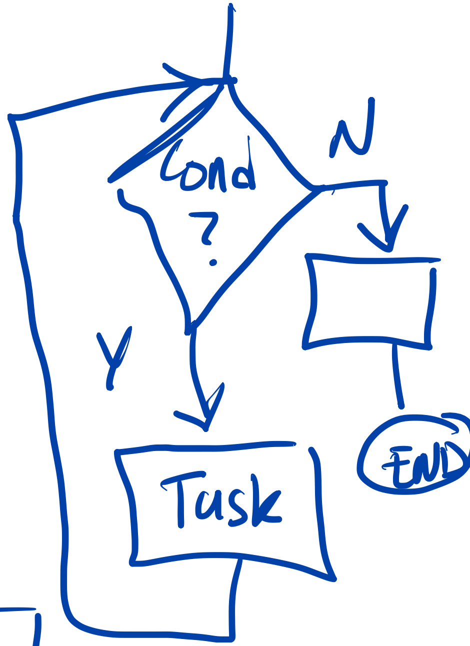
Sequential



Conditional



Iterative



Course code: **265393**
Attendance code: **5351**

Flowcharts and Programming

- Flowcharts are a graphical, high-level way to describe what your program will do
  - Determine the overall program flow
  - Identify all the major actions/tasks
  - Identify all of required decisions
- Once you have a flowchart, mapping each block into assembly language is relatively easy
  - Assembly language is very low-level – it is NOT a good vehicle for organizing a program
- Don’t write an assembly language program unless you can draw a flowchart!

Course code: **265393**
Attendance code: **5351**

Control Flow

Assembly

desc

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	n	z	p	PCoffset9								

BRx label ; Conditional branch to label
; x is condition ∈{n,z,p,zp,np,nz,nzp}

if ((n AND N) OR (z AND Z) OR (p AND P))
then PC ← PC^+ + SEXT(PCoffset9)

0 1 0

BRz →

x → any comb of NZP

BRnz 1 1 0

BRnzp => BR

Course code: **265393**
Attendance code: **5351**

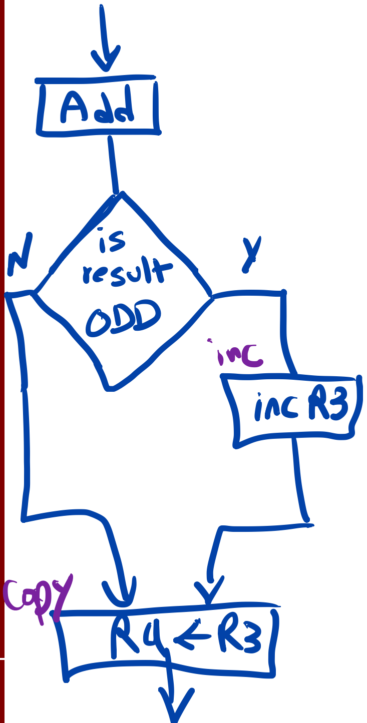
4 00100 EVEN 5 00101 ODD

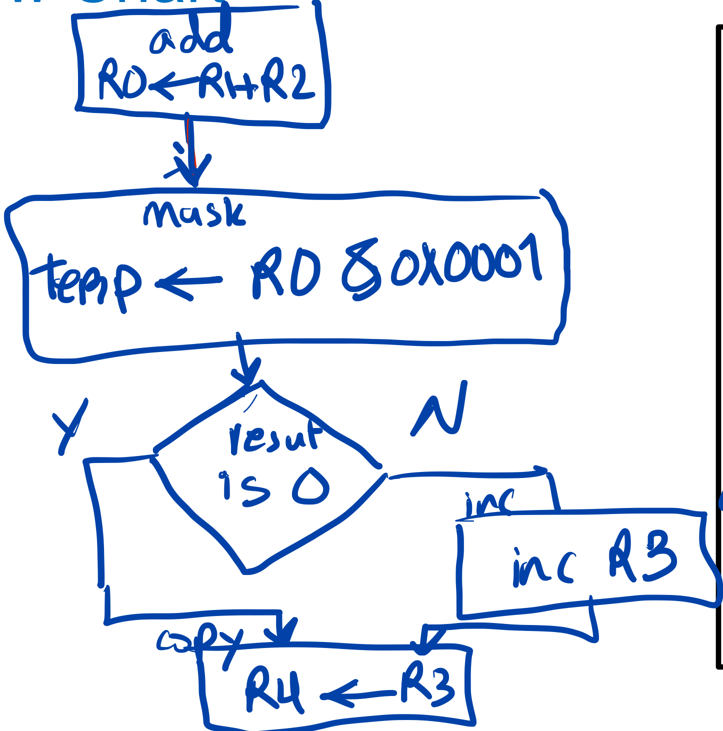
Example: Conditional Branches

- Perform operation R0 ← R1 + R2
- See if the result is ODD
  - If so, increment register R3 (i.e., R3 ← R3 + 1)
- Next, perform R4 ← R3

xxxx xxxx xxxx xxxx
& 0000 0000 0000 0001
0000 0000 0000 000X

Flow Chart





Instructions

```
        ADD R0, R1, R2
        AND R0, R0, #1
        BRz Copy
INC     ADD R3, R3, #1

COPY    ADD R4, R3, #0
```

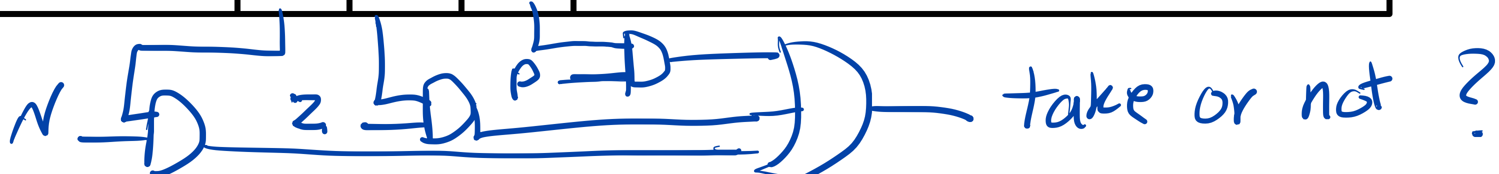
Encode The Branch Instruction

	Address	Instruction
	0x3132	. . . .
	0x3133	ADD R0, R1, R2
	0x3134	AND R0, R0, 1
PC →	0x3135	BRz COPY
PC+ →	0x3136	ADD R3, R3, 1
COPY	0x3137	AND R4, R3, R3
	0x3138	. . . .

PC = 3135

$PC^+ = 3136$

opcode	n	z	p	PCoffset9
0000	0	1	0	0 0000 0001



Encode The Branch Instruction, more practice

	Address	Instruction
	0x3132	. . . .
LOOP	0x3133	AND R0, R2, R1
	0x3134	ADD R0, R0, -5
	0x3135	ADD R3, R0, R2
PC →	0x3136	BRz LOOP
PC+ →	0x3137	AND R4, R3, R3
	0x3138	. . . .

offset = ?

= -4

X X X X

-2^3 2^2 2^1 2^0

1 1 0 0

opcode	n	z	p	PCoffset9
0000	0	1	0	1 1111 1100

What Happens?

R1: 8888
R2 2222
AAAA

→ 1010 1010 1010 1010
0000 0000 0000 0001
0000 0000 0000 0000

	Address	Instruction	New NZP	New PC
	0x3132	. . . .		
	0x3133	ADD R0, R1, R2	N 100	3134
	0x3134	AND R0, R0, 1	Z 010	3135
	0x3135	BRz COPY	〃 〃	3137
	0x3136	ADD R3, R3, 1	not executed	
COPY ✓	0x3137	AND R4, R3, R3	P 001	3138
	0x3138	. . . .		

Register Values After Executing Instruction At…

	0x3132	0x3133	0x3134	0x3136	0x3137
R0	0x0000	0x AAAA	0X0000 →	→	→
R1	**0x8888** →	→ The same →	→	→	→
R2	0x2222 →	→	→	→	→
R3	0x3333 →	→	→	→	→
R4	0xFFFF →	→	→	→	0x3333

What Happens?

```
 9999
 2222
-----
 BBBB
```

```
 1011
 0001
-----
 0001
```

	Address	Instruction	New NZP	New PC
	0x3132	. . . .		
	0x3133	ADD R0, R1, R2	N	3134
	0x3134	AND R0, R0, 1	P	3135
	0x3135	BRz COPY Not taken	//	3136
	0x3136	ADD R3, R3, 1	P	3137
COPY	0x3137	AND R4, R3, R3	P	
	0x3138	. . . .		

Register Values After Executing Instruction At…

	0x3132	0x3133	0x3134	0x3136	0x3137
R0	0x0000	0xBBBB	0x0001	→	→
R1	**0x9999**	→	→	→	→
R2	0x2222	→	→	→	→
R3	0x3333	→	→	0x3334	→
R4	0xFFFF	→	→	→	3334

Example 0 – Problem Statement

- Count the number of bits in a word that are 1.

Method 1: mask

Similar to checking if odd, but also check each other bit!

Limited by the immediate… (-16 ≤ imm5 ≤ 15)

Put mask in register; shift it left 1 position each iteration

Example: $\boxed{10110}$ $\boxed{3}$

Count=0

10110 and 0000$\boxed{1}$=00000

Count=0

10110 and 000$\boxed{1}$0=00010

≠2

Count=1

10110 and 00100=00100

Count=2

10110 and 01000=00000

Count=2

10110 and 10000=10000

Count=3

00000

Example 0 – Problem Statement

- Count the number of bits in a word that are 1.

Method 2: check sign bit

Example: 10110

BRx instruction lets us easily check the msb of a number
1: N 0: Z or P

How to check other bits?

Shift word left 1 position each iteration!

10110 → N
Count=1

Shift left

01100 → P
Count=1

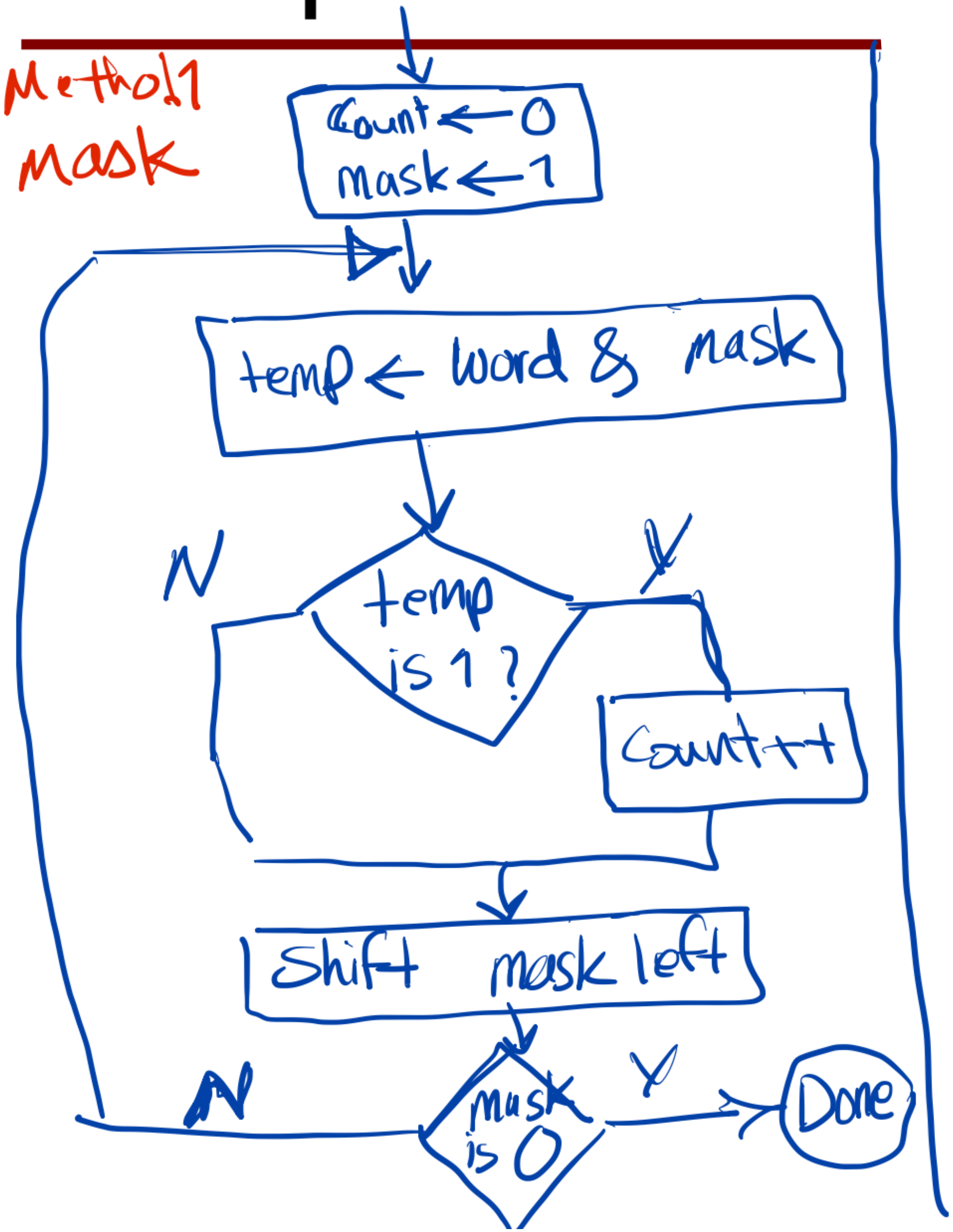
11000 → N
Count=2

10000 → N
Count=3

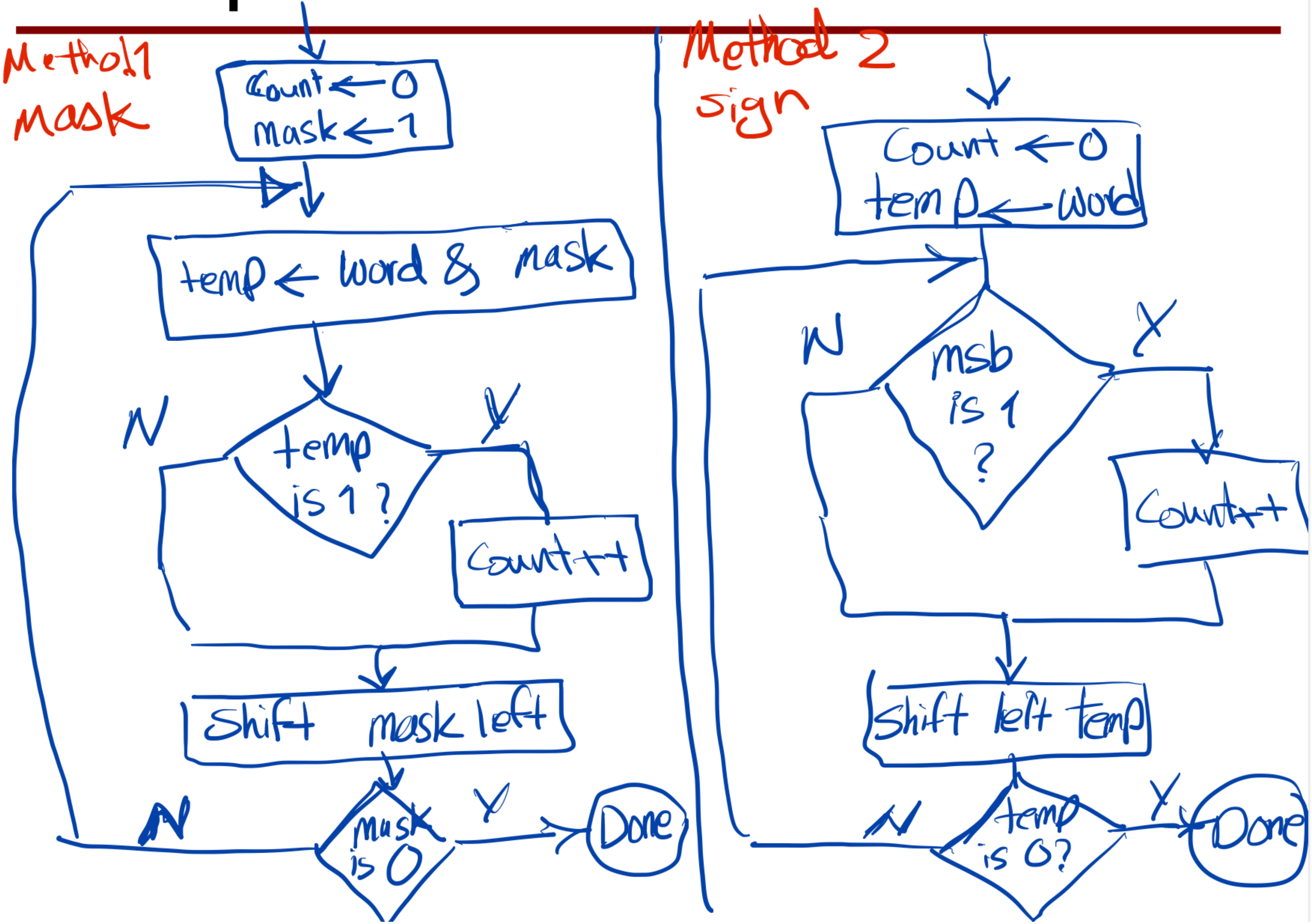
00000 → Stop

Example 1 – Flowchart

Method1
mask



Example 1 – Flowchart



Variable Mapping

- Assume the word to count the 1s in is in R0
- Assign purposes to other registers

Coding Tips

- When coding in assembly language, it's usually better to write the comments first
  - This helps organize your thought process
- Comments should tell someone why an instruction is there, not what it does
  - BAD comment
    - ADD R2, R2, #1 ; add 1 to R2
  - GOOD comment
    - ADD R2, R2, #1 ; increment count of 1s
  - Always assume the reader knows the ISA

Write the Program!

Wrapping Up

- Up Next:
  - LC-3 Assembly Language
  - LC-3 Data Allocation
  - LC-3 Assembler
- Remember your videos and reading
  - Including the video quiz!
- Reminder: HW9 (A and B) due tomorrow
- Questions?