Computer Programming Using C
COP 3275 - Summer 2017

Lecture 4: C Fundamentals

{C}

By Ahmed E. Khaled

/* Programming */
Recap to previous lecture!

- Comments
- Variables and Types
- Declaration and assignment
- Printing variables
Comments

• Comments may appear almost anywhere in a program, either on separate lines or on the same lines as other program text.

• Single line comment can also be written in the following way:

        // This is a comment

        printf("Cat");   // Printing Cat

• A comment begins with /* and end with */ as for multiple line.

        /* This is a comment */

        /* first line
         second line
         third line */
Variables

- Variables are used to store information to be referenced and manipulated in a computer program.

- They also provide a way of labeling data with a descriptive name, so our programs can be understood more clearly by the reader and ourselves.

- It is helpful to think of variables as containers that hold information and store data in memory. Such data can then be used throughout your program.
Types

• Every variable must have a *Type*.

• The type of the variable specifies *what kind of data* such variable can *hold*.

• Proper selection of the variable type is very important, since the type affects:
  – How the variable is *stored in memory*
  – What *operations* can be performed on the variable
Types

• A variable of type `int` (short for `integer`) can store a *whole number* such as 0, 1, 392, or –2553.

• A variable of type `float` (short for `floating-point`) can store numbers with *digits after the decimal point*, like 379.125.
Variable Declaration and Assignment

```plaintext
int height;
height = 8;
int heigh = 8;
```
Variable Declaration and Assignment

float profit;

profit = 10.4;

float profit = 10.4;
Printing the Value of a Variable

• To write the message \texttt{Height: } \texttt{h}
  where \texttt{h} is the current value of the \texttt{height}
  variable, we’d use the following call of \texttt{printf}:

\begin{verbatim}
int height = 8;
printf("Height: %d\n", height);
\end{verbatim}
Printing the Value of a Variable

• There’s no limit to the number of variables that can be printed by a single call of `printf`:

```c
int height = 8;
int length = 5;
printf("Height: %d  Length: %d\n", height, length);
```
Printing the Value of a Variable

- \%d works only for int variables; to print a float variable, use \%f instead.

```c
float profit = 2150.485443;
printf("Profit: $%f\n", profit);
```

Output is > Profit: $2150.485443
Printing the Value of a Variable

• By default, %f displays a number with six digits after the decimal point.

• To force %f to display \( p \) digits after the decimal point, put \( .p \) between % and f.

\[
\text{printf("Profit: $%.2f\n", profit);}
\]

Output is > Profit: $2150.48
Program: Computing the Dimensional Weight of a Box

• Shipping companies often charge extra for boxes that are large but very light, basing the fee on volume instead of weight.

• The usual method to compute the “dimensional weight” is to divide the volume by 166.
Program: Computing the Dimensional Weight of a Box

- Division is represented by \( / \) in C, so the obvious way to compute the dimensional weight would be:
  \[
  \text{weight} = \frac{\text{volume}}{166};
  \]

- In C, however, when one integer is divided by another, the answer is “truncated”: all digits after the decimal point are lost.
  - The volume of a 12” \( \times \) 10” \( \times \) 8” box will be 960 cubic inches.
  - Dividing by 166 gives 5 instead of 5.783.
dweight.c

/* Computes the dimensional weight of a 12" x 10" x 8" box */

#include <stdio.h>

int main(void)
{
    int height, length, width, volume, weight;

    height = 8;
    length = 12;
    width = 10;
    volume = height * length * width;
    weight = volume / 166;

    printf("Dimensions: %dx%dx%d\n", length, width, height);

    printf("Volume (cubic inches): %d\n", volume);
    printf("Dimensional weight (pounds): %d\n", weight);

    return 0;
}
Reading Input

• The previous dweight.c program is not very useful. (why?)

• To improve the program, we will need to allow the user to enter the dimensions.
Reading Input

• `scanf` is the C library’s counterpart to `printf`.

• `scanf` requires a *format string* to specify the appearance of the input data, that means `scanf` need to know in what form the input data will be exactly like `printf` that require to know how to display the output data.
Reading input

- Example of using `scanf` to read an `int` value:

```c
int i;
scanf("%d", &i);
/* reads an integer and stores into i */
```
Reading Input

• Reading a float value requires a slightly different call of `scanf`:

```c
float x;
scanf("%f", &x);
```

• "%f" tells `scanf` to look for an input value in float format (the number may contain a decimal point, but doesn’t have to).
Program: Computing the Dimensional Weight of a Box (Revisited)
/* Computes the dimensional weight of a box from input provided by the user */

#include <stdio.h>

int main(void)
{
    int height, length, width, volume, weight;

    printf("Enter height of box: ");
    scanf("%d", &height);
    printf("Enter length of box: ");
    scanf("%d", &length);
    printf("Enter width of box: ");
    scanf("%d", &width);
    volume = height * length * width;
    weight = (volume + 165) / 166;

    printf("Volume (cubic inches): %d\n", volume);
    printf("Dimensional weight (pounds): %d\n", weight);

    return 0;
}
Program: Computing the Dimensional Weight of a Box (Revisited)

• Sample output of program:

Enter height of box: 8
Enter length of box: 12
Enter width of box: 10
Volume (cubic inches): 960
Dimensional weight (pounds): 6

• Note that a prompt shouldn’t end with a new-line character.
4. Write a program that asks the user to enter a dollars-and-cents amount, then displays the amount with 5% tax added:

Enter an amount: 100.00
With tax added: $105.00