This is CS50

Section, Week 2

Agenda

- Course Logistics
- Loops
- ASCII
- Arrays
- Functions
- Command Line Arguments
- pset2

Who am I?

- Andi Peng
  - Head TA, running buddy, shoulder to cry on, etc.
- andi.peng@yale.edu

Support

- Sections!
- Office Hours
  - Mondays: TEAL Classroom 17 Hillhouse
  - Tues – Thursdays: Commons
- Shorts
- Walkthroughs
- Study50, Reference50
- CS50 Discuss

Problem Sets

- Read the WHOLE specification
- Meet us halfway!

- Grading
  - Psets: 50%
    - scope × (3 × correctness + 2 × design + 1 × style)
    - Lowest score dropped – only if full scope!
  - Quizzes: 40%
  - Final Project: 10%

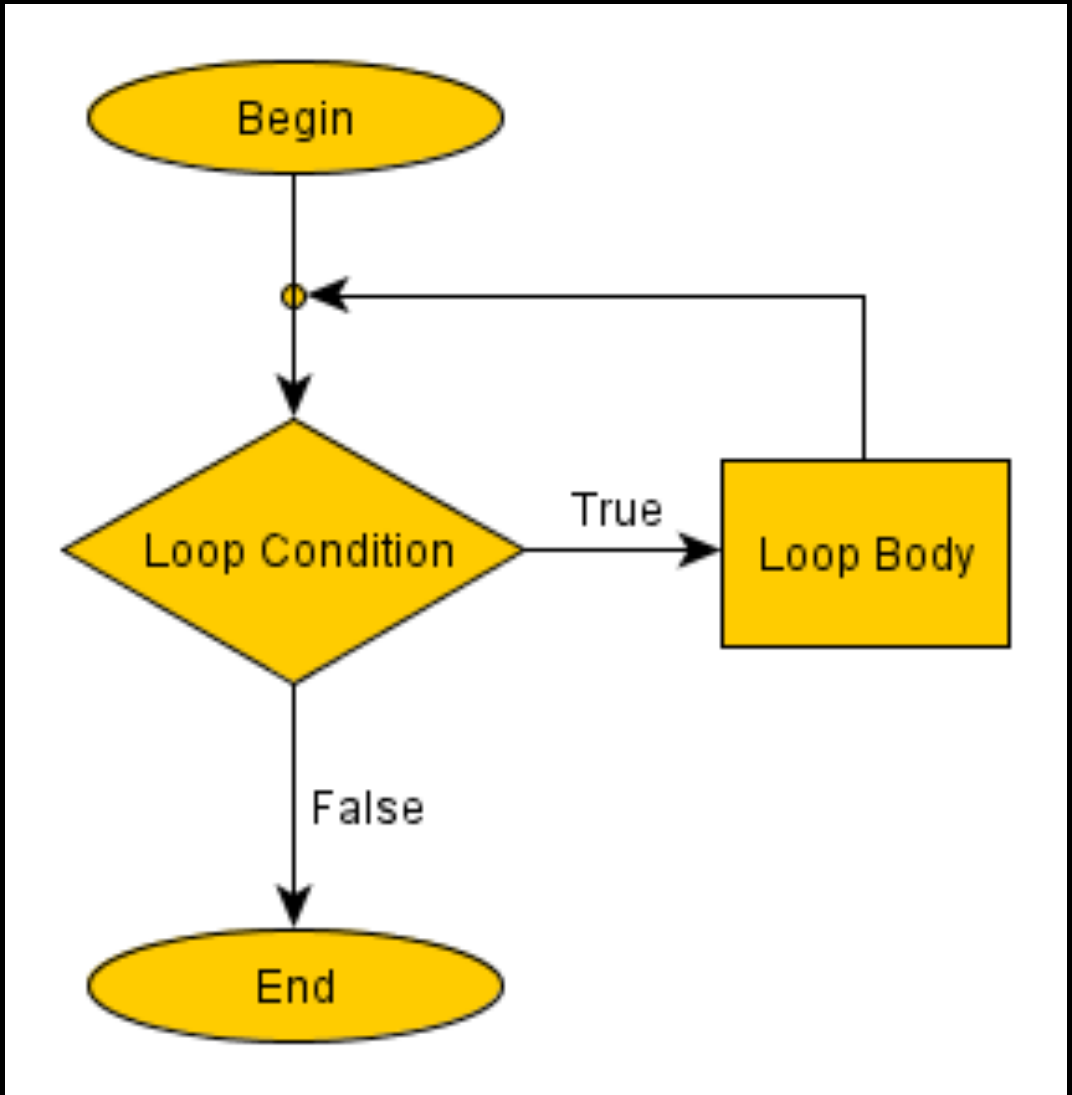
Academic Honesty

All work that you submit in this course must be your own.

- “Be reasonable”
- When in doubt, DON’T DO IT
- Regret clause
  - 72 hours

While Loops

```
while (condition)
{
// do this repeatedly
}
```



Do While Loops

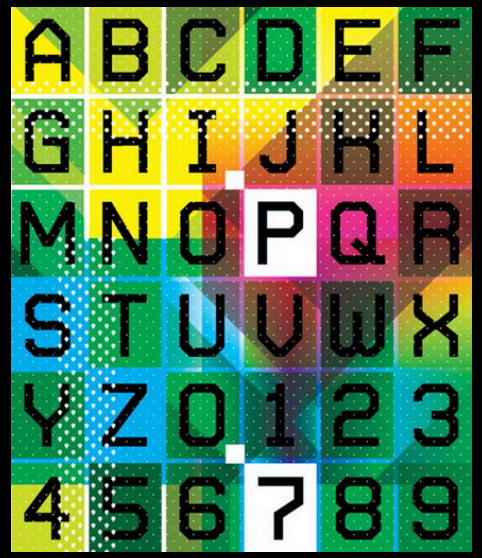
```
do
{
// do this repeatedly
}
while (condition);
```

For Loops

```
for (int i = 0; i < 10; i++)
{
    printf("I love SAJ!\n");
}
```

ASCII

- Characters must be encoded in binary
- ASCII maps characters to numbers



ASCII

A	B	C	D	E	F	G	H	I	J	K	L	M	...
65	66	67	68	69	70	71	72	73	74	75	76	77	

a	b	c	d	e	f	g	h	i	j	k	l	m	...
97	98	99	100	101	102	103	104	105	106	107	108	109	

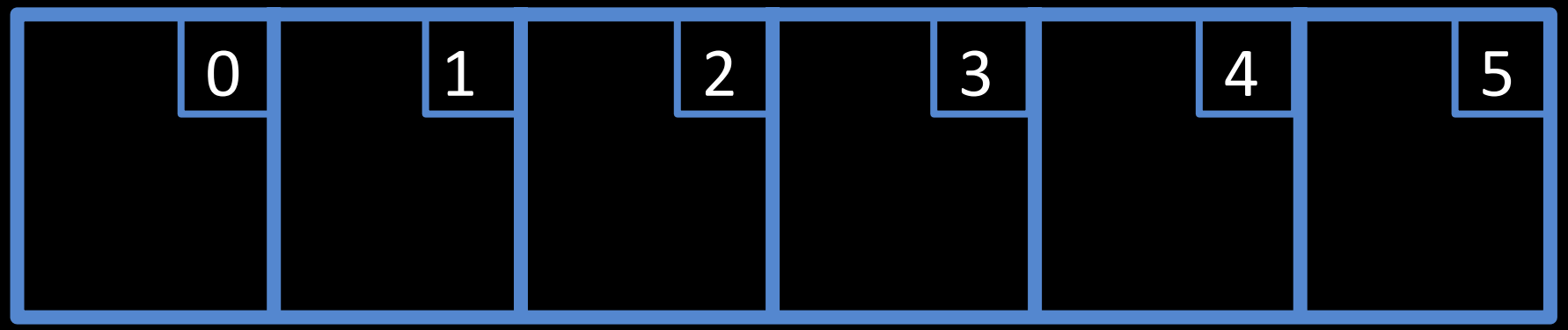
ASCII Math

- ASCII maps characters to numbers

What will print?

```
printf("%d\n", 'a' - 'A');
printf("%c\n", 'B' + ('a' - 'A'));
printf("%c\n", 'b' - ('a' - 'A'));
printf("%c\n", 'B' + 1);
printf("%c\n", ('z' - 'a' + 1) % 26 + 'a');
```

Arrays

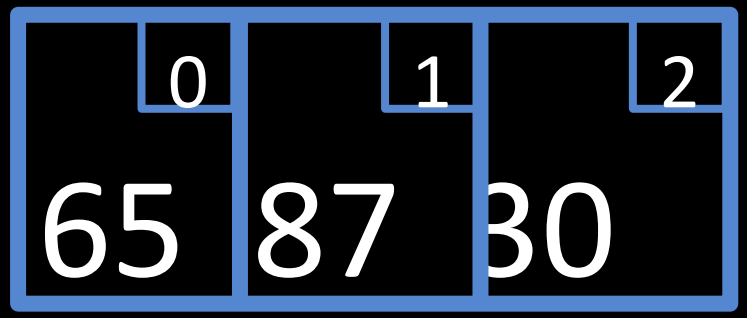


Creating an Array

```
<data type> name[<size>];
```

Example:

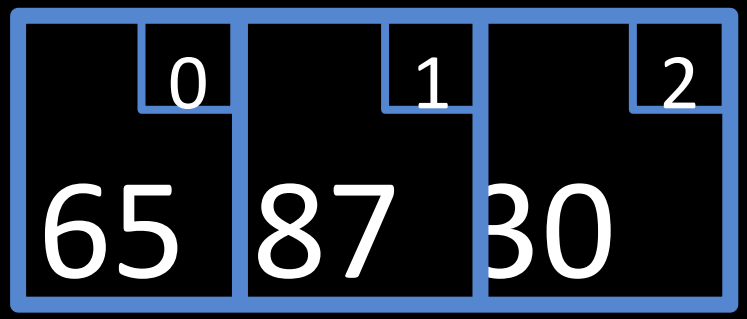
```
int temperature[3];
temperature[0] = 65;
temperature[1] = 87;
temperature[2] = 30;
```



OR

```
int temperature[] = { 65, 87, 30 };
```

Accessing Array Elements



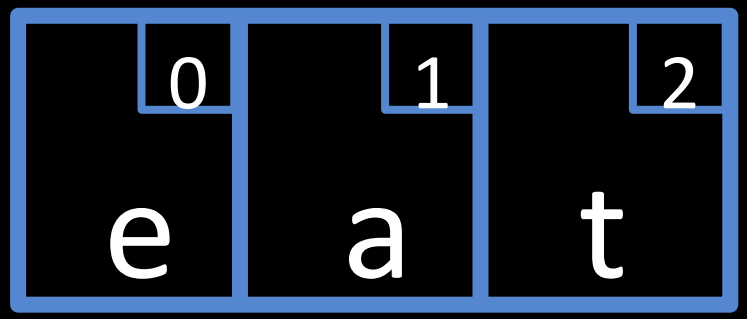

```
for (int i = 0; i < 3; i++)
{
        printf("%d\n", temperature[i]);
}
```

```c
#include <stdio.h>
#include <cs50.h>

int main(void)
{
    // declare array
    int scores[18];

    // populate array
    for (int i = 0; i < 18; i++)
    {
    printf("Enter score for student %d: ", i + 1);
    scores[i] = GetInt();
    }
}
```

Strings




```
string word = GetString();

for (int i = 0, n = strlen(word); i < n; i++)
{
        printf(”%c\n", word[i]);
}
```

Where’s the Bug?

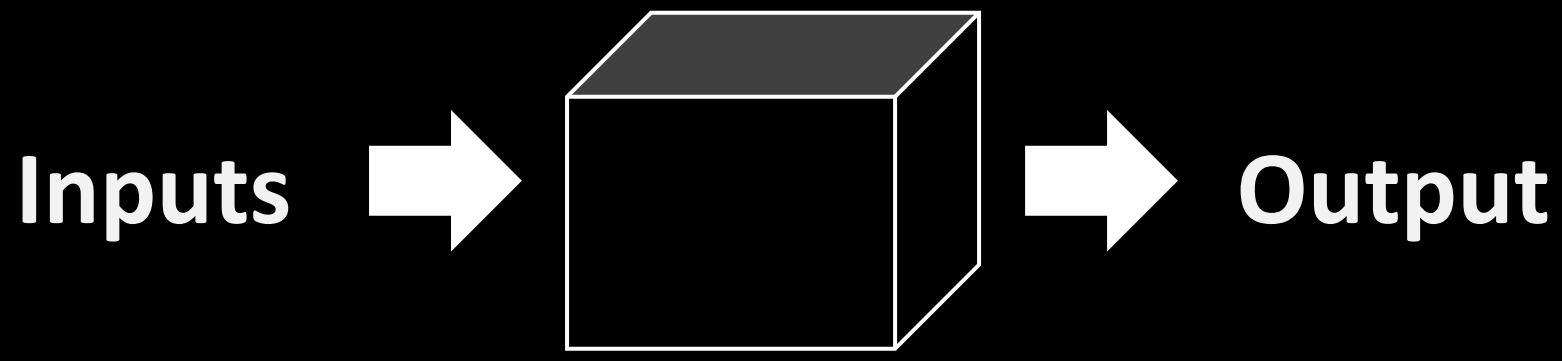
```
string class[3] = { "Sam", "Jess", "Kim" };

for (int i = 0; i <= 3; i++)
{
      printf("%s\n", class[i]);
}
```

Problem:

Create a program, upper.c, that converts a lowercase string to an uppercase string

Functions



Why Functions?

- Organization
- Simplification
- Reusability

A Function Definition

```
int cube(int input)
{
    int output = input * input * input;
    return output;
}
```

Header

function name

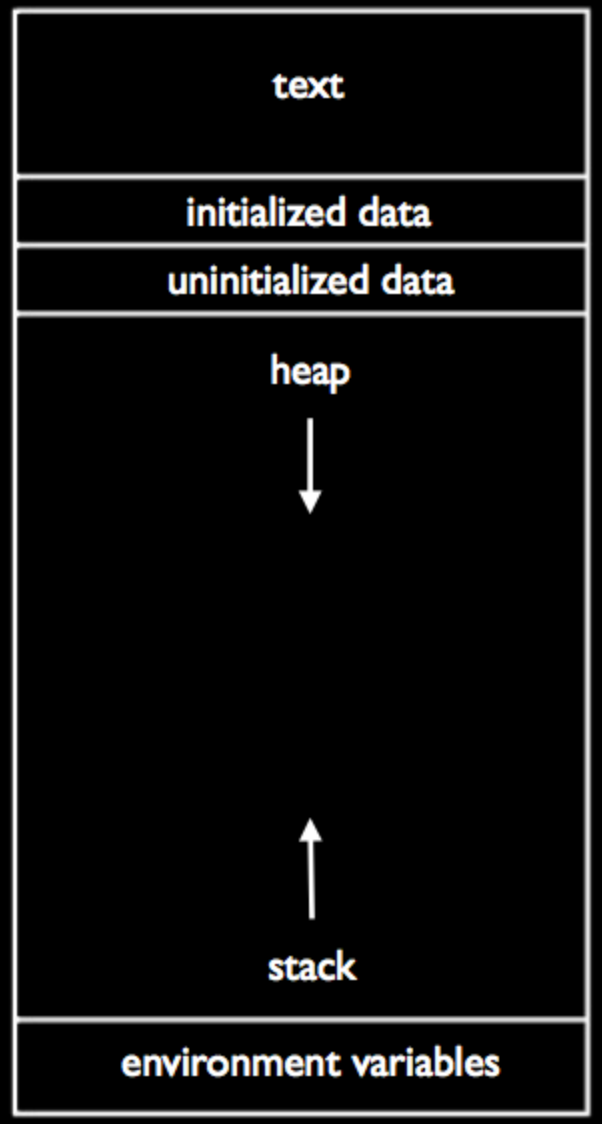
return type

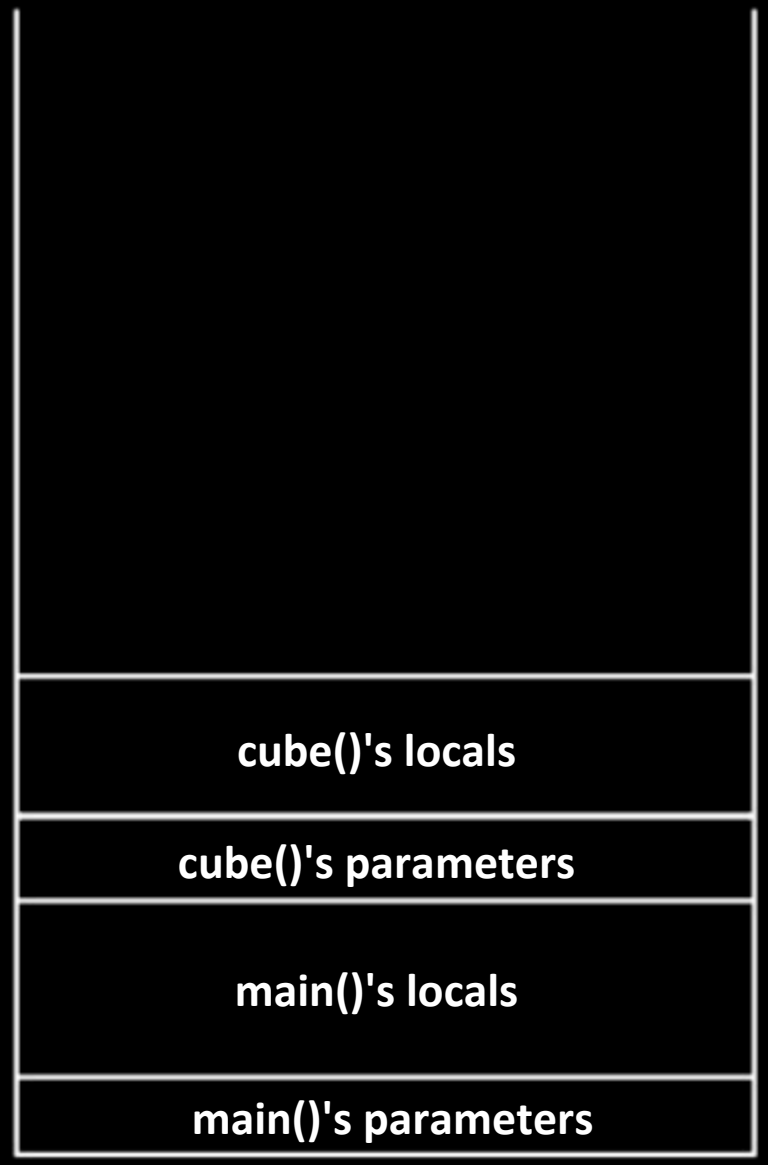
parameter list

```
int cube(int input)
{
    int output = input * input * input;
    return output;

}
```

Body


text
initialized data
uninitialized data
heap
stack
environment variables


cube()'s locals
cube()'s parameters
main()'s locals
main()'s parameters

```c
#include <stdio.h>
void swap(int a, int b);

int main(void)
{
    int x = 1;
    int y = 2;
    swap(x, y);
    printf("x is %i\n", x);
    printf("y is %i\n", y);
}

void swap(int a, int b)
{
    int tmp = a;
    a = b;
    b = tmp;
}
```

Command Line Arguments

```
int main(void)

int main(int argc, string argv[])
```

Test Yourself

```
jharvard@appliance (~): ./copy infile outfile
```

1. What is argc?
2. What is argv[0]?
3. What is argv[1]?
4. What is argv[2]?
5. What is argv[3]?
6. What is argv[4]?

Pset2: Crypto

- `initials.c`
- `caesar.c`
- `vigenere.c`

http://www.nuitari.de/crypto.html

Pset2: Crypto

atoi()

- Converts a string to an int
- argv will be a string so we need to change it to an int

modulo %

- What if we are given really large numbers for Caesar?
- What happens when we reach the end of the string in Vigenere?

ctype.h

- isupper(), toupper(), isalpha(), etc…