

Writing Pretty Programs

The C programming language gives you a lot of freedom in how you write your programs. There aren't any rules about how lines should be indented, for example, and you can choose to write long statements as one long line or break them up over multiple lines. Two programs that do exactly the same thing can look very different. For instance, here's another way we could have written Program 1.3 (`line.cpp`):

```
#include <stdio.h>
int main () { double x; double y; double slope = 2.0;
double yint = 3.0; int i; x = 0.0;
for ( i=0; i<10; i++ ) {y = slope * x + yint;
printf ( "%lf %lf\n", x, y ); x = x + 1.0;}}
```

I think you'll agree that this is harder to read than the earlier version. Here are four rules for writing pretty programs:

Rule 1: Use Indentation

For making your programs pretty, the most important thing you should remember is that programs are made out of parts that *can hold other parts inside them*. When writing a program we use indentation to make it clear that some parts are inside of others¹⁷

In a C program, curly brackets tell the computer that something is contained inside something else. For example, Program 1.3 consists of a main program that has a `for` loop nested inside it. The statements inside the `for` loop are almost like a little program that the main program runs ten times:

```
#include <stdio.h> { Main Program { A for Loop } The Rest of the Main Program }
```

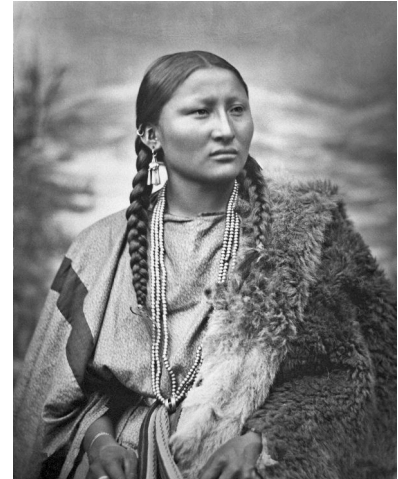
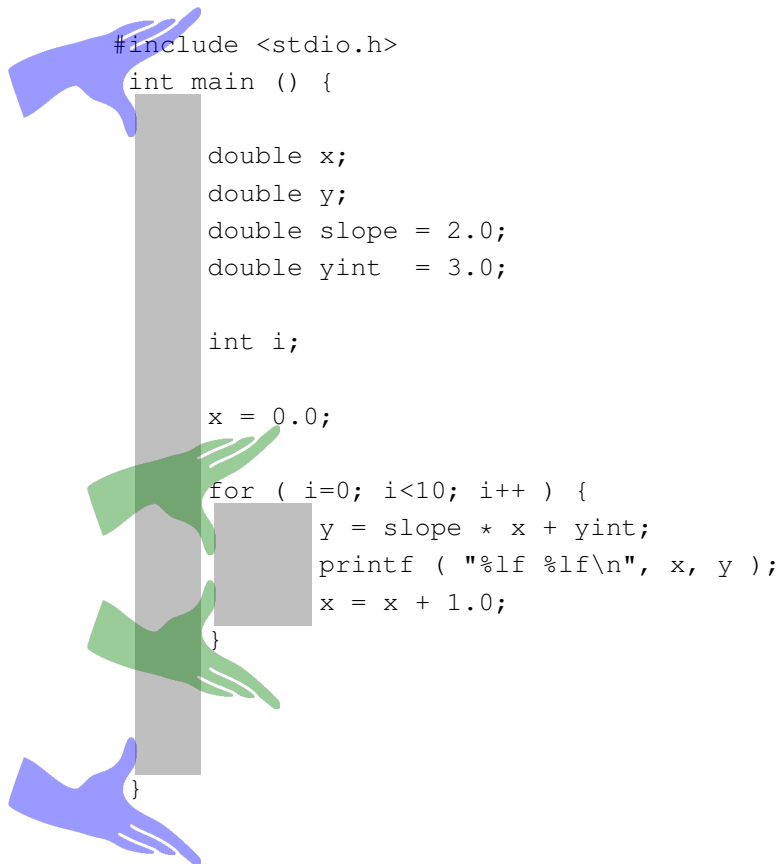


Figure 1.22: The Cheyenne or Arapaho woman named Pretty Nose was a war chief who fought at the Battle of Little Bighorn. Her grandson, Mark Soldier Wolf, was a U.S. Marine who fought in Korea. Pretty Nose was 101 years old when he returned home, and greeted him with a war song.

Source: Wikimedia Commons

¹⁷ You'll find lots of other tips for writing pretty programs here: https://www2.cs.arizona.edu/mccann/indent_c.html.

As you're writing your programs, when you see an opening bracket, {, start indenting. When you see a closing bracket, }, stop indenting. If you're in an already-indented section and you see another opening curly bracket, add more indentation:



```
#include <stdio.h>
int main () {
    double x;
    double y;
    double slope = 2.0;
    double yint = 3.0;

    int i;

    x = 0.0;

    for ( i=0; i<10; i++ ) {
        y = slope * x + yint;
        printf ( "%lf %lf\n", x, y );
        x = x + 1.0;
    }
}
```

It doesn't matter how much space you use for indentation. Some people like to use a tab for each level of indentation. Other people prefer something "shallower", maybe only a couple of spaces. Either way is OK. Just be consistent inside each program you write.

Rule 2: Group Variable Definitions

At this stage in your programming career, I recommend that you define all variables at the top of your programs, right under the "int main ()" statement. Later on you'll learn that C++ allows you to define variables anywhere in a program, and there are advantages to using that ability, but pure C compilers don't allow this. If you want your programs to be as portable as possible, stick to defining variables at the top for now. This also gives you one handy place to look to see your variable definitions.

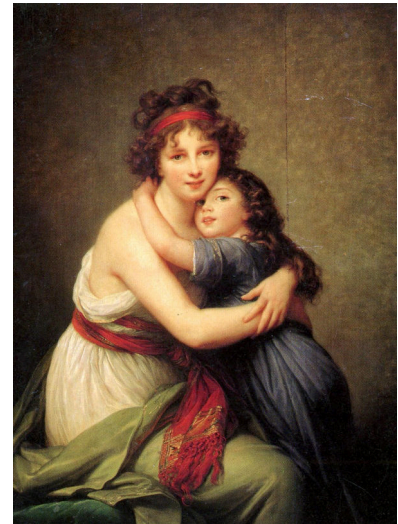


Figure 1.23: Curly brackets are also called "braces", a word that originally meant "arms", as in "embrace". The curly brackets in our programs embrace the statements they enclose. (Louise Élisabeth Vigée Le Brun, Self-portrait with Her Daughter, Julie, c. 1789).

Source: Wikimedia Commons

Rule 3: Use Comments

“Comments” are text that you put into your program to explain what the program does, tell people who wrote the program, give advice about how to run the program, warn about copyrights and patents, or anything else you want to say. Comments are just ignored by the compiler, so they don’t affect the way your program runs. As far as the compiler is concerned, the comment isn’t even there.

The g++ compiler lets you add comments to your program in a couple of ways. Here’s one of them:

```
#include <stdio.h>
int main() {
    // This program was written by Bryan Wright.
    int i;
    for (i = 0 ; i < 10 ; i++) { // Start loop.
        printf("loop number %d\n", i);
    }
}
```

Almost any text between a double slash (//) and the end of the line is a comment. I say “almost” only because this doesn’t work inside quotes, so that:

```
printf("Hello World! // and some other stuff");
```

would print out “Hello World! //and some other stuff”.

The second way to add comments is an older one that will work in any compiler that understands C or C++, and it has the advantage that comments can extend over multiple lines. Look at the following example:

```
#include <stdio.h>
int main() {
    int i;
    /* This program was written by Bryan Wright.
       Copyright 2015.
       All rights reserved.
       Seriously. I'll call my lawyer.
       Don't mess with me. */
    for (i = 0 ; i < 10 ; i++) {
        printf("loop number %d\n", i);
    }
}
```



Figure 1.24: Source: Wikimedia Commons

Any text between `/*` and `*/` is a comment. One caveat is that you can't have a comment inside another comment, so something like:

```
/*Some words /* and some more words */and the end */
```

would cause the compiler to complain, and refuse to compile your program.

Comments are a great way to make your program more readable, but they're also very useful for temporarily removing parts of your program. If the program contains a line that we want to keep, but temporarily disable, we can just put a `/// at the beginning of the line. You'll find that this can help you find problems in your programs. If something isn't working right, you can selectively turn off parts of the program to help you find the problem.`

Rule 4: Look Around You¹⁸

It's possible that someday you'll join a research group or a business where other people have already written lots of programs. When you start contributing your own programs, it's important that your programming style matches the stylistic conventions that are already in use. If everybody uses tabs for indentation, you should probably do so, too. This is especially important if you start modifying programs that other people have written.

¹⁸ Not to be confused with the BBC series of the same name, which you should watch if at all possible:

<https://www.youtube.com/watch?v=gaI6kBVyu0o>



Figure 1.25: Now you're programming with *style!* (Fred Astaire and his sister Adele cutting a rug in 1921.)

Source: Wikimedia Commons