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.reddit.com/user/davidjmalan)
(https://www.threads.net/@davidjmalan) (https://twitter.com/davidjmalan)

Mario

Getting Started

Open VS Code (https://cs50.dev/).

Start by clicking inside your terminal window, then execute `cd` by itself. You should find that its "prompt" resembles the below.

```
$
```

Click inside of that terminal window and then execute

```
wget https://cdn.cs50.net/2022/fall/psets/1/mario-more.zip
```

followed by Enter in order to download a ZIP called `mario-more.zip` in your codespace. Take care not to overlook the space between `wget` and the following URL, or any other character for that matter!

Now execute

```
unzip mario-more.zip
```

to create a folder called `mario-more`. You no longer need the ZIP file, so you can execute

```
rm mario-more.zip
```

and respond with "y" followed by Enter at the prompt to remove the ZIP file you downloaded.

Now type

```
cd mario-more
```

followed by Enter to move yourself into (i.e., open) that directory. Your prompt should now resemble the below.

```
mario-more/ $
```

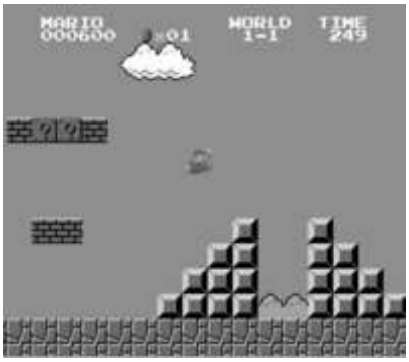
If all was successful, you should execute

```
ls
```

and see a file named `mario.c`. Executing `code mario.c` should open the file where you will type your code for this problem set. If not, retrace your steps and see if you can determine where you went wrong!

World 1-1

Toward the beginning of World 1-1 in Nintendo's Super Mario Brothers, Mario must hop over adjacent pyramids of blocks, per the below.



Let's recreate those pyramids in C, albeit in text, using hashes (`#`) for bricks, a la the below. Each hash is a bit taller than it is wide, so the pyramids themselves will also be taller than they are wide.

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

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

The program we'll write will be called `mario`. And let's allow the user to decide just how tall the pyramids should be by first prompting them for a positive integer between, say, 1 and 8, inclusive.

Here's how the program might work if the user inputs `8` when prompted:

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

Here's how the program might work if the user inputs `4` when prompted:

```
$ ./mario
Height: 4
   #  #
  ##  ##
 ###  ###
####  ####
```

Here's how the program might work if the user inputs `2` when prompted:

```
$ ./mario
Height: 2
 #  #
##  ##
```

And here's how the program might work if the user inputs `1` when prompted:

```
$ ./mario
Height: 1
#  #
```

If the user doesn't, in fact, input a positive integer between 1 and 8, inclusive, when prompted, the program should re-prompt the user until they cooperate:

```
$ ./mario
Height: -1
Height: 0
Height: 42
Height: 50
Height: 4
```

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

Notice that width of the "gap" between adjacent pyramids is equal to the width of two hashes, irrespective of the pyramids' heights.

Open your `mario.c` file to implement this problem as described!

Walkthrough

How to Test Your Code

Does your code work as prescribed when you input

- `-1` (or other negative numbers)?
- `0`?
- `1` through `8`?
- `9` or other positive numbers?
- letters or words?
- no input at all, when you only hit Enter?

You can also execute the below to evaluate the correctness of your code using `check50`. But be sure to compile and test it yourself as well!

```
check50 cs50/problems/2023/x/mario/more
```

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/2023/x/mario/more
```