


This is CS50

CS50's Introduction to Computer Science

OpenCourseWare

Donate  (<https://cs50.harvard.edu/donate>)

David J. Malan (<https://cs.harvard.edu/malan/>)

malan@harvard.edu

 (<https://www.facebook.com/dmalan>)  (<https://github.com/dmalan>) 

(<https://www.instagram.com/davidjmalan/>)  (<https://www.linkedin.com>

[in/malan/](https://www.linkedin.com/in/malan/))  (<https://www.reddit.com/user/davidjmalan>) 

(<https://www.threads.net/@davidjmalan>)  (<https://twitter.com/davidjmalan>)

Mario



Problem to Solve

Toward the end of World 1-1 in Nintendo's [Super Mario Bros.](https://en.wikipedia.org/wiki/Super_Mario_Bros.) (https://en.wikipedia.org/wiki/Super_Mario_Bros.), Mario must ascend right-aligned pyramid of bricks, as in the below.



In a file called `mario.c` in a folder called `mario-less`, implement a program in C that recreates that pyramid, using hashes (`#`) for bricks, as in the below:

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

But prompt the user for an `int` for the pyramid's actual height, so that the program can also output shorter pyramids like the below:

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

Re-prompt the user, again and again as needed, if their input is not greater than 0 or not an `int` altogether.

▼ Hints

- Recall that you can get an `int` from a user with `get_int`, which is declared in `cs50.h`.
- Recall that you can print a `string` with `printf`, which is declared in `stdio.h`.

Demo

```
$ make mario
$ ./mario
Height: 8
  #
 ##
###
####
#####
#####
#####
#####
#####
$
```

Recorded with [asciinema](#)

Advice

▼ Write some code that you know will compile

Even though this program won't do anything, it should at least compile with `make`!

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

int main(void)
{

}
```

▼ Write some pseudocode before writing more code

If unsure how to solve the problem itself, break it down into smaller problems that you can probably solve first. For instance, this problem is really two problems:

1. Prompt the user for the pyramid's height
2. Print a pyramid of that height

So write some pseudocode as comments that remind you to do just that:

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

int main(void)
```

```
{  
    // Prompt the user for the pyramid's height  
  
    // Print a pyramid of that height  
}
```

▼ Convert the pseudocode to code

First, consider how you might prompt the user for the pyramid's height. Recall that a `do while` loop is helpful when you want to do something at least once, and possibly again and again, as in the below:

```
#include <cs50.h>  
#include <stdio.h>  
  
int main(void)  
{  
    // Prompt the user for the pyramid's height  
    int n;  
    do  
    {  
        n = get_int("Height: ");  
    }  
    while (n < 1);  
  
    // Print a pyramid of that height  
}
```

Second, consider how you might print a pyramid of that height, from top to bottom. Notice how the first row should have one brick, the second row should have two bricks, and so on. Odds are you'll want a loop, as in the below, even if not (yet!) sure what to put in that loop. So add some more pseudocode as a comment for now:

```
#include <cs50.h>  
#include <stdio.h>  
  
int main(void)  
{  
    // Prompt the user for the pyramid's height  
    int n;  
    do  
    {  
        n = get_int("Height: ");  
    }  
    while (n < 1);  
  
    // Print a pyramid of that height  
    for (int i = 0; i < n; i++)  
    {  
        // Print row of bricks  
    }  
}
```

How to print that row of bricks? Well, wouldn't it be nice if there were a function called `print_row` that could do just that? Let's suppose that there is:

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

void print_row(int bricks);

int main(void)
{
    // Prompt the user for the pyramid's height
    int n;
    do
    {
        n = get_int("Height: ");
    }
    while (n < 1);

    // Print a pyramid of that height
    for (int i = 0; i < n; i++)
    {
        // Print row of bricks
    }
}

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

We could then call that function from `main`, as in the below:

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

void print_row(int bricks);

int main(void)
{
    // Prompt the user for the pyramid's height
    int n;
    do
    {
        n = get_int("Height: ");
    }
    while (n < 1);

    // Print a pyramid of that height
    for (int i = 0; i < n; i++)
    {
        // Print row of bricks
        print_row(i + 1);
    }
}

void print_row(int bricks)
```

```
{  
    // Print row of bricks  
}
```

Why `i + 1`, though?

Let's now implement `print_row`:

```
#include <cs50.h>  
#include <stdio.h>  
  
void print_row(int bricks);  
  
int main(void)  
{  
    // Prompt the user for the pyramid's height  
    int n;  
    do  
    {  
        n = get_int("Height: ");  
    }  
    while (n < 1);  
  
    // Print a pyramid of that height  
    for (int i = 0; i < n; i++)  
    {  
        // Print row of bricks  
        print_row(i + 1);  
    }  
}  
  
void print_row(int bricks)  
{  
    for (int i = 0; i < bricks; i++)  
    {  
        printf("#");  
    }  
    printf("\n");  
}
```

Why the `\n` at the end, though?

Unfortunately, this code prints a left-aligned pyramid, but you need a right-aligned one!

Perhaps we should print some blank spaces before some of the bricks, to move them to the right? Let's change `print_row` as follows so that it can print both:

```
#include <cs50.h>  
#include <stdio.h>  
  
void print_row(int spaces, int bricks);  
  
int main(void)  
{  
    // Prompt the user for the pyramid's height
```

```
int n;
do
{
    n = get_int("Height: ");
}
while (n < 1);

// Print a pyramid of that height
for (int i = 0; i < n; i++)
{
    // Print row of bricks
}

void print_row(int spaces, int bricks)
{
    // Print spaces

    // Print bricks
}
```

Some pseudocode now remains in both `main` and `print_row`, so that we leave to you!

And consider whether you could factor out some of the code in `main` to a `get_height` function, too, that returns the `int` you need!

Walkthrough

Note this walkthrough specifies your program should prompt the user for a pyramid's height and *re-prompt* if the user inputs a value less than 1 or greater than 8. The specification only requires you to re-prompt the user if they input a value less than 1.



How to Test

Does your code work as prescribed when you input:

- `-1` or other negative numbers?
- `0`?
- `1` or other positive numbers?
- letters or words?
- no input at all, when you only hit Enter?

Correctness

In your terminal, execute the below to check your work's correctness.

```
check50 cs50/problems/2024/x/mario/less
```

Style

Execute the below to evaluate the style of your code using `style50`.

```
style50 mario.c
```

How to Submit

In your terminal, execute the below to submit your work.

```
submit50 cs50/problems/2024/x/mario/less
```