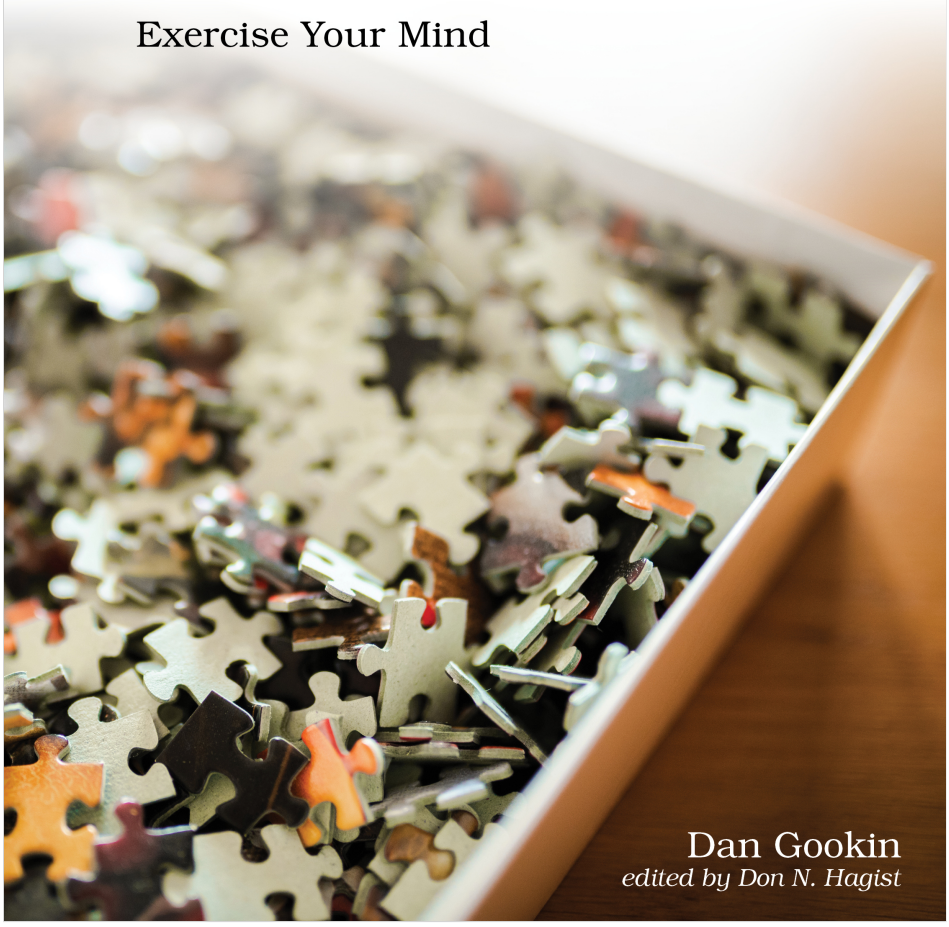
The Pragmatic Programmers

C Brain Teasers

Exercise Your Mind



Dan Gookin
edited by Don N. Hagist

This extract shows the online version of this title, and may contain features (such as hyperlinks and colors) that are not available in the print version.

For more information, or to purchase a paperback or ebook copy, please visit https://www.pragprog.com.

Puzzle 1

Count the Digits

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

int main()
{
        printf("%2.4f\n", M_PI); /* M_PI = 3.14159265358979323846 */

        return(0);
}
```

Guess the Output

Try to guess what the output is before moving to the next page.

The program displays the following output:

```
3.1416
```

Discussion

The defined constant M_PI is declared in the math.h header file. It represents the value of π out to 20 digits. Any floating point value could be specified in the printf() statement, though M_PI represents a known value with oodles of digits after the decimal.

The %2.4f placeholder directs the printf() statement to output a floating point value in a width of six digits: a minimum of two to the left of the decimal and four to the right. Because only one digit is to the left of the decimal, one character is output; a space isn't added. For the right of the decimal, only four digits are output. The value of the last digit is rounded, in this case up.

The Nerdy Details on printf() Formatting

The power in the printf() function comes from the % placeholders, which format the output. While a single conversion specifier directs printf() to output a specific format—%f for floating point, for example—more directions can dwell between the % sign and the conversion specifier to further hone the output. These specifiers define or restrict the output width.

Here are the options available for the %f placeholder:

```
%[significant][.][decimal]f
```

Two optional values, significant and decimal, set output width values left or right of the decimal, which is also optional.

Here is the output when using the %6f placeholder:

```
3.141593
```

Six digits are output after the decimal. When using the %6.f placeholder, this text is output:

```
3
```

Five spaces appear before the three. To see them, prefix the six with a zero: The %06.f placeholder outputs this text:

```
000003
```

You may find this output frustratingly inconsistent—and you're correct. It's difficult to gauge what the output will be unless you've completely immersed yourself in the features and quirks for the %f placeholder's width specifiers. Who has time for that?

A printf() Cheat You Probably Don't Know About

To better understand how many characters the printf() statement outputs, I use a common and little-known trick: the function's return value represents the number of characters in the output. Here is an update to the code:

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

int main()
{
        int n;

        n = printf("%2.4f\n", M_PI);
        printf("(That's %d characters)\n", n);

        return(0);
}
```

The output now looks like this:

```
3.1416
(That's 7 characters)
```

The output counts the newline, which is why 7 appears instead of 6. Also note that the output is rounded, not truncated.

The printf() function's return value, saved in variable n, provides a solid resource for the number of characters output, including the decimal (period). Alas, you can't use this printf() value until after the function returns, which means that you must approximate the number of characters output if that's a concern at runtime. Or you can use snprintf() to store the output in a buffer for further examination.

Further Reading

Detailed list of conversion specifiers and their options

https://cplusplus.com/reference/cstdio/printf/

Placeholder/conversion specifier overview

https://c-for-dummies.com/blog/?p=288