ECE/CS 252
Intro to Computer Engineering

Week 10 Discussion

Attendance via TopHat

Course code: **265393**

Attendance code:

4585

Reminders

- Up Next:
  - Subroutines
- Remember your videos and reading
  - Including the video quiz!
  - HW10, Tuesday 10 PM
- **A4 on Friday Nov 15**

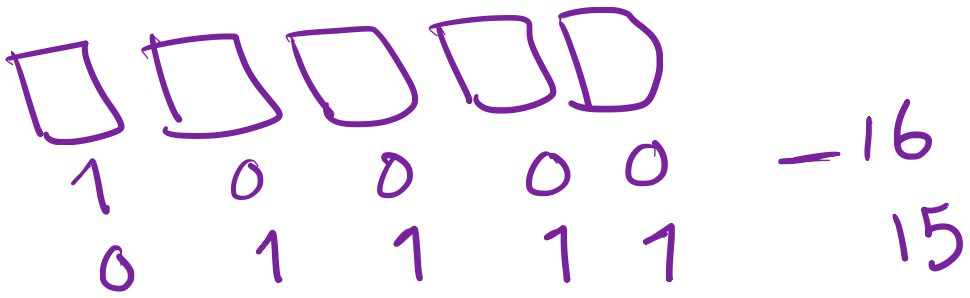
Course code: **265393**
Attendance code: **4585**

Exam Review

- Set a register to a value: R5← -3

```
AND  R5,R5,#0      ; clear R5
ADD  R5,R5,-3
```

[-16,15]



- Set a register to a value: R5← -17

```
LD   R5, value17
```

at the end of code

```
value17  .Fill  #-17
```

Course code: **265393**
Attendance code: **4585**

Exam Review

- Copy register value to another register: R1← R2

① ADD R1, R2, #0

② ADD R1, R2, -1

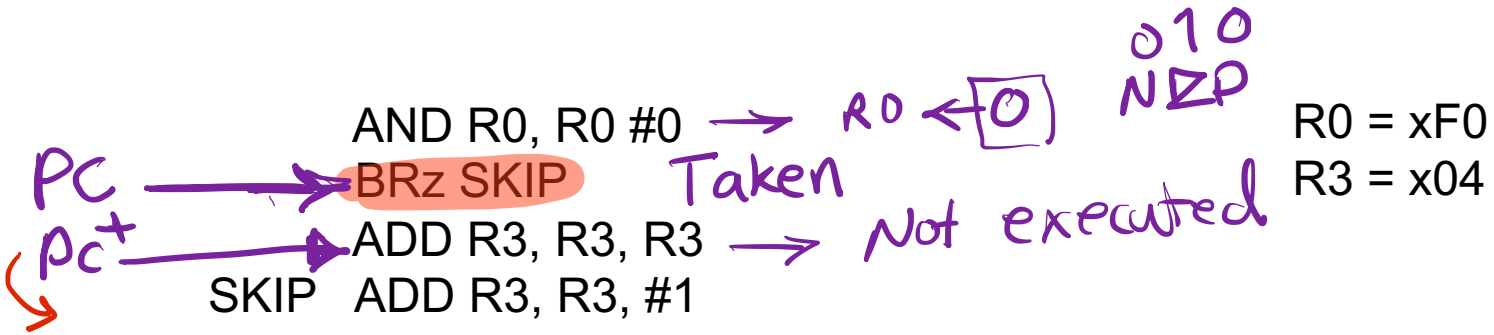
-1 ↓ all one

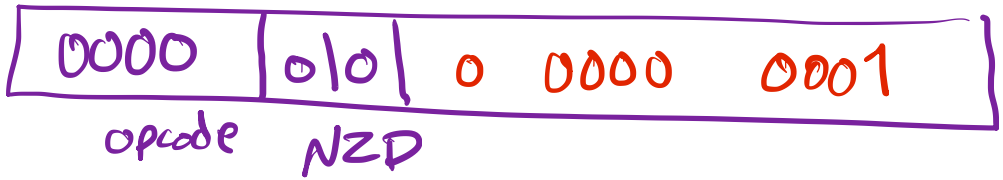
111 . . . 1

Course code: **265393**
Attendance code: **4585**

Exam Review

- Branch is taken or not?





effective addr = PC+ + SEXT(offset)

⇓

SKIP =

Course code: **265393**
Attendance code: **4585**

Exam Review

- Branch is taken or not?

```
        AND R0, R0, R2
        BRz SKIP
        ADD R3, R3, R3
SKIP    ADD R3, R3, #1
```

NOT Taken
executed

R0 = xF0
R2 = x10
R3 = x04

F0 → 1111 0000
10 → & 0001 0000
0001 0000

NZP → 001

Course code: **265393**
Attendance code: **4585**

Exam Review

- Branch is taken or not?

```
        AND R0, R0, R2
        BRzp SKIP
        ADD R3, R3, R3
SKIP    ADD R3, R3, #1
```

Taken

x not executed

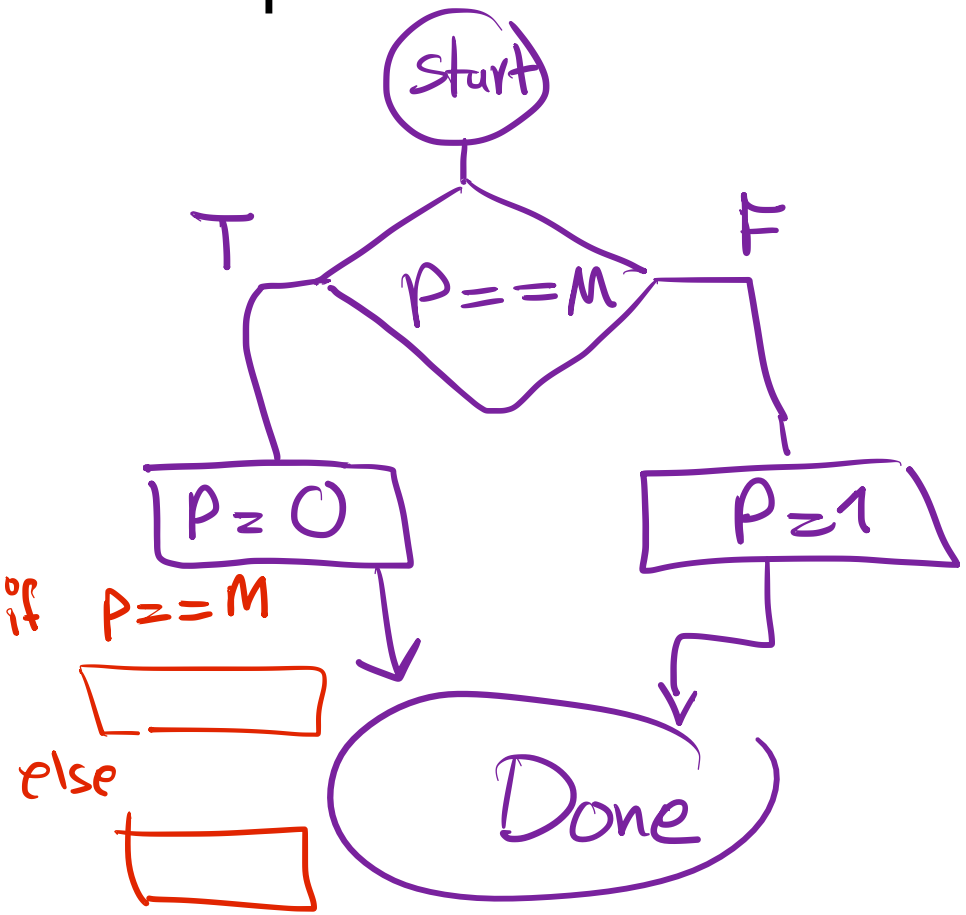
✓

R0 = xF0
R2 = x10
R3 = x04

Course code: **265393**
Attendance code: **4585**

Exam Review

- Complete Branch condition



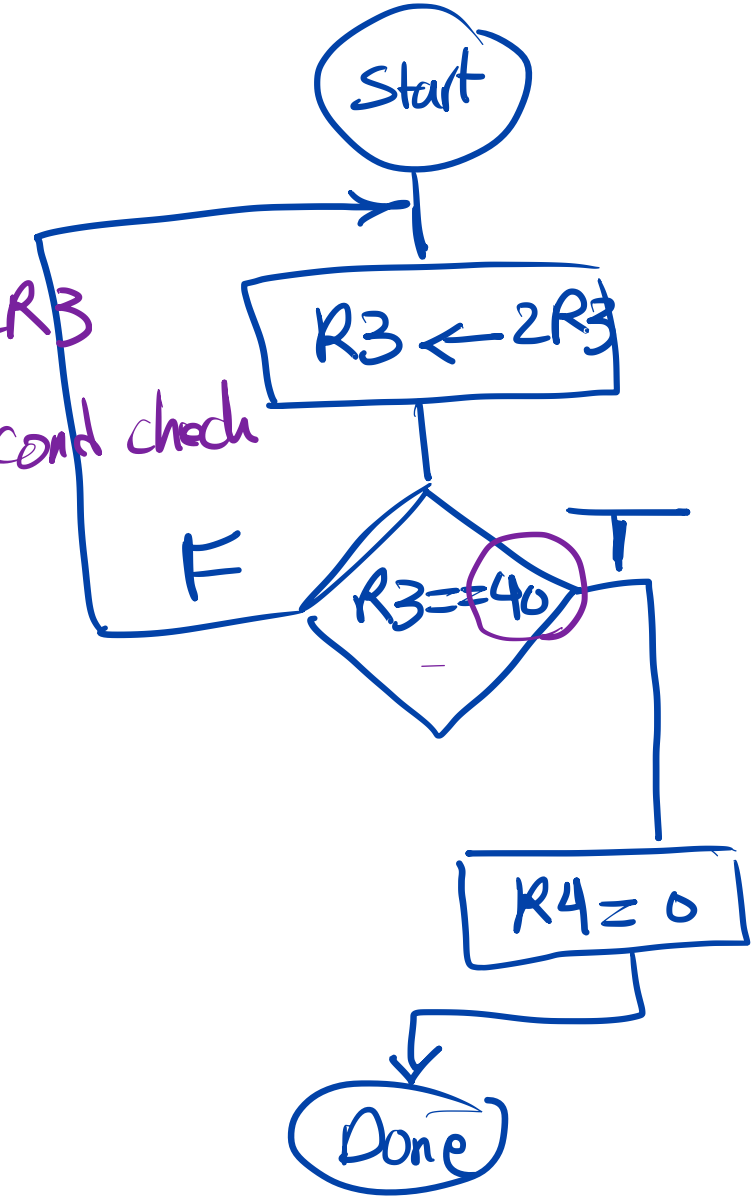

```
                .orig x0200
START
        LD R0, P
        LD R1, M
        NOT R1, R1
        ADD, R1, R1, #1
        ADD R2, R0, R1
        BR[ ]   PgetsOne
PgetsZero
        AND R4, R4, #0
        ST R4, P
        [ BR  Done]
PgetsOne
        AND R4, R4, #0
        AND R4, R4, #1
        ST R4, P
        [          ]
        [ Done     ]
P .FILL x10
M .FILL x20
```

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

Exam Review

- write code

```
Start
    LD    R1, ValueNeg40
Loop ADD  R3,R3,R3   ; 2R3
     ADD  R2, R1, R3 ; cond check
     BRNP loop
     AND  R4,R4,#0
Spin BR   Spin
ValueNeg40 .Fill  -40
```



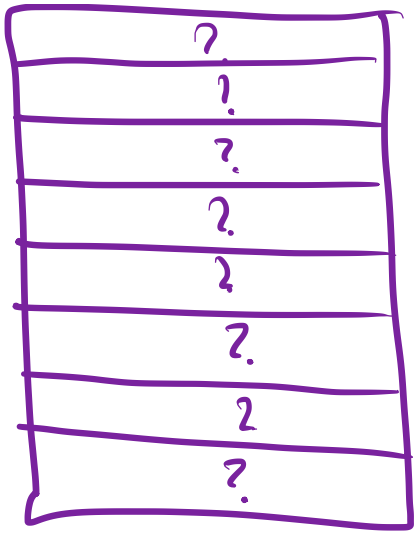
.FILL, .BLKW, .STRINGZ

- .FILL #8

1 line

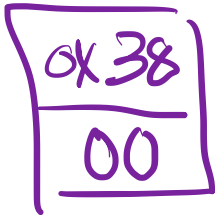
- .BLKW #8

8 lines



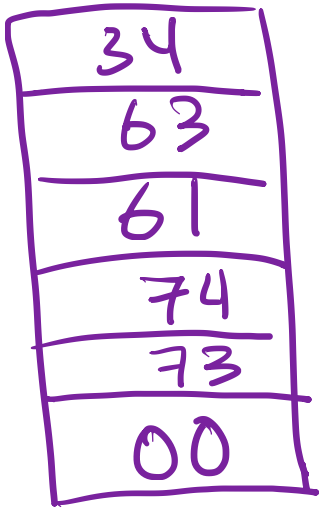
- .STRINGZ "8"

2 lines



.STRINGZ

- VAR .STRINGZ "4cats"

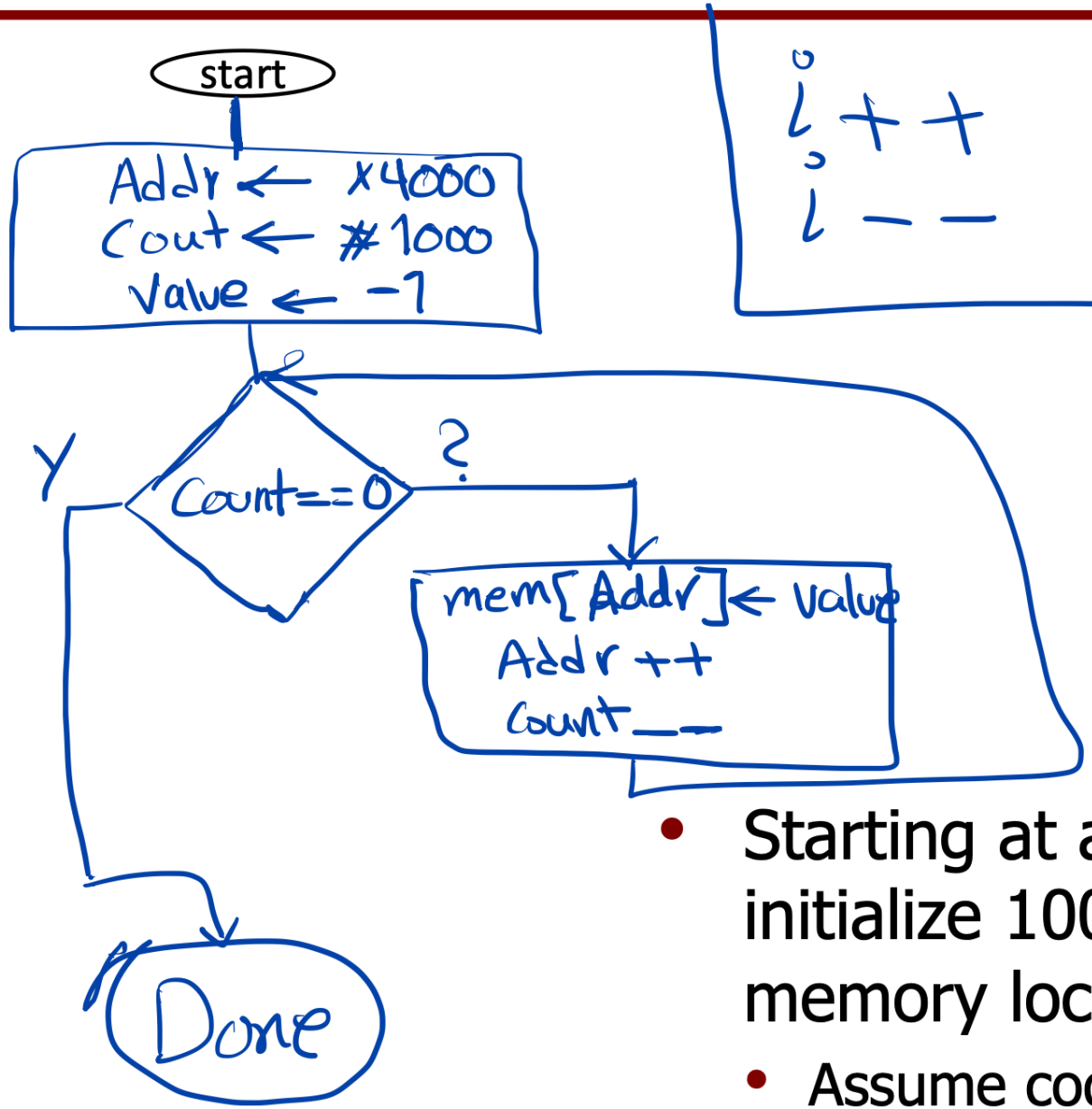


HEX	CHAR	*description*	HEX	CHAR	HEX	CHAR	HEX	CHAR
00	**NUL**	*null*	20	space	40	@	60	`
01	**SOH**	*start of header*	21	!	41	A	61	a
02	**STX**	*start of text*	22	"	42	B	62	b
03	**ETX**	*end of text*	23	#	43	C	63	c
04	**EOT**	*end of trans*	24	$	44	D	64	d
05	**ENQ**	*enquiry*	25	%	45	E	65	e
06	**ACK**	*acknowledge*	26	&	46	F	66	f
07	**BEL**	*bell*	27	'	47	G	67	g
08	**BS**	*backspace*	28	(	48	H	68	h
09	**HT**	*horizontal tab*	29	)	49	I	69	i
0A	**LF**	*line feed*	2A	*	4A	J	6A	j
0B	**VT**	*vertical tab*	2B	+	4B	K	6B	k
0C	**FF**	*form feed*	2C	,	4C	L	6C	l
0D	**CR**	*carriage return*	2D	-	4D	M	6D	m
0E	**SO**	*shift out*	2E	.	4E	N	6E	n
0F	**SI**	*shift in*	2F	/	4F	O	6F	o
10	**DLE**	*data link escape*	30	0	50	P	70	p
11	**DC1**	*device control 1*	31	1	51	Q	71	q
12	**DC2**	*device control 2*	32	2	52	R	72	r
13	**DC3**	*device control 4*	33	3	53	S	73	s
14	**DC4**	*device control 4*	34	4	54	T	74	t
15	**NAK**	*negative ack*	35	5	55	U	75	u
16	**SYN**	*synchronous idle*	36	6	56	V	76	v
17	**ETB**	*end of trans block*	37	7	57	W	77	w
18	**CAN**	*cancel*	38	8	58	X	78	x
19	**EM**	*end of medium*	39	9	59	Y	79	y
1A	**SUB**	*substitute*	3A	:	5A	Z	7A	z
1B	**ESC**	*escape*	3B	;	5B	[	7B	{
1C	**FS**	*file separator*	3C	<	5C	\	7C	|
1D	**GS**	*group separator*	3D	=	5D	]	7D	}
1E	**RS**	*record separator*	3E	>	5E	^	7E	~
1F	**US**	*unit separator*	3F	?	5F	_	7F	**DEL**

Programming Example

- Starting at address x4000, initialize 1000 consecutive memory locations to -1
  - Assume code starts at x0200

Programming Example



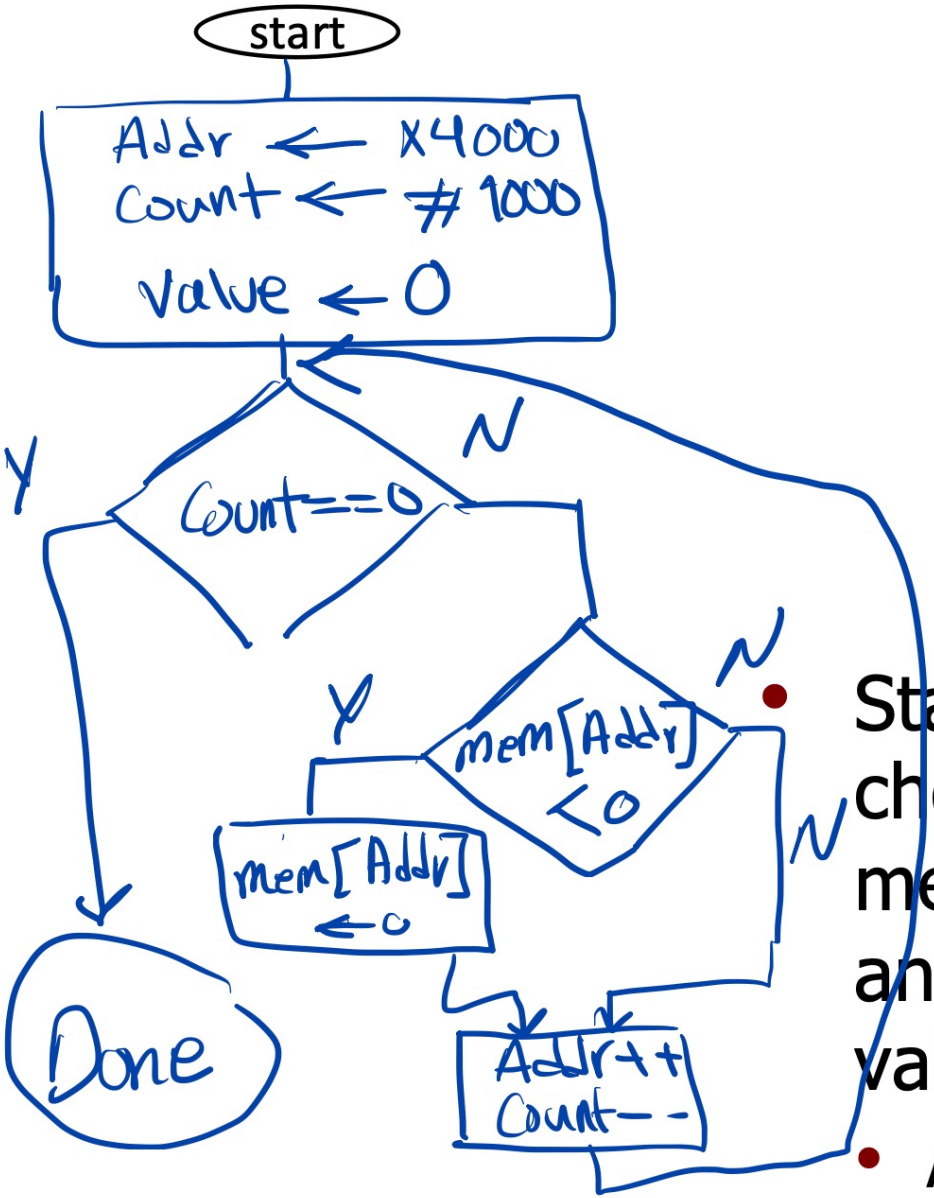
i++ i = i+1
i-- i = i-1

- Starting at address x4000, initialize 1000 consecutive memory locations to -1
  - Assume code starts at x0200

Programming Example

- Starting at address x4000, check 1000 consecutive memory locations and set any negative values to 0
  - Assume code starts at x0200

Programming Example



- Starting at address x4000, check 1000 consecutive memory locations and set any negative values to 0
  - Assume code starts at x0200