This is CS50

Week 1

Yuliia Zhukovets

Preceptor

yuliia@cs50.harvard.edu

Agenda

- Variables and Operators
- Functions
- Loops
- Problem Set 1

Part 1

Variables and Types
Input and Printing

Variables

`calls`

Variables

```
int calls = 4;
```

Variables

```
int calls = 4;
```

name

calls

Variables

```
int calls = 4;
```

type

calls

Variables

```
int calls = 4;
```

value

calls

Variables

```
int calls = 4;
```

assignment operator

calls

Variables

```
int calls = 4;
```

type name | value

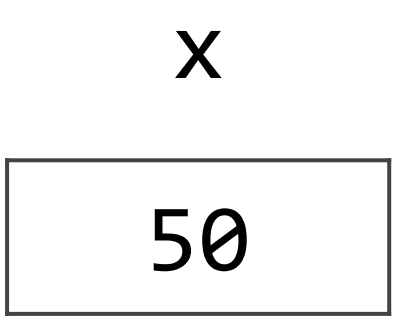
assignment operator

calls

"Create an **integer** named **calls** that **gets** the **value 4**."

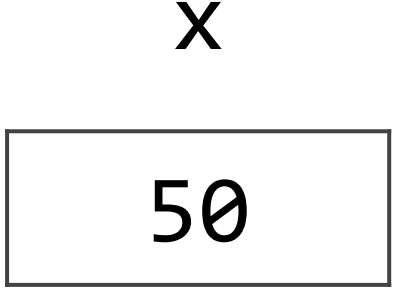
Variables

```
int x = 50;
```



Variables

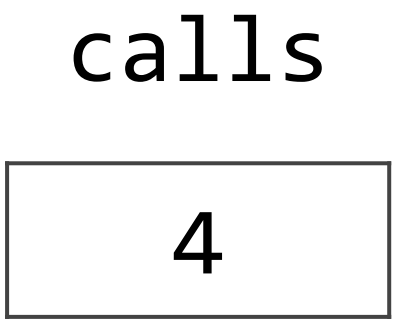
```
int x = 50;
```



"Create an **integer** named **x** that **gets** the **value 50**."

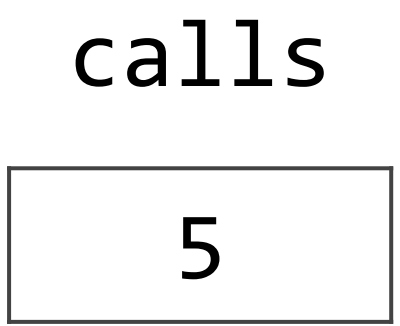
Variables

```
int calls = 4;
calls = 5;
```



Variables

```
int calls = 4;
calls = 5;
```



Variables

```
int calls = 4;
calls = 5;
```

name | value

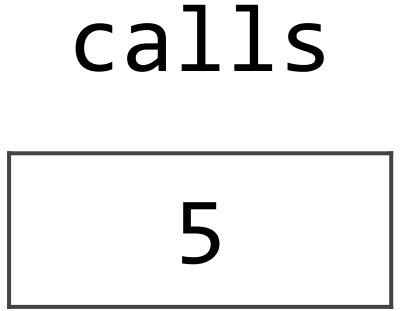
assignment operator

calls

"Calls gets 5."

Operators

```
int calls = 4;
calls = calls + 1;
```

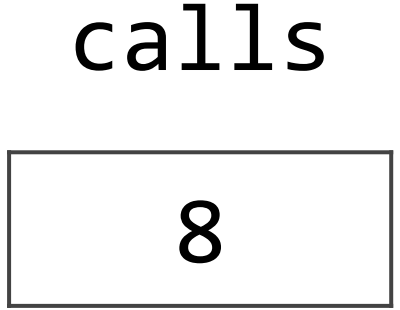


Operators

```
int calls = 4;
calls = calls - 1;
```

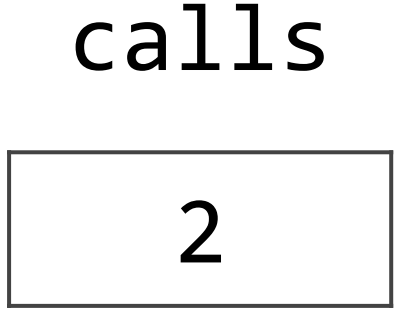
Operators

```
int calls = 4;
calls = calls * 2;
```



Operators

```
int calls = 4;
calls = calls / 2;
```



Getting input

```
int calls = get_int("Calls: ");
```

type

name

assignment operator

function

Functions

```
int calls = get_int("Calls: ");
```

function

Functions

```
int calls = get_int("Calls: ");
```

function name

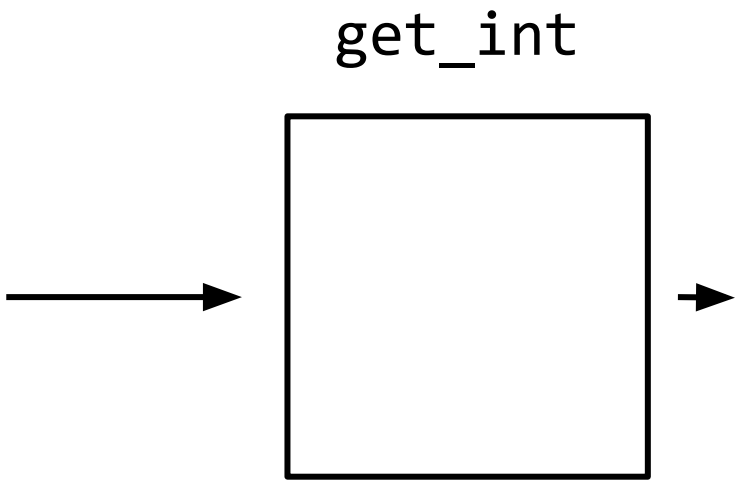
Functions

```
int calls = get_int("Calls: ");
```

function input

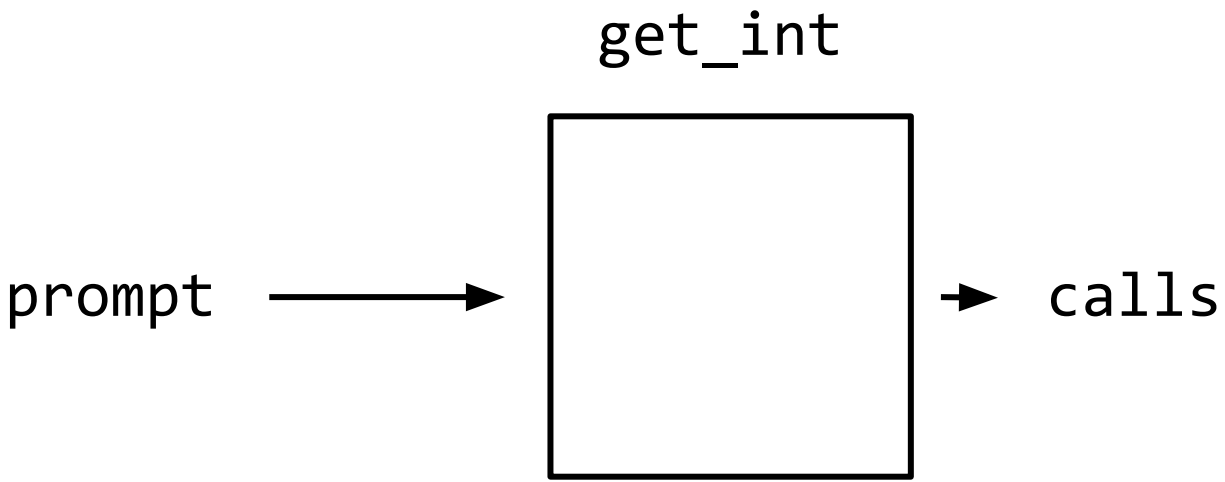
Functions

```
int calls = get_int("Calls: ");
```



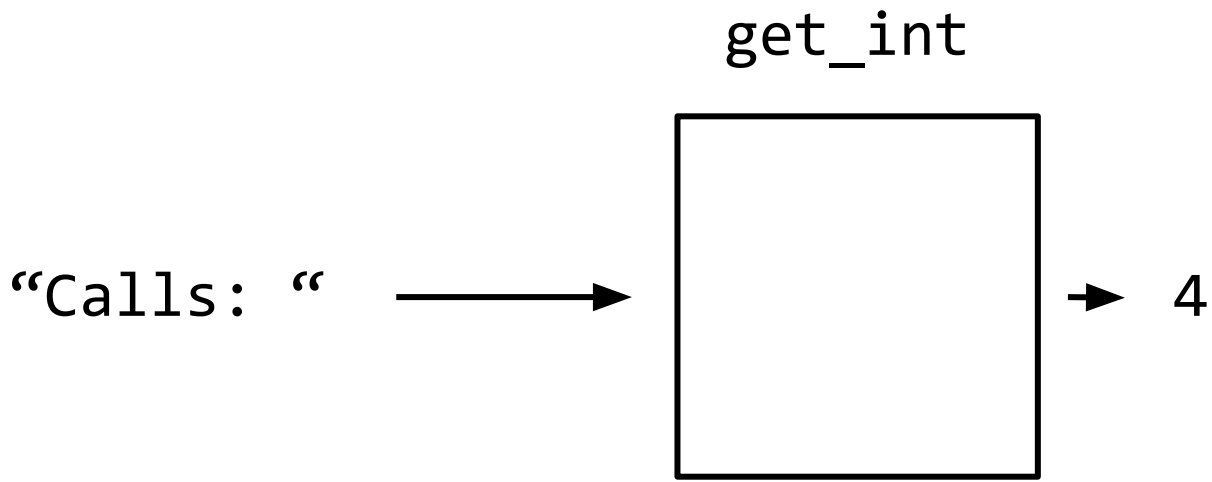
Functions

```
int calls = get_int("Calls: ");
```



Functions

```
int calls = get_int("Calls: ");
```



Return values

```
int calls = 4;
```

value

Storing return values

```
int calls = 4;
```

type name | value

assignment operator

calls

"Create an **integer** named **calls** that **gets** the **value 4**."

Printing values

```
int calls = 4;
printf("calls equals %i", calls);
```

Printing values

```
int calls = 4;
printf("calls equals %i", calls);
```

format code

Printing values

```
int calls = 4;
printf("calls equals %i", calls);
```

Types and format codes

Numbers	Text	True/False
`int (%i)`	`char (%c)`	`bool (%i)`
`float (%f)`	`string (%s)`	

Hello, world

- Let's write a program that prints out "*Hello, world*" to complete the first step of Problem Set 0.

Hello, me

- Let’s write a program “*Hello, me*” to complete the second step of Problem Set 0.

Hello, friends!

- Let’s write a program that stores and prints out some information (like name, age, phone number) of your friends.

Part 2

Breaking down loops and conditionals

```
if (calls < 1)
{
    printf("Call more often!");
}
```

boolean expression

```
if (calls < 1)
{
    printf("Call more often!");
}
```

conditional

```
if (calls < 1)
{
    printf("Call more often!");
}
```

```
if (calls < 1)
{
    printf("Call more often!");
}
```

conditional code

```
if (calls < 1)
{
    printf("Call more often!");
}
else
{
    printf("Thanks for calling!");
}
```

```
if (calls < 1)
{
    printf("Call more often!");
}
else
{
    printf("Thanks for calling!");
}
```

mutually exclusive

```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```

initialization

```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```

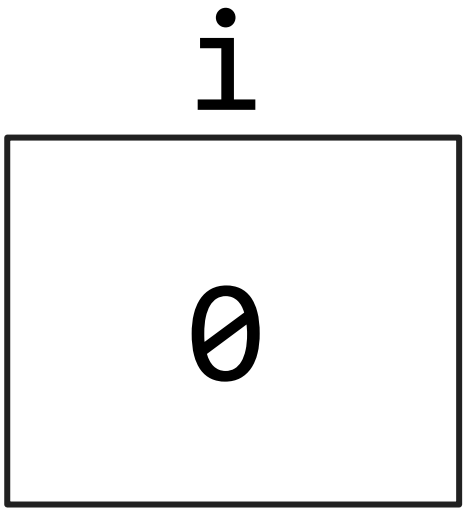
boolean expression

```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```

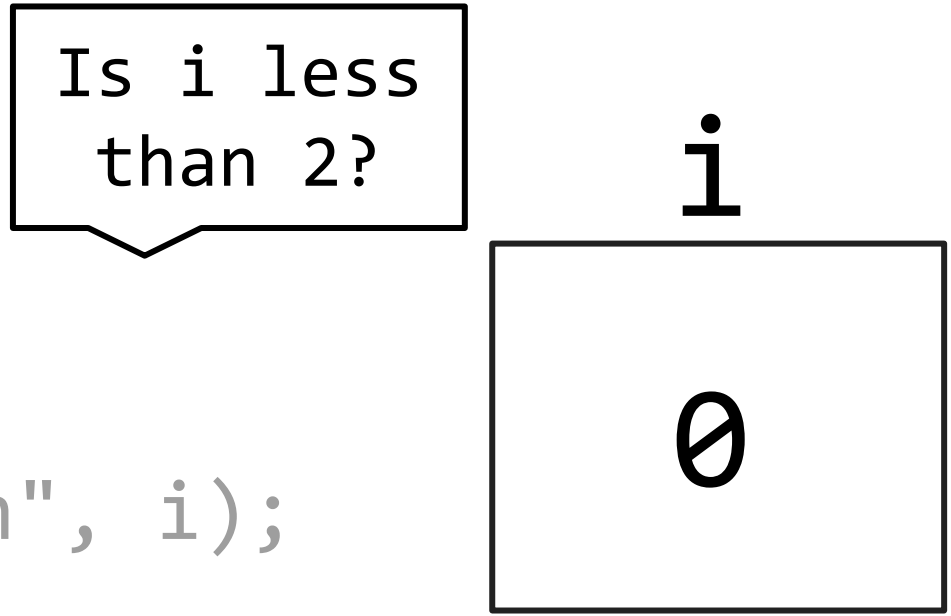
```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```

increment

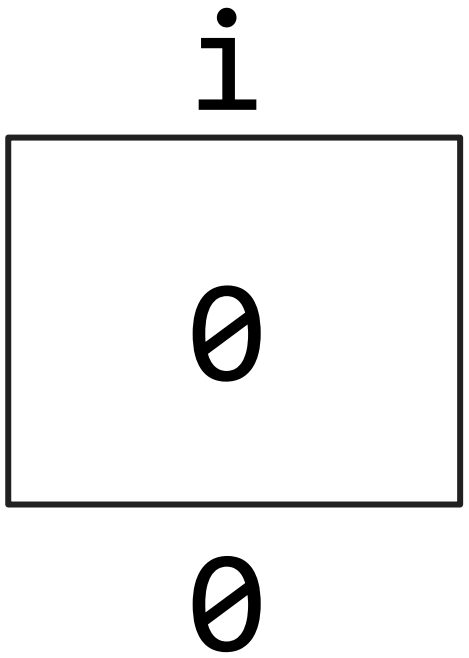
```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```



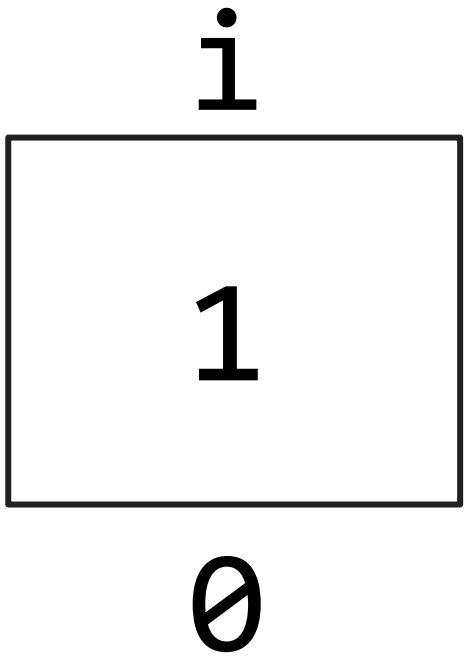
```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```



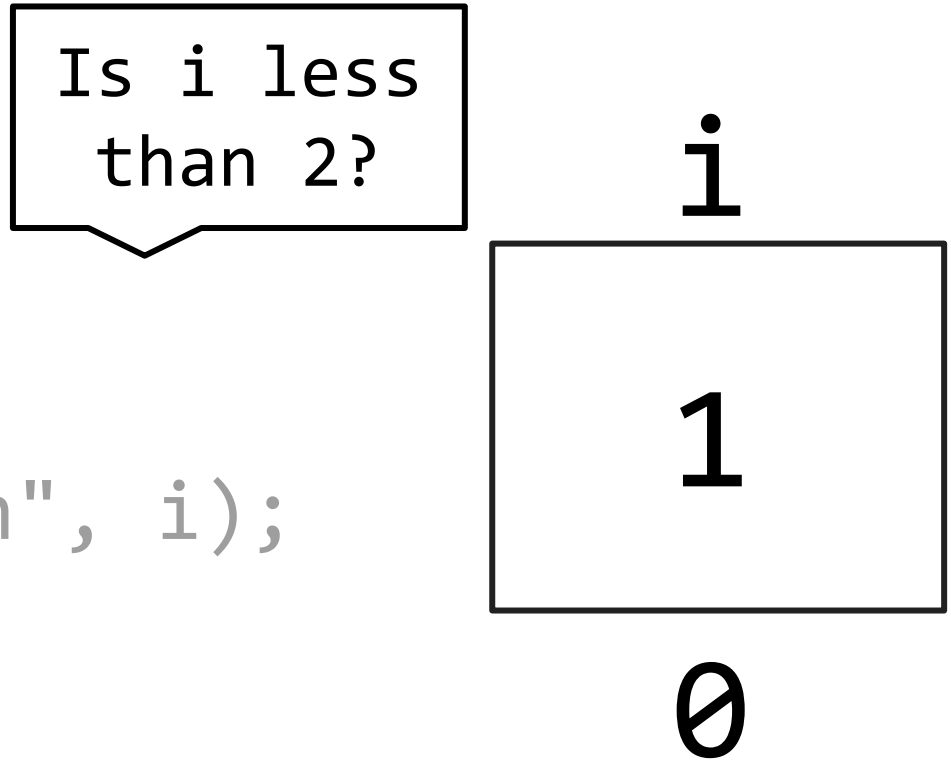
```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```



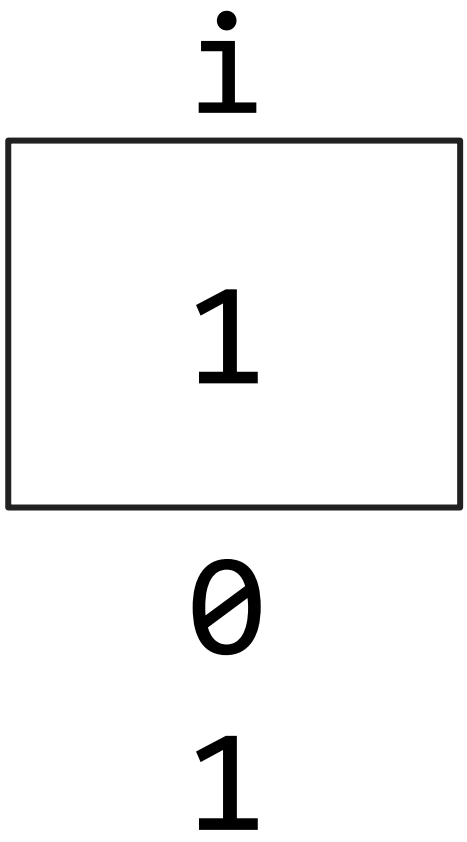
```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```



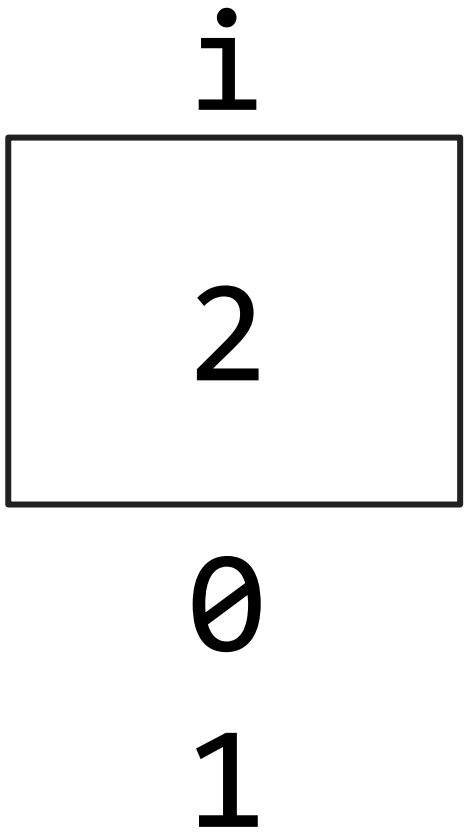
```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```



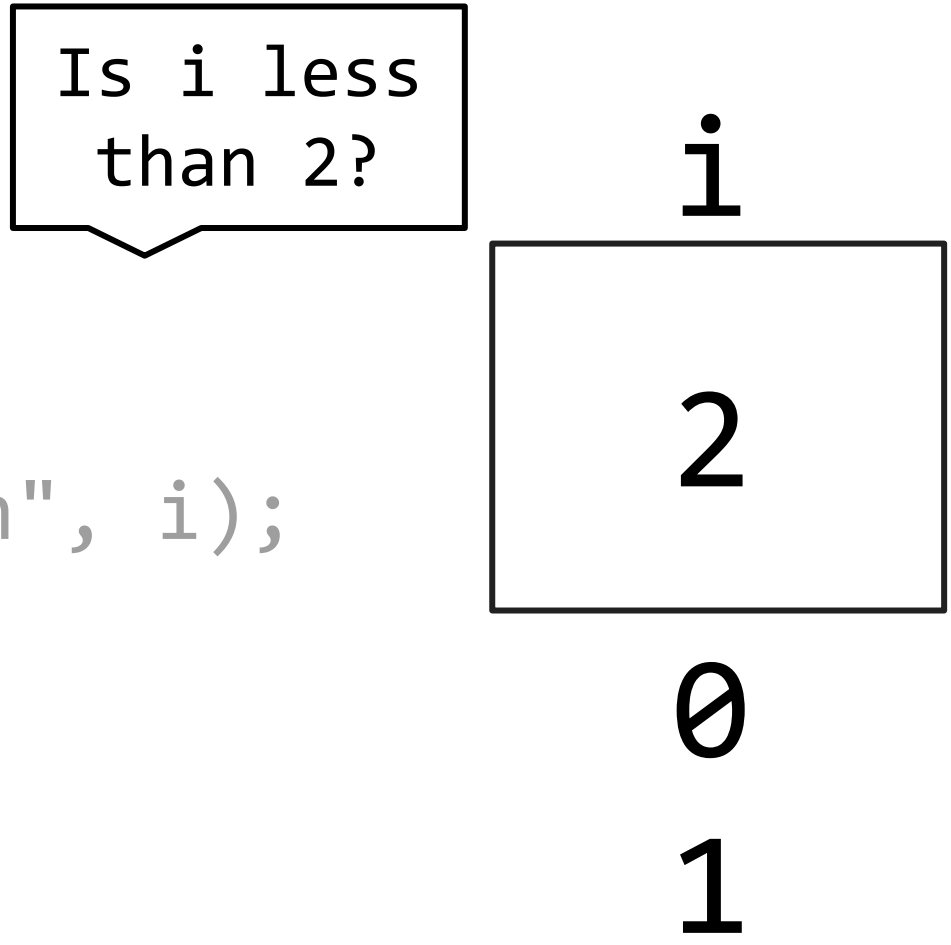
```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```



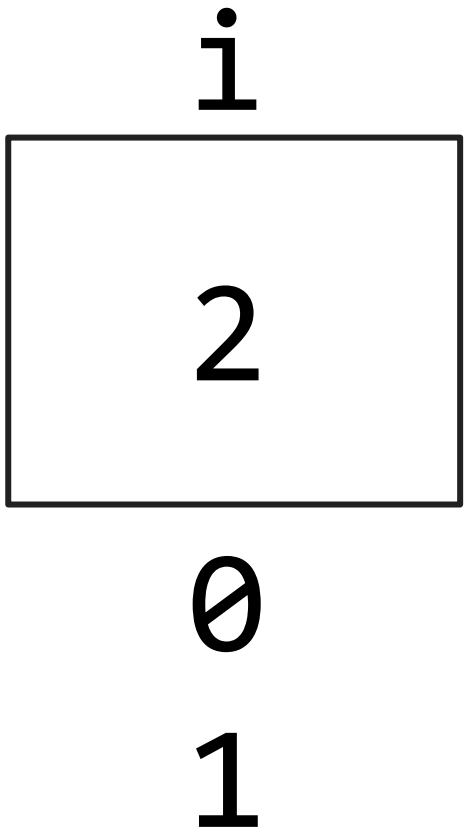
```c
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```



```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```



```
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```



```c
int i = 0;
while (i < 2)
{
    printf("%i\n", i);
    i = i + 1;
}
```

```
for (int i = 0; i < 2; i++)
{
    printf("%i\n", i);
}
```

initialization

```
for (int i = 0; i < 2; i++)
{
    printf("%i\n", i);
}
```

boolean expression

```
for (int i = 0; i < 2; i++)
{
    printf("%i\n", i);
}
```

increment

```
for (int i = 0; i < 2; i++)
{
    printf("%i\n", i);
}
```

Part 3 Mario


MARIO
000700
×01
WORLD
1-1
TIME
287


MARIO
000700
×01
WORLD
1-1
TIME
287

#

##

###

####

######

#######

Let’s start with a left-aligned pyramid first!

```
#
##
###
####
######
#######
```

```
void print_row(int bricks)
{
    # Print row of bricks
}
```

return type

```
void print_row(int bricks)
{
    # Print row of bricks
}
```

function name

```
void print_row(int bricks)
{
    # Print row of bricks
}
```

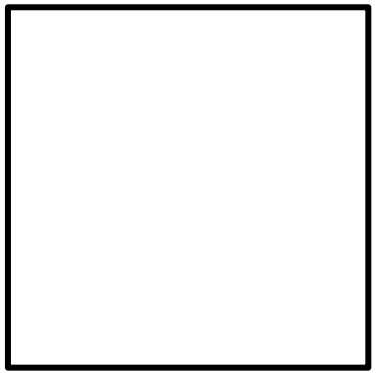
input

```
void print_row(int bricks)
{
    # Print row of bricks
}
```

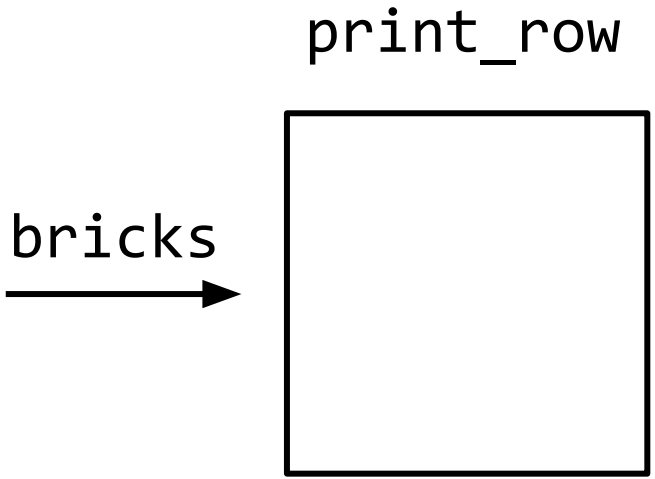
```
void print_row(int bricks)
{
    # Print row of bricks
}
```

```
void print_row(int bricks)
{
      # Print row of bricks
}
```

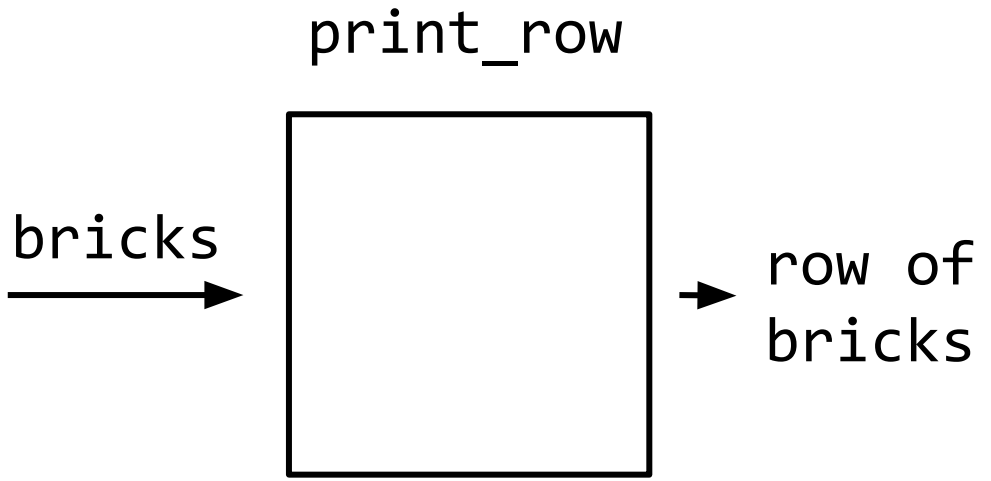
print_row



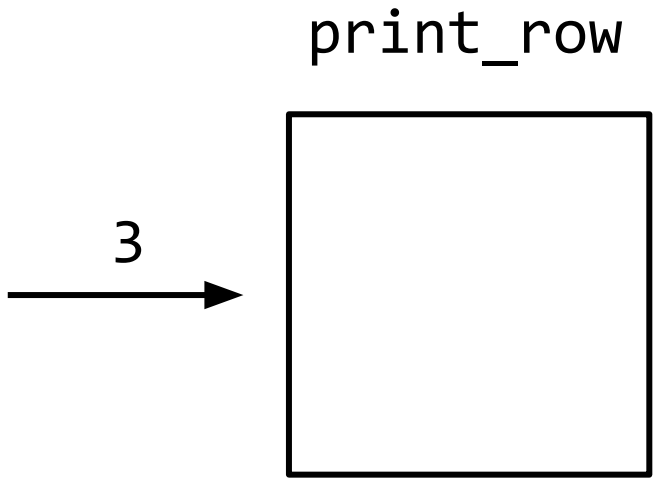
```
void print_row(int bricks)
{
      # Print row of bricks
}
```



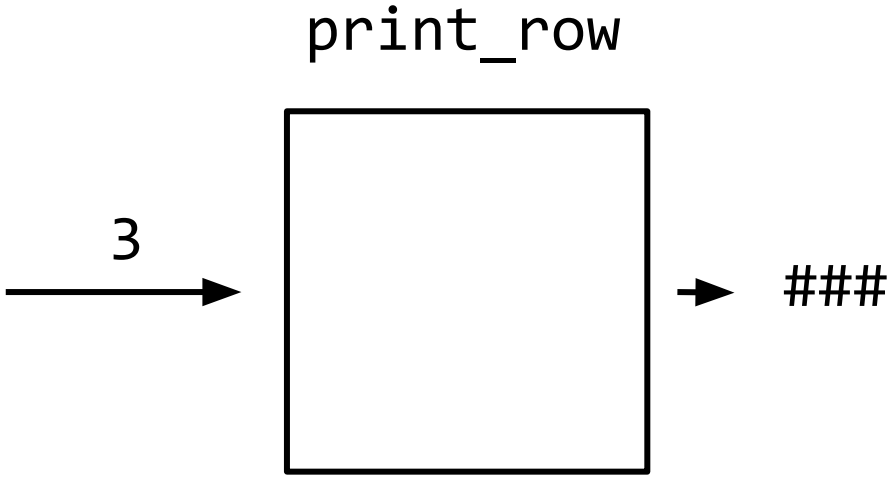
```
void print_row(int bricks)
{
      # Print row of bricks
}
```



```
void print_row(int bricks)
{
      # Print row of bricks
}
```



```
void print_row(int bricks)
{
      # Print row of bricks
}
```



This is CS50

Week 1