

Chapter 7: Getting Input from the User

Any useful computer program has some *input* and some *output*. You've probably already noticed this: in the last assignment, your program had output (i.e., the results of the various calculations), but there wasn't really any input (i.e., you just hardwired certain inputs into the calculations). In this chapter, we'll see that there is an easy way to get input from the user.

The scanf Function

Typically, to get input from a user, you need to follow these steps:

1. Ask the user a question.
2. Read in the user's answer for the question.

Step 1 is quite easy—this is normally accomplished using a `printf` statement. However, the second step is going to require something new—the “scanf” function—as seen in this example:

```
float tempC;

printf ("Enter the temperature, in Celsius.\n");
scanf ("%f", &tempC);
```

In this simple example, we've declared a floating point variable called `tempC`. We want the user to enter a temperature in Celsius, which we will then store in the variable “`tempC`”. We prompted the user with a `printf` statement, telling the user what kind of information we want. The last line—the `scanf` statement, reads in this information.

Typically, the `scanf` function will have two “arguments”—one is a format for the variable that we are going to be reading from the keyboard, and the other is the name of the variable where we want to put the value that we read in. In this case, the “`%f`” is telling us that we should be reading in a floating point number. (Note: that means that the user really does have to enter a floating point number—you'll get fairly spectacular errors of the user types in something like “oh, about 15” or something like this!) The second argument tells us that we want to put the value that the user enters into a variable called `tempC`. Notice that the variable's name is preceded by an ampersand. For the time being, don't worry about *why* the ampersand is there—just know that it always has to be there for a `scanf` function.

Assignment 4: Getting input from the user

Write a program that prompts a user for the current temperature, dewpoint, wind speed, and wind direction. Then, use that information to compute and print the following meteorological parameters:

- Vapor pressure, in millibars.
- Saturation vapor pressure, in millibars.
- Relative humidity, in percent.
- Potential temperature, in Kelvin.
- Wind chill temperature, in Fahrenheit.
- Zonal and meridional wind speed (i.e., u and v winds).

Note number 1: I don't care whether you want the user to enter the temperature and dewpoint in Celsius or Fahrenheit. Just make sure that you *tell* the user how to enter the temperatures, and then, if necessary, perform the conversions. Similarly, I don't care what units you use for the wind speed; just be sure that the *user* knows which units you are using and that any necessary conversions are computed.

Note number 2: The necessary formulas are provided in the "Common Meteorological Calculations" appendix to this book.

Completing the assignment as shown is worth a grade of 85%. You can impress me mildly with a very neat program that is well-documented. Or you could perform more calculations than just these. Or use your imagination—that's what really impresses me!

This assignment is worth 5 points.