This is  Section.

Week 2: Arrays

Gabe LeBlanc

Attendance Form: tinyurl.com/gabesection2

- What are the steps involved in **compilation**?
- When should we use **arrays**?
- What are **strings**, really?
- What's the point of **command-line arguments**?

- What are the steps involved in **compilation**?
- When should we use **arrays**?
- What are **strings**, really?
- What's the point of **command-line arguments**?
- What makes for good **design**?

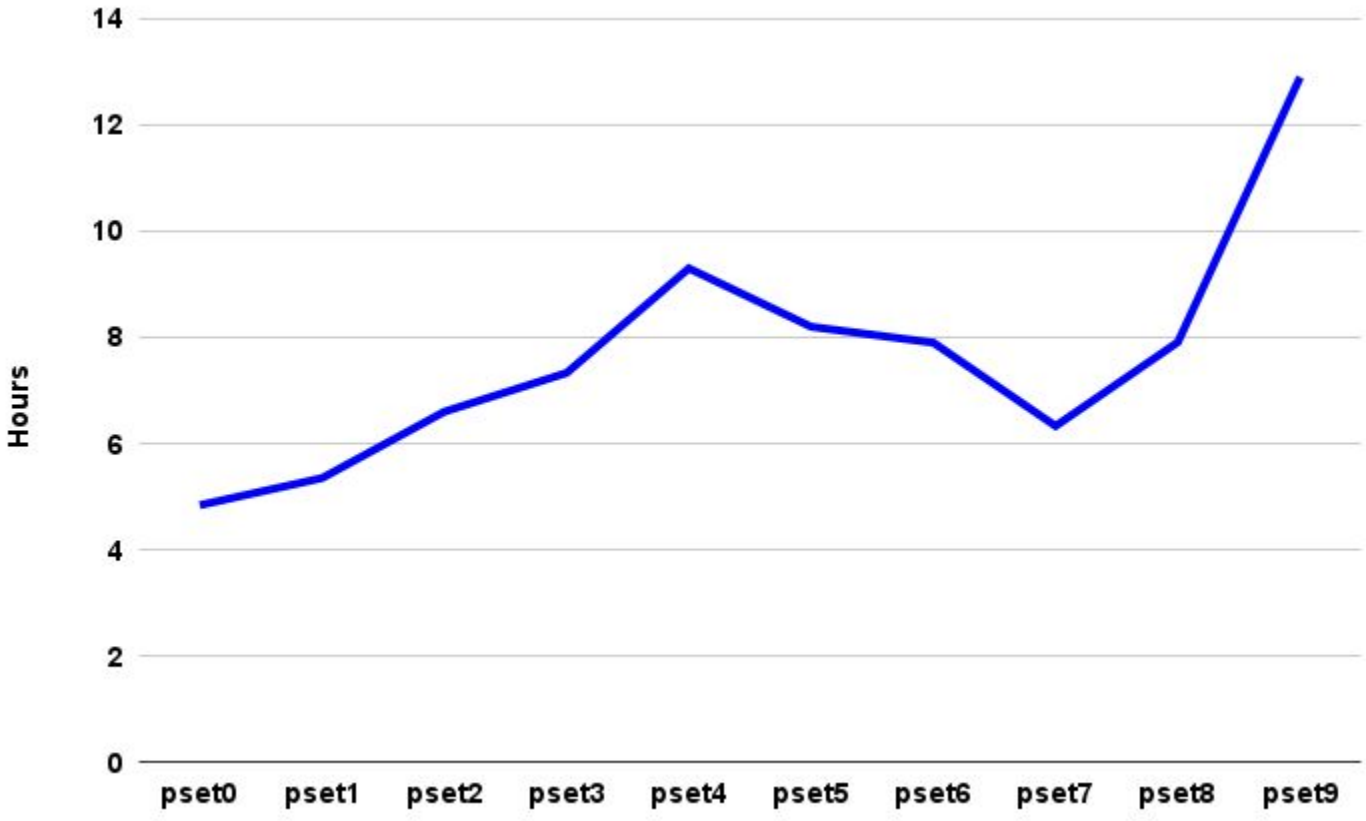
Questions from last week?

Arrays

PS1 Time Breakdown

How many hours total did you spend on PS1? *

Your answer

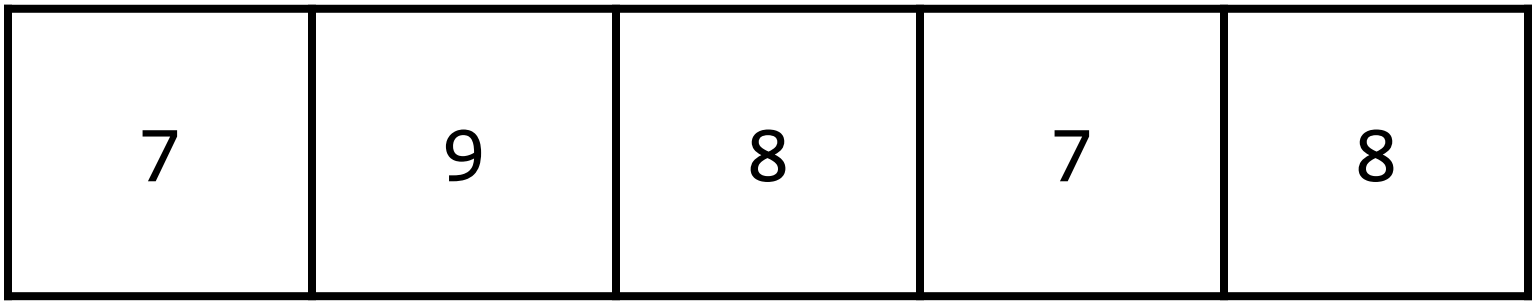

Hours
14
12
10
8
6
4
2
0
pset0
pset1
pset2
pset3
pset4
pset5
pset6
pset7
pset8
pset9

- int ps1_hours = 8
- int ps2_hours = 4
- int ps3_hours = 12
- …

[8, 4, 12, 5, 6, 10, 2, 11]

[“eggs”, “milk”, “flour”, “pineapple”]

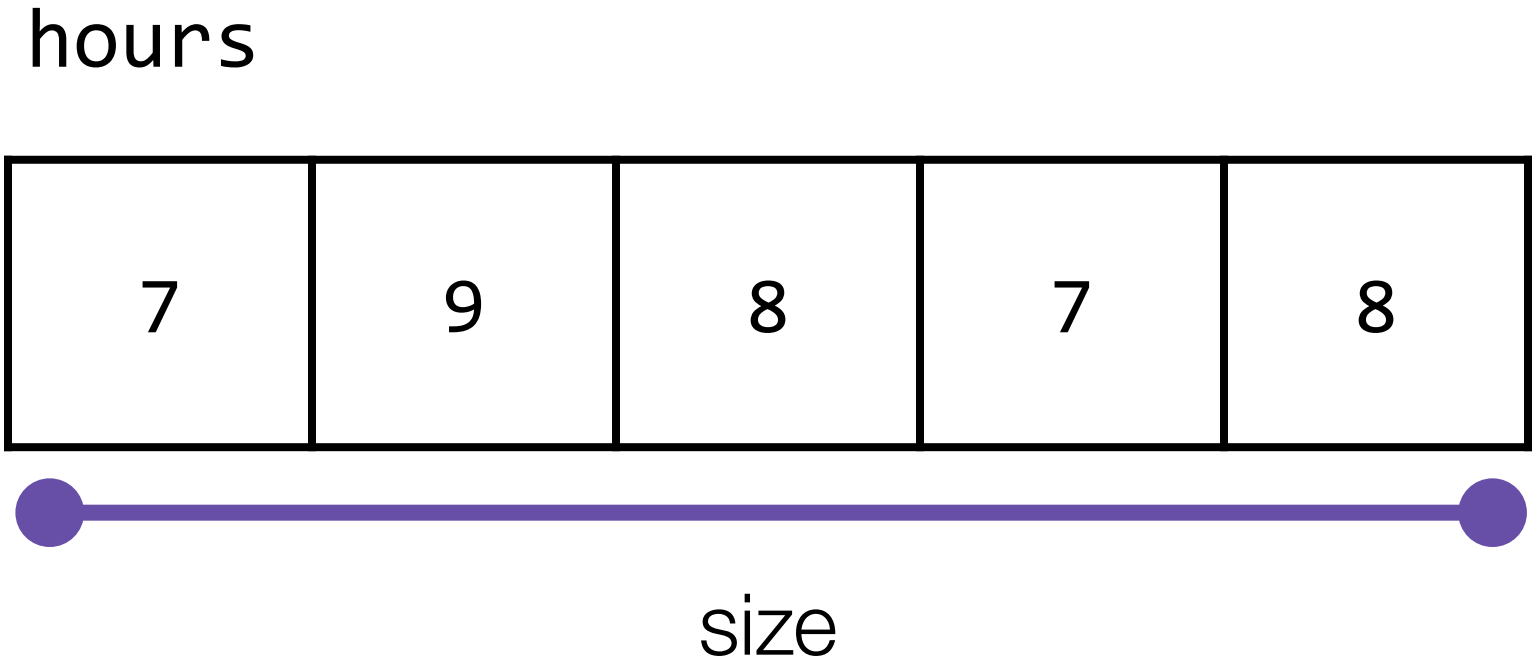
hours



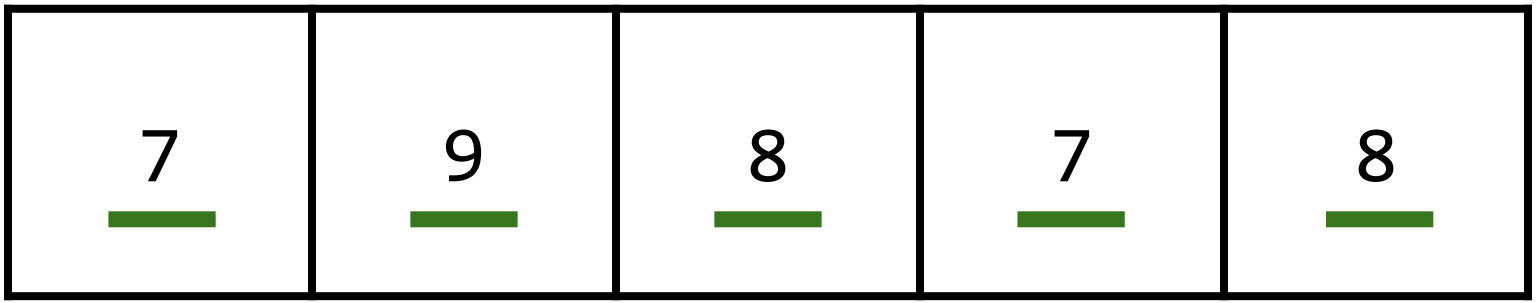
name

hours

7	9	8	7	8

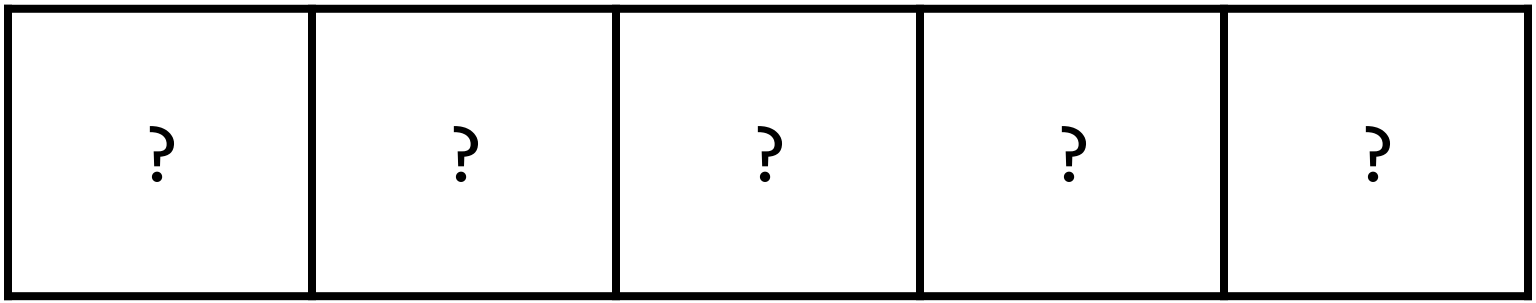

hours
7
9
8
7
8
size

`hours` type (int)



```
int hours[5];
```

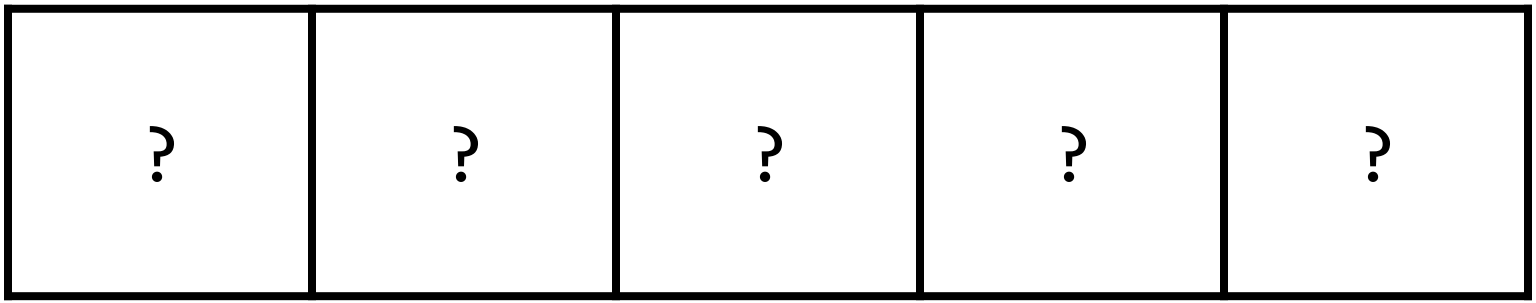
hours



name

```
int hours[5];
```

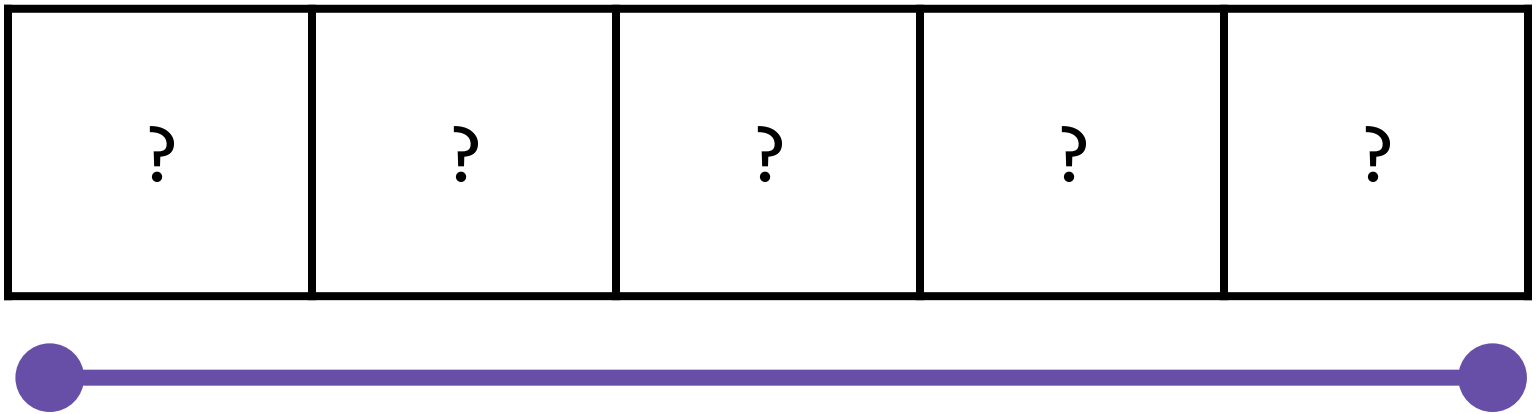
hours



size

```
int hours[5];
```

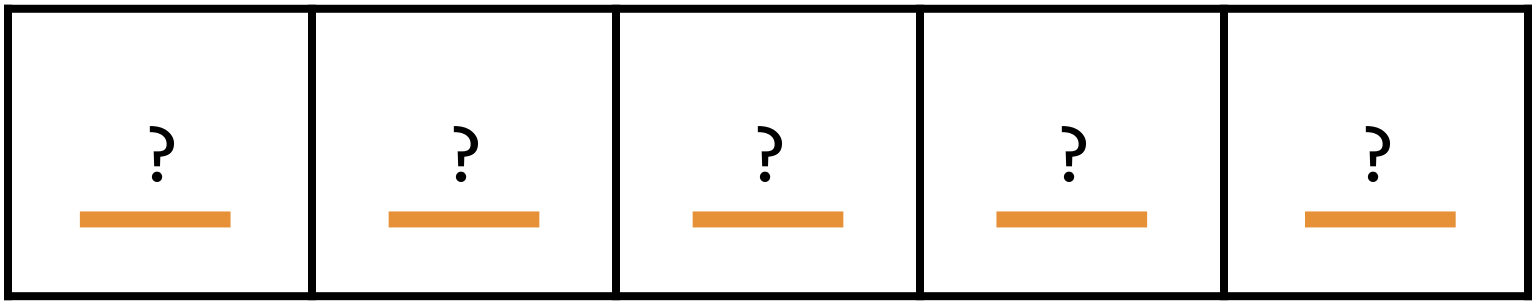
hours



type

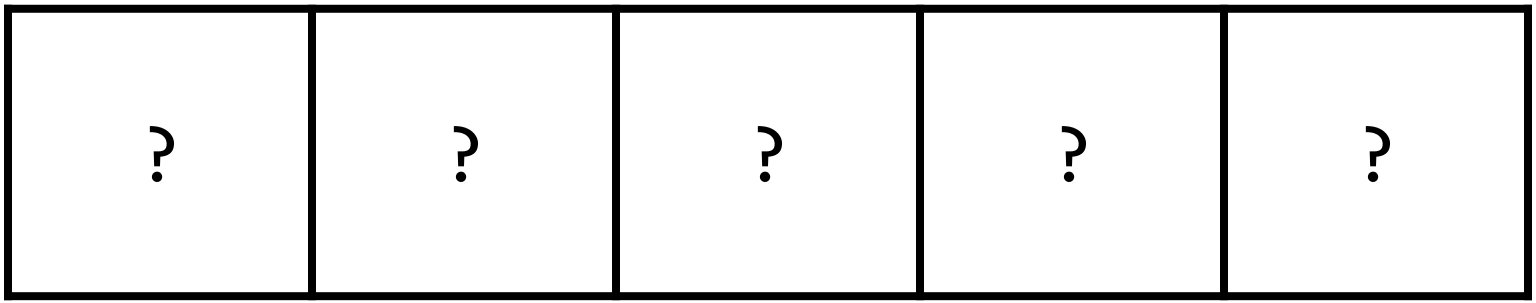
```
int hours[5];
```

hours



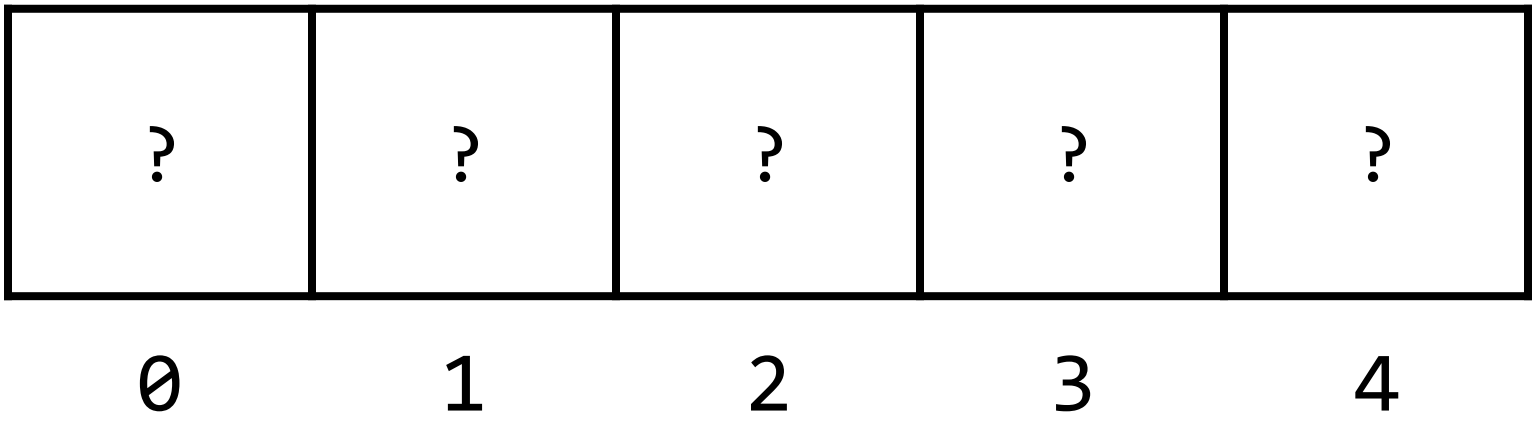
```
int hours[5];
```

hours



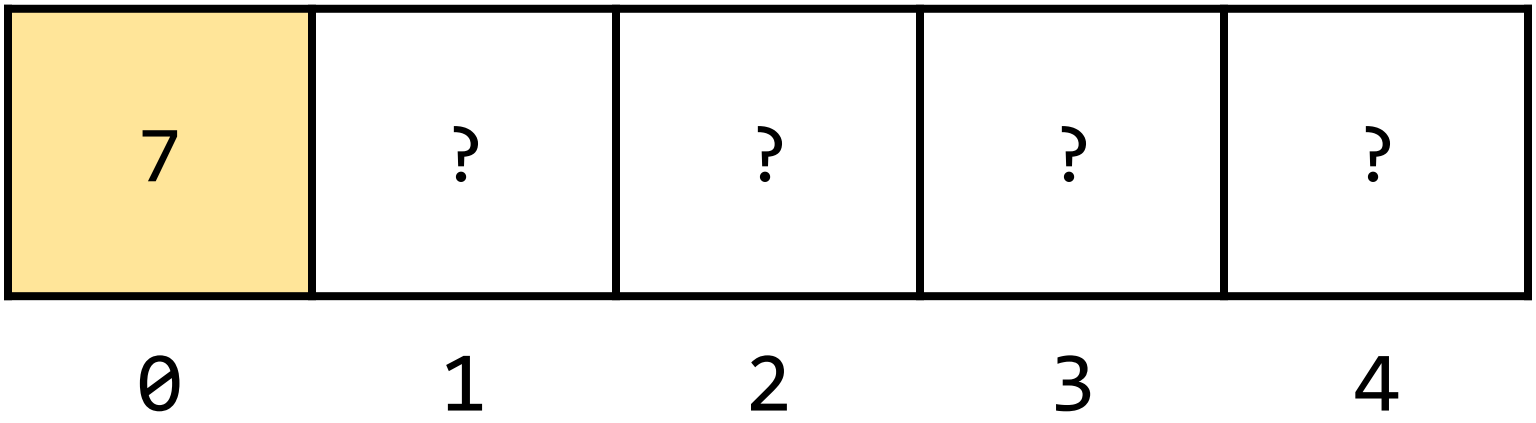
```
int hours[5];
```

hours



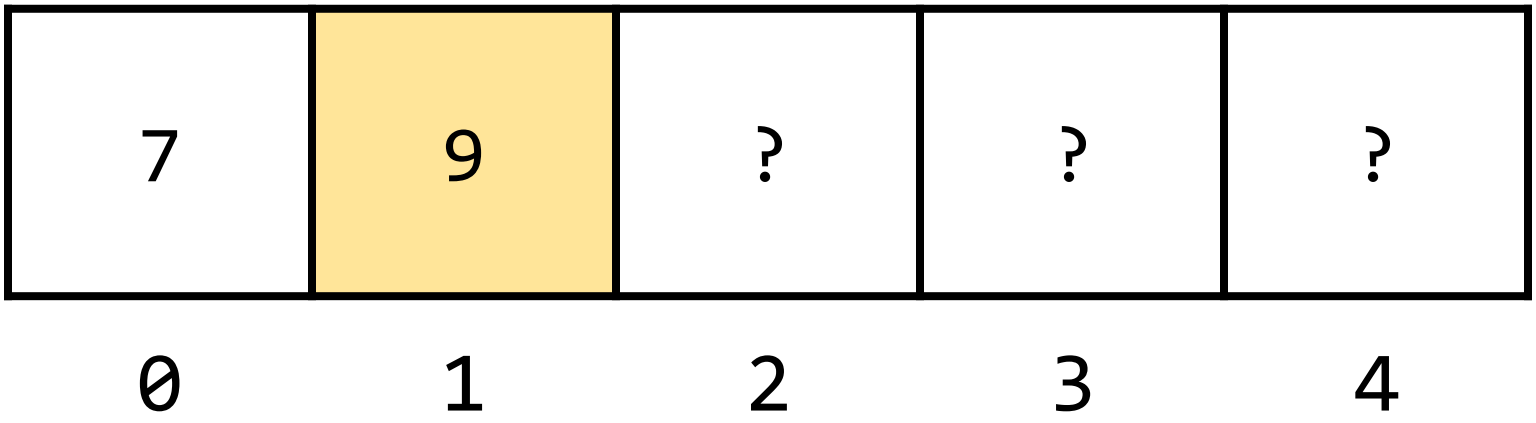
```
int hours[5];
hours[0] = 7;
```

hours



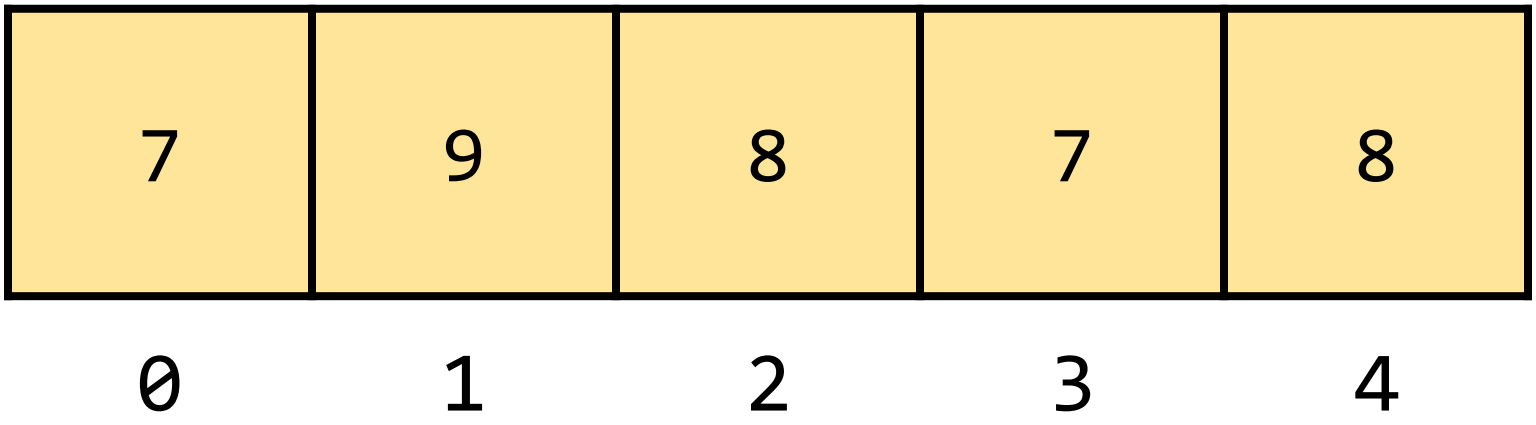
```
int hours[5];
hours[0] = 7;
hours[1] = 9;
```

hours



```
int hours[5] = {7, 9, 8, 7, 8};
```

hours



iterating through an array?

```
int hours[5] = {7, 9, 8, 7, 8};

for (int i = 0; i < 5; i++)
{
    printf("%i\n", hours[i]);
}
```

```c
int hours[5] = {7, 9, 8, 7, 8};

for (int i = 0; i < 5; i++)
{
    printf("%i\n", hours[i]);
}
```

Array Exercise

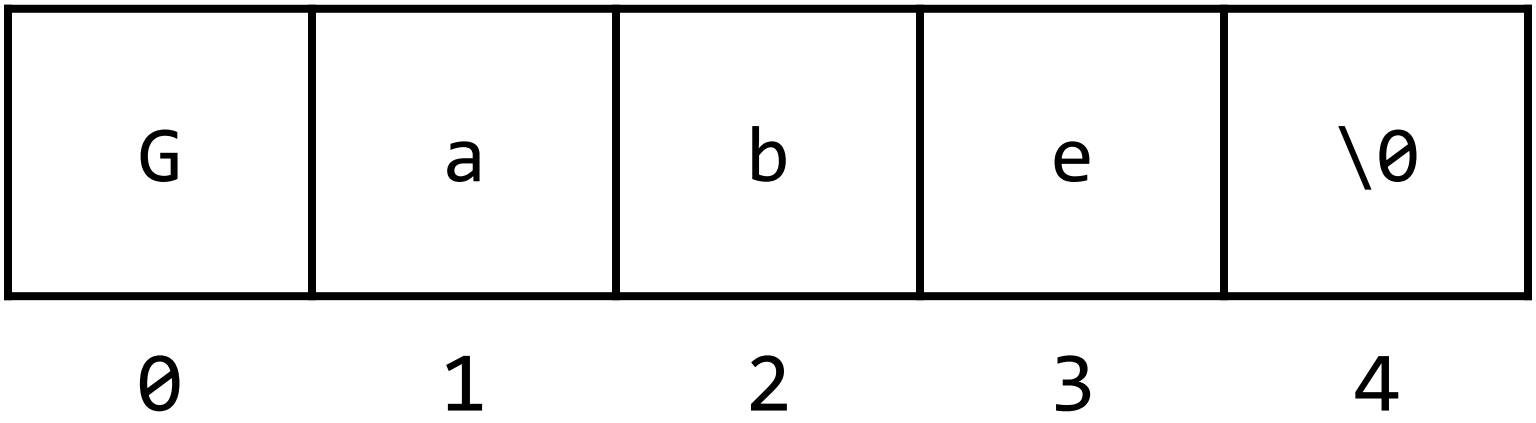
Create an array of size 5 where each element is two times the previous and the first element is 1.

Print the array, integer by integer.

Strings

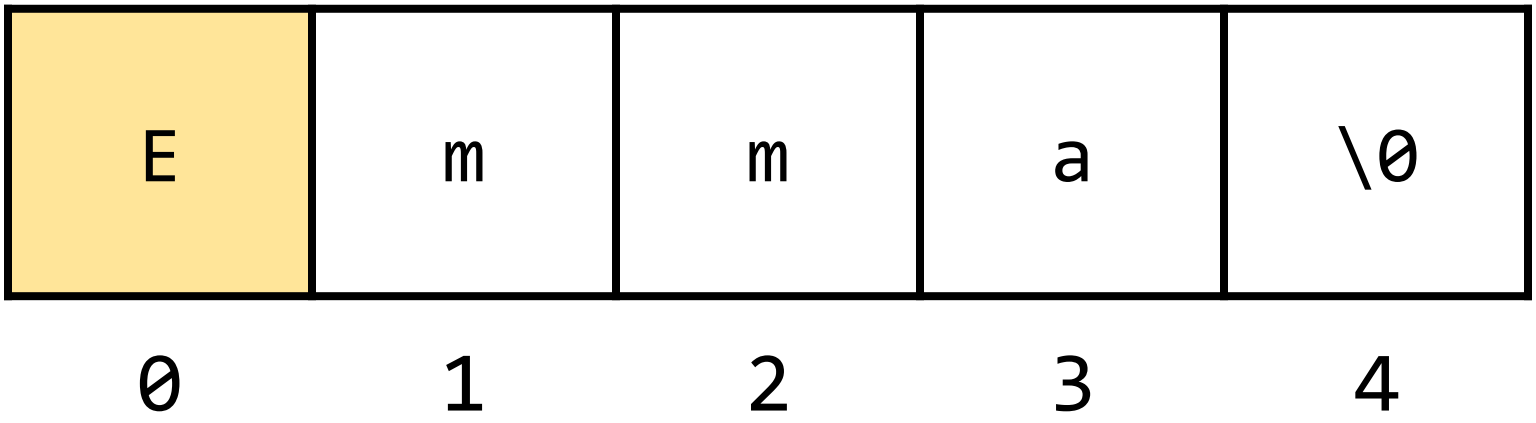
```
string name = "Gabe";
```

name



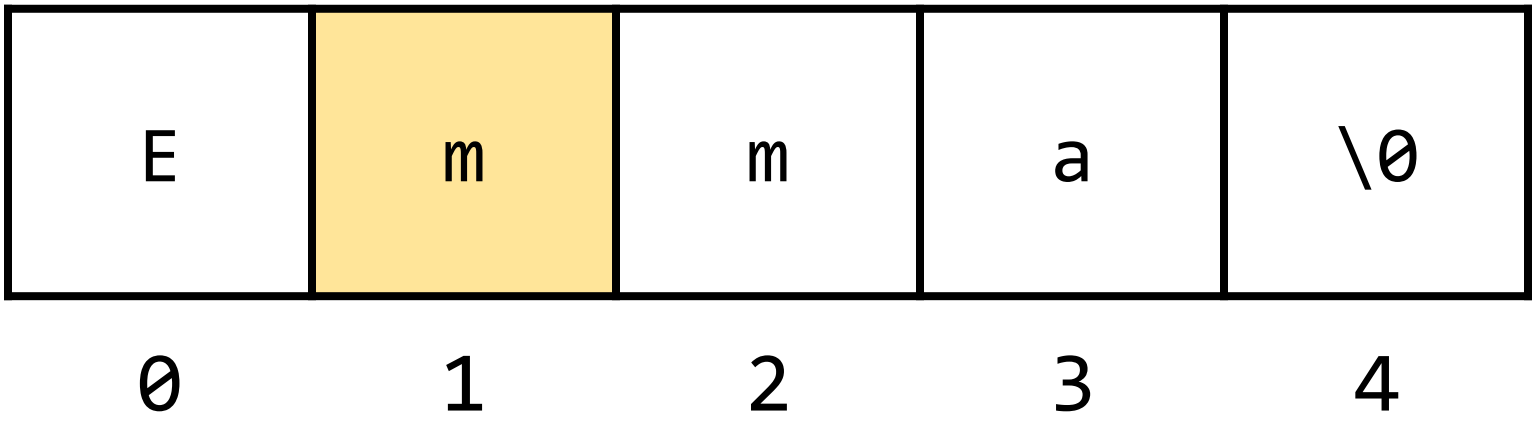
```
name[0];
```

```
name
```



```
name[1];
```

```
name
```



Character-by-Character Exercise

Create a string and print the string character by character.

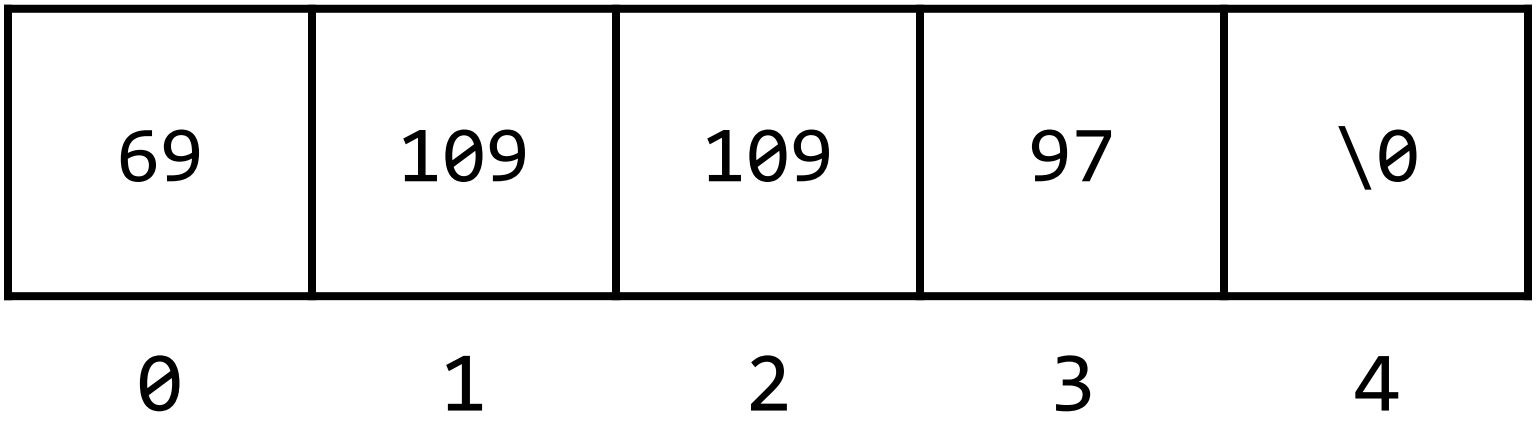
Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-_	63	3F	?	95	5F	_	127	7F	DEL

A	B	C	...	Z
65	66	67	...	90

a	b	c	...	z
97	98	99	...	122

```
string name = "Emma";
```

name



```
int ex1 = ‘B’ - ‘A’;
```

```
string name = “Gabe”

printf(“%c”, name[0]);
```

```
string name = “Gabe”

printf(“%i”, name[0]);
```

Alphabetical Exercise

Check if a lowercase string's characters are in alphabetical order. If yes, print "Yes". If no, print "No".

asciichart.com

functions


Sun Chips
Doritos
Lay's
Fritos
Cheetos
SNICKERS


input
output

input

- cash
- snack code

output

- snack

input

- (dirty) clothes
- detergent
- mode

output

- (clean) clothes

strlen(string)

isupper(char)

islower(char)

manual.cs50.io

Command-line Arguments

What are some examples of programs we've seen that take command-line arguments?

```
$ make mario
```

```
$ check50 cs50/...
```

```
$ ./caesar 13
```

```
int calculate_quarters(int cents)
{
    ...
}
```

Function argument(s)

```
int calculate_quarters(int cents)
{
    ...
}
```

Return type

```
int calculate_quarters(int cents)
{
    ...
}
```

```
int main(void)
{
    ...
}
```

```
int main(int argc, string argv[])
{
    ...
}
```

```
$ make mario
```

argv[0] argv[1]

```
$ ./caesar 13
```

```
$ ./initials Carter Zenke
```

```
$ ./initials Carter Zenke
```

argv[1] argv[2]

```
$ ./initials Carter Zenke
```

argv[1][0] argv[2][0]

Scrabble

- Work an example yourself
- Write down exactly what you did
- Create a generalization (algorithm) after working multiple examples
- Test your algorithm by hand
- Translate your algorithm to code
- Find bugs in your code by running test cases
- Debug (and critique) your code

- Work an example yourself
- Write down exactly what you did
- Create a generalization (algorithm) after working multiple examples
- Test your algorithm by hand
- Translate your algorithm to code
- Find bugs in your code by running test cases
- Debug (and critique) your code

- What **syntax** should we use to access each individual character of a string?
- How should we get the **point value** of a character?
- How should our program handle **uppercase** and **lowercase** inputs differently?

- Work an example yourself
- Write down exactly what you did
- Create a generalization (algorithm) after working multiple examples
- Test your algorithm by hand
- Translate your algorithm to code
- Find bugs in your code by running test cases
- Debug (and critique) your code

If feeling more comfortable…

Our Scrabble program will accept any word, whether it's correctly spelled or not! How might you check to see if a user's input is part of a list of valid words?

Office Hours