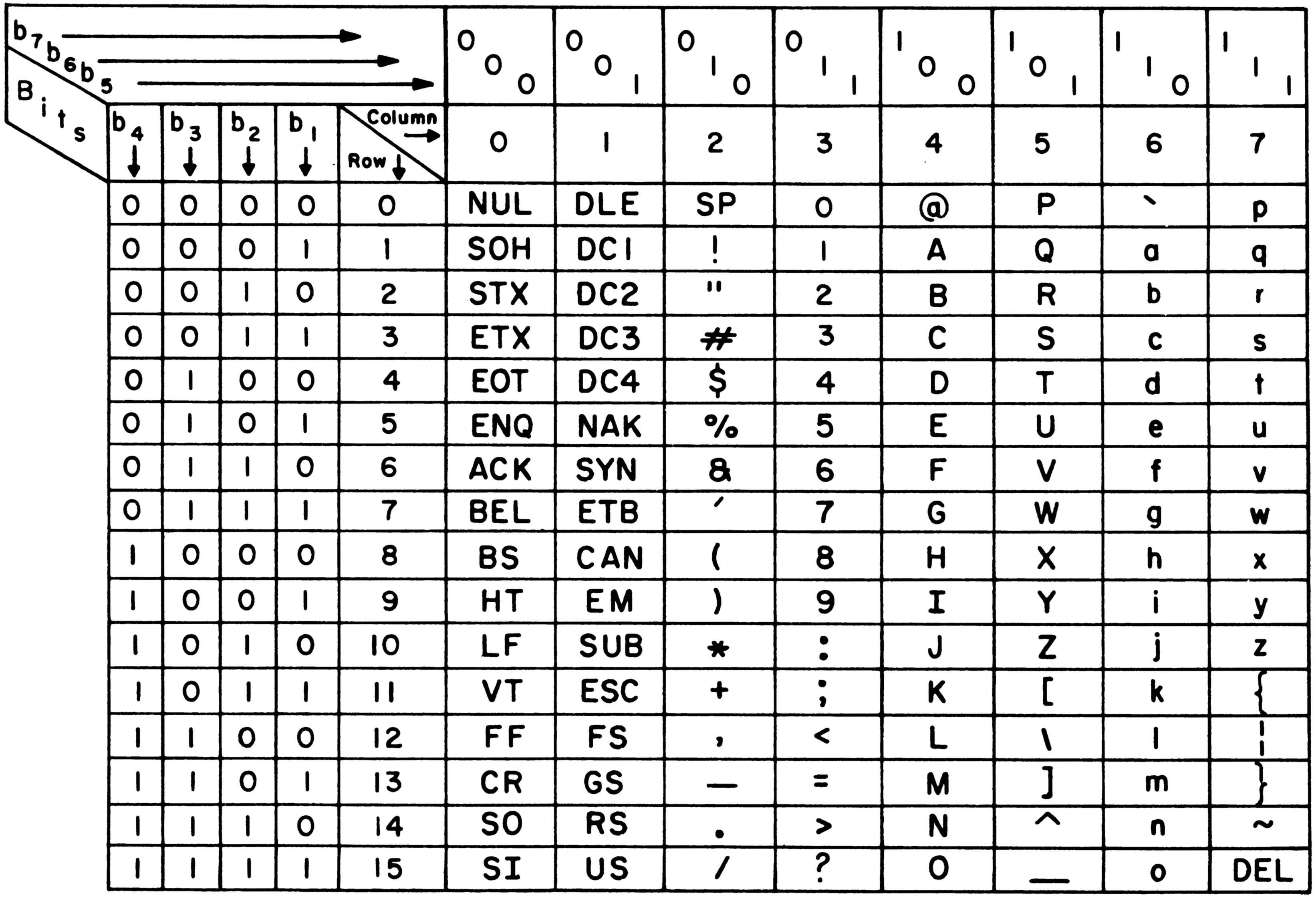


Bits $b_7 b_6 b_5$ →					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	Column → Row ↓	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	´	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	¦
1	1	0	1	13	CR	GS	—	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

CSE 13S Spring 2023

Computer Systems and C Programming

The American Standard Code for Information Interchange (ASCII)

Classroom information

Class time and location

M/W/F from 9:20 am – 10:25 am
Performing Arts M110 (Media Theater)

Final-exam day/time

Monday, June 12, 8:00 am – 11:00 am

Instructor

Dr. Kerry Veenstra

veenstra@ucsc.edu

Engineering 2 Building, Room 247A

(this is a shared office)

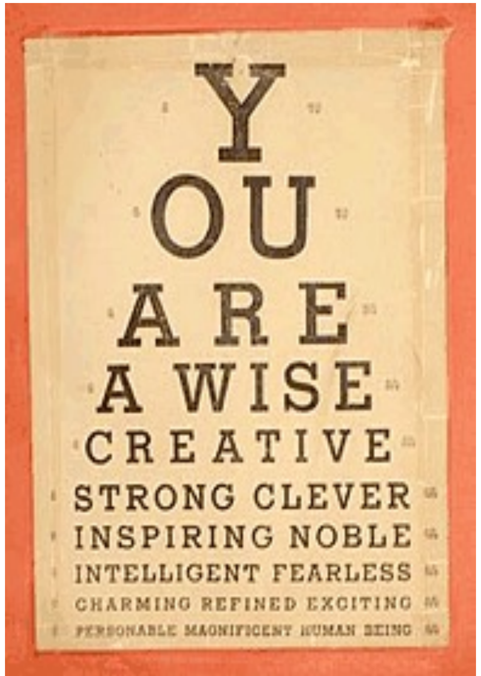
Office hours:

Tuesday 10:30 am – 12:30 pm

Thursday 2:00 pm – 4:00 pm

I’m totally supportive of DRC accommodations

- Bring me or email me your form ASAP
- Some folks need accommodations for the final only, some may need something for the quizzes: if so, we need to talk SOON!



So where does your grade come from?

- 20% Quizzes (top *n*–1 scores)
    - In class every Friday
    - I drop your lowest quiz score
- 50% Programming Assignments
- 30% Final Exam

I record the classes and post slides. **You** choose if you come to lecture—except for the quizzes.

NOTE: Assigned seats for the final exam

Canvas Web Site

- https://canvas.ucsc.edu/courses/62884
- Staff & Schedules (*still* under construction)
  - Office Hours
  - Discussion Section Times
  - Tutors & Times

Do-While Loop

- The last loop
  - The do-while loop is a "bottom-test" loop.
  - Always executes the body at least once.

```
do {
    // something
} while (condition);
```

printf() Format Specifications

- K&R Page 13
  - **`%d`** – print as decimal integer (corresponding parameter is int type)
  - **`%6d`** – print as decimal integer, at least 6 characters wide
  - **`%f`** – print as floating point (parameter is float or double type)
  - **`%6f`** – print as floating point, at least 6 characters wide
  - **`%.2f`** – print as floating point, 2 characters after decimal point
  - **`%6.2f`** – print as floating point, at least 6 wide and 2 after decimal point
- K&R 7.2, Pages 153–155
  - **`%x`** – print an int as hexadecimal number (compare **`%8x`** and **`%08x`**)
  - **`%c`** – print an int as the single ASCII character for that value
  - **`%%`** – print a single percent character (`%`)

`%c` uses ASCII (American Standard Code for Information Interchange)

- The base characters of modern computer character sets
- 95 "printable" characters (including a single space character)

  -
    ```
     !"#$%&'()*+,-./0123456789:;<=>?
    @ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
    `abcdefghijklmnopqrstuvwxyz{|}~
    ```
- Several "control" characters (invisible, but cause actions)
  - **`'\n'`** – New line (cursor down)
  - **`'\r'`** – Carriage return (cursor to leftmost column)
  - **`'\b'`** – Backspace (cursor left by one position)
  - **`'\\'`** – Not a control character, but a single backslash character (\)

Declaring variables

- int a; – contains an integer value
- float b; – contains a single-precision floating-point value
- double c; – contains a double-precision floating-point value
- char d; – contains a byte (8 bits, or 0 to 255, or –128 to 127)
- long e; – contains a "longer" integer value, a.k.a. long int

- **`type_name variable_name;`**
- **`type_name var1, var2, var3;`**

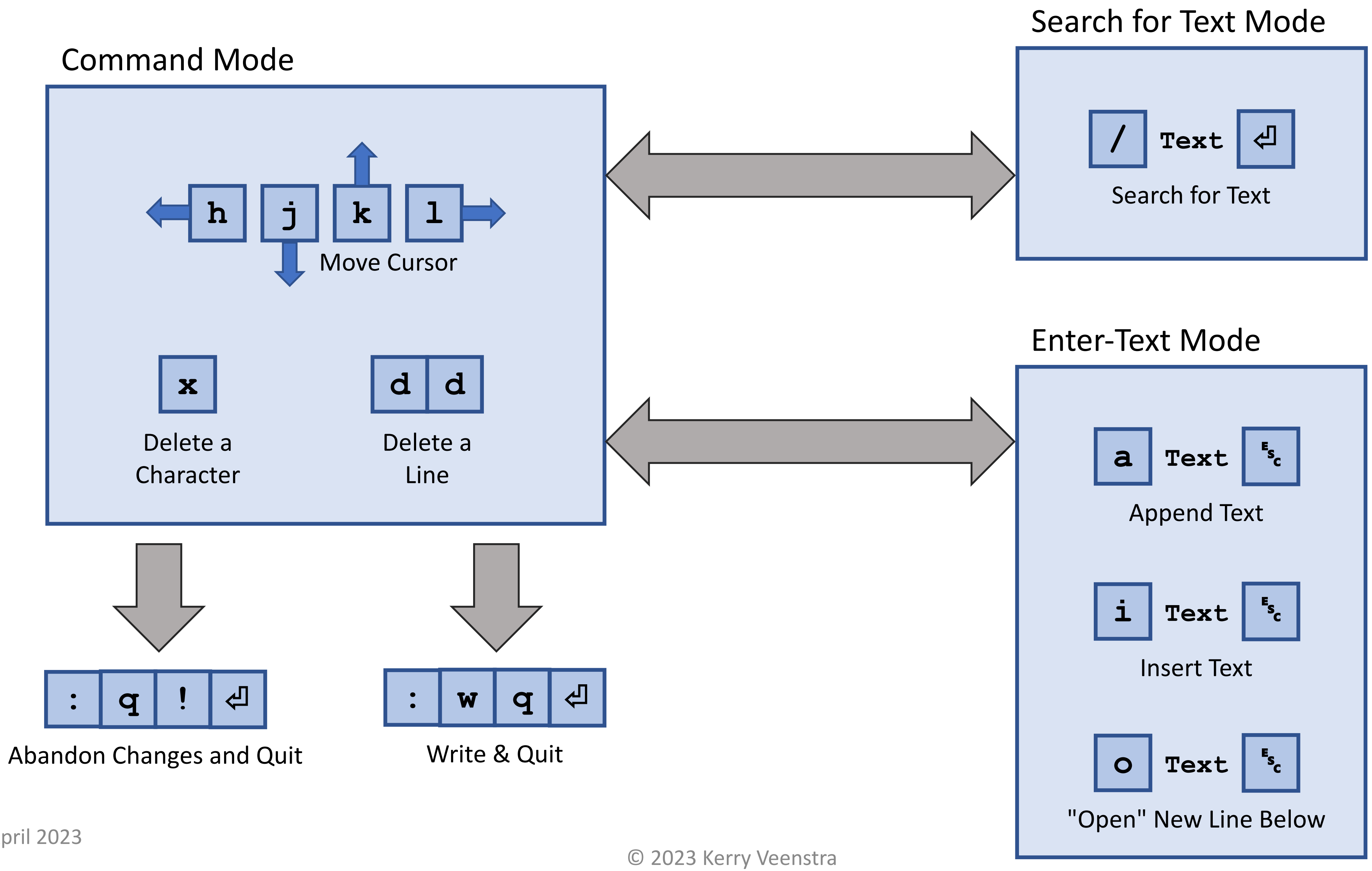
Declaring and Using Arrays

- **`int a[10];`** – Declare an array of 10 integers
- **`a[i]`** – access element **`i`** of **`a`** (0 ≤ **`i`** ≤ 9)
- **`a[11]`** – Oh, Bad! Don't do this!
- **`a[10]`** – Also Bad! Don't do this, either!
- **`a[-1]`** – No, no, no!

Assignment 0

- Posted!
- Due Friday at 11:59 pm
- Assumes that your laptop is running its Ubuntu VM
- Submit two files for "grading"

$ vi filename



Command Mode

h j k l

Move Cursor

x

Delete a Character

d d

Delete a Line

: q ! ⏎

Abandon Changes and Quit

: w q ⏎

Write & Quit

Search for Text Mode

/ Text ⏎

Search for Text

Enter-Text Mode

a Text Esc

Append Text

i Text Esc

Insert Text

o Text Esc

"Open" New Line Below

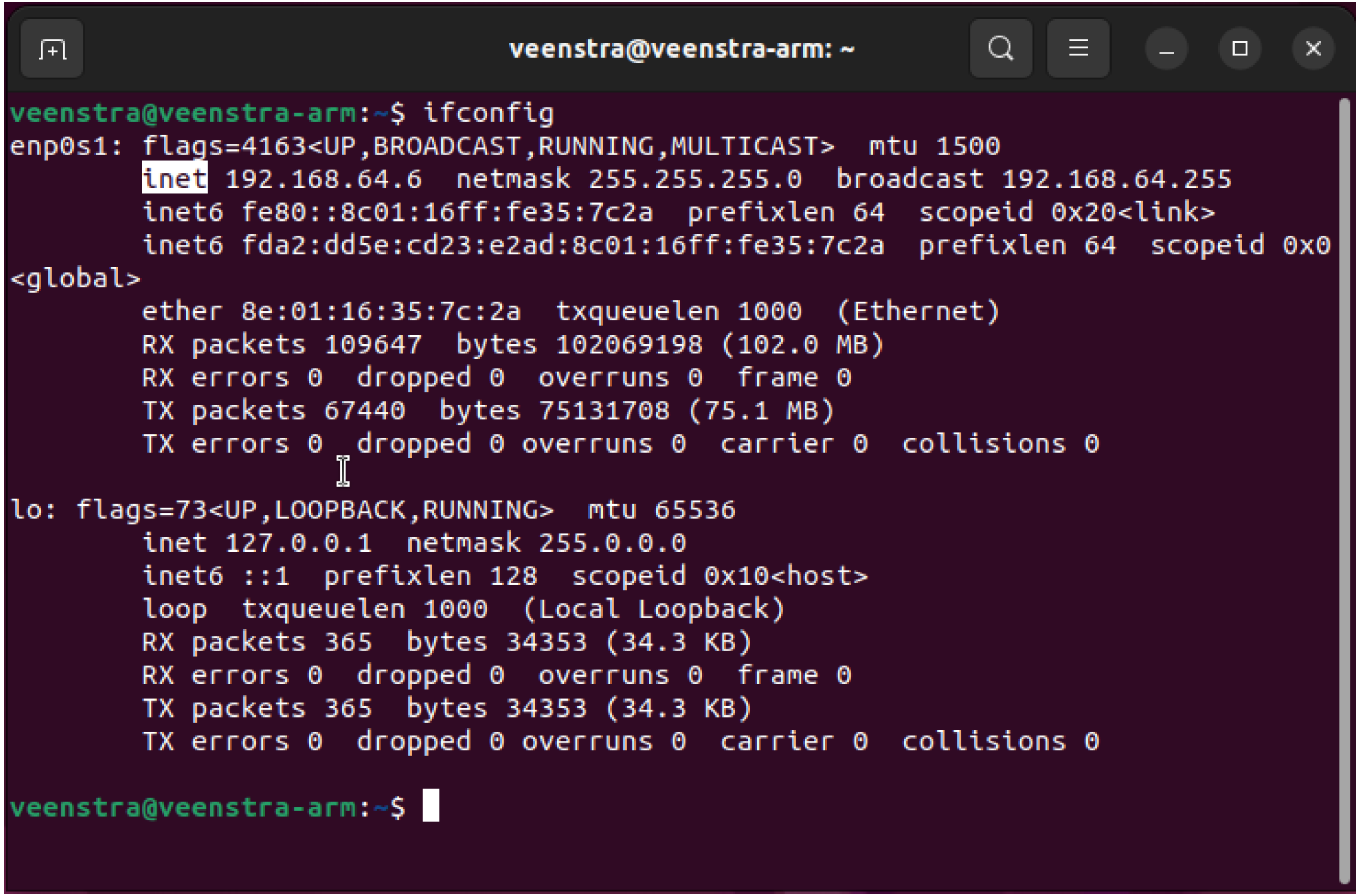
Painless Way to Learn a Programming Language

Write a series of tiny programs to verify your understanding of what you read.

Studying with a Blank Sheet of Paper

1. Using slides, notes, and textbook, make a list of topics that were covered.
2. For each topic, briefly summarize on a blank sheet of paper what you remember.
3. Peek at the slides/nodes/book to check that you remembered correctly and completely.
4. Did you forget something or not understand? That’s what you need to study!
5. Repeat until you’ve covered all topics.

How to Log Into Your VM From a Mac Termainal



```
veenstra@veenstra-arm:~$ ifconfig
enp0s1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.64.6  netmask 255.255.255.0  broadcast 192.168.64.255
        inet6 fe80::8c01:16ff:fe35:7c2a  prefixlen 64  scopeid 0x20<link>
        inet6 fda2:dd5e:cd23:e2ad:8c01:16ff:fe35:7c2a  prefixlen 64  scopeid 0x0
<global>
        ether 8e:01:16:35:7c:2a  txqueuelen 1000  (Ethernet)
        RX packets 109647  bytes 102069198 (102.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 67440  bytes 75131708 (75.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 365  bytes 34353 (34.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 365  bytes 34353 (34.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

veenstra@veenstra-arm:~$
```