```
  printf("Celsius equivalent is: %.1f\n", celsius);

  return 0;
}
```

After preprocessing, the program will have the following appearance:

```
Blank line
Blank line
Lines brought in from stdio.h
Blank line
Blank line
Blank line
Blank line
Blank line
int main(void)
{
  float fahrenheit, celsius;

  printf("Enter Fahrenheit temperature: ");
  scanf("%f", &fahrenheit);

  celsius = (fahrenheit - 32.0f) * (5.0f / 9.0f);

  printf("Celsius equivalent is: %.1f\n", celsius);

  return 0;
}
```

The preprocessor responded to the `#include` directive by bringing in the contents of `stdio.h`. The preprocessor also removed the `#define` directives and replaced `FREEZING_PT` and `SCALE_FACTOR` wherever they appeared later in the file. Notice that the preprocessor doesn't remove lines containing directives; instead, it simply makes them empty.

As this example shows, the preprocessor does a bit more than just execute directives. In particular, it replaces each comment with a single space character. Some preprocessors go further and remove unnecessary white-space characters, including spaces and tabs at the beginning of indented lines.

In the early days of C, the preprocessor was a separate program that fed its output into the compiler. Nowadays, the preprocessor is often part of the compiler, and some of its output may not necessarily be C code. (For example, including a standard header such as `<stdio.h>` may have the effect of making its functions available to the program without necessarily copying the contents of the header into the program's source code.) Still, it's useful to think of the preprocessor as separate from the compiler. In fact, most C compilers provide a way to view the output of the preprocessor. Some compilers generate preprocessor output when a certain option is specified (GCC will do so when the `-E` option is used). Others come with a separate program that behaves like the integrated preprocessor. Check your compiler's documentation for more information.

A word of caution: The preprocessor has only a limited knowledge of C. As a result, it's quite capable of creating illegal programs as it executes directives. Often the original program looks fine, making errors harder to find. In complicated