Introduction

Programming is the art of solving problems. A computer program is a set of instructions that tell the computer what to do. These instructions can be written in various programming languages, such as C, Python, Java, etc. Each language has its own syntax and structure, but the fundamental concepts are similar:

1. **Variables** - hold values that can be changed.
2. **Data Types** - define the kind of data a variable can hold (int, float, string, etc.).
3. **Control Structures** - allow for conditional logic and looping.
4. **Functions** - reusable blocks of code that perform a specific task.
5. **Classes** - define a blueprint for objects.

These concepts are foundational in programming and are essential for solving any problem. In this chapter, we will explore how to use good programming practices to create a simple program.
Purposa Whid you the program what does it do?

An add program or modify the program can come to you for information.

Author: Poor Man's Interface

If you've used a lot of trouble to learn this program, I'm credit for

The best common comment is to the name of the program.

POOR MAN'S INTERFACE

POOR MAN'S INTERFACE

POOR MAN'S INTERFACE
Common Coding Practices

Inserting Comments—The Easy Way

In the previous section, we discussed how to use three different comment mechanisms. The following code shows a simple example of using each of these mechanisms:

```plaintext
// A single-line comment

/*
 * A multi-line comment
 *
 */

/**
 * A trailing comment
 *
 */
```

These two lines define a variable and function, so that they can be

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuing comments after

ابة

continuating
**Chapter 3: Scope**

In order to make programs easier to understand, programmers often use a coding convention known as **Scope and Code Format**. This convention helps to organize code in a way that makes it easier to read and understand. Here are some key points to consider:

1. **Variable Scope**:
   - **Local Scope**: Variables declared within a function or block are local to that function or block. They are not accessible outside the function or block.
   - **Global Scope**: Variables declared outside all functions are global. They can be accessed from anywhere in the program.

2. **Code Format**:
   - Use consistent indentation to improve readability.
   - Use clear naming conventions for variables and functions.
   - Avoid using abbreviations or jargon that may not be universally understood.
   - Keep lines of code short and readable.

Understanding and applying these conventions helps to make your code more understandable and maintainable.
Chapter 3 - Page 46

Complexity vs. Simplicity

Complexity: The single function should not be longer than two pages.

Simplicity: Your program should be simple. Some general rules of thumb are:

- Don't repeat yourself. If you find yourself writing the same code in multiple places, extract it into a function.
- Avoid nesting too deep. Each level of nesting adds complexity to the code.
- Ensure that each function is independent and has a single responsibility.
- Use readable variable names that accurately describe their purpose.
- Include comments that explain the purpose and logic of the code.

A better version could be:

```c
#include <stdio.h>

int main() {
    int x = 5;
    int y = 10;
    printf("The sum is %d\n", x + y);
    return 0;
}
```

In this case, the comment at the bottom of the page is removed, shortening the code snippet corresponding to the previous section and improving the code's readability. The function's purpose is clearly stated in a comment above it, making the code more self-explanatory.


```c
#include <stdio.h>

int main() {
    int x = 5;
    int y = 10;
    printf("The sum is %d\n", x + y);
    return 0;
}
```
Elements of a Program

- Variables
- Constants
- Procedures
- Functions
- Expressions
- Statements
- Decisions
- Loops
- Arrays
- Structures
- Classes
- Objects

Basic Declaration

Expressions

Procedures

Functions

Statements

Decisions

Loops

Arrays

Structures

Classes

Objects

In this chapter
Simple Expressions

In a C program, expressions are used to specify simple computations. These are
instructions that can do more than just print strings—they can also perform other
operations as well.

### Simple Expressions

To understand expressions, you'll need to know a number of rules. Correctly,

1. Expressions are evaluated from left to right, with the
2. Operators that have the same precedence are evaluated from left to
3. Operators with higher precedence are evaluated before those with
4. Parentheses can be used to override the order of operations.
5. The result of an expression is the value that is returned.

#### Example

```
int a = 5, b = 3;

int c = a + b;
```

In this example, the expression `a + b` is evaluated first, giving a result of 8. Then, the result is assigned to `c`.
Variables and Storage

We need to store the results of our calculations.

"We just go back and forth.
"Do you want me to carry picks in the building site, sir?"

The word "storage" refers to the process of storing and retrieving data. In the context of programming, storage is used to hold values that are computed and used later in the program.

Although we calculate the answer, we don't do anything with it (this program is a simple example).
Two other expressions:

In Example 4.2, we use the variable term to store an integer value that is used
as an initialization value. The question mark indicates the value of t

Assignment Statements

Assignment statements are used to give a variable a value. For example:

```
| t = 12; |
| t = t + 1; |
| t = 12 * t; |
| t = t * (1 + 2); |
```

In an assignment statement, the value is assigned to the variable named on the left side of the equal sign.

Variables are given a value through the use of assignment statements.

Assignment Statements

Why does this Java code output a result when it runs on a PC?

```
int t = 12;
```

The following will work on a UNIX machine, but will fail on a PC:

```
int t = 12;
```

Another example:

```
int t = 12;
int v = t + 1;
int x = t * (1 + 2);
```

Chapter 3: Modular Programming

For more information on modular programming, see the references at the back of the book.

```
int t = 12;
int v = t + 1;
int x = t * (1 + 2);
```

Chapter 4: C Preprocessor

C Preprocessor syntax is provided in the documentation that accompanies this book.

```
/* These #include directives are executed at compile time.
   They are not part of the source code.
   Links are used to incorporate additional source files into the main program.
   Links are similar to #include directives, but they are not executed at compile
   time. Links are executed at link time, when the executable program is created.
*/
```

Integers

```
int x = 12;
```

Integer numbers have no fractional parts or decimal points.

```
int x = 12;
```
Floating Point

For example, 1.234 is a shorthand version of 1.234 x 10^0.

Although the number may include an exponent specification of the form

\[ \times 10^n \]

where \( n \) is 0, 1, 2, 3, or 4, the extra 0 decimal indication that you are using a floating-point number does not change the number. Although you could omit digits before the decimal point and specify a number.

0.5, 0.1, and 8,888 digits are floating-point numbers. For example, 0.5 is a floating-point number whereas 5 is an integer. Floating-point numbers include 3.14159, 0.00000000000000000000001 and 1.26 are all floating-point numbers. The numbers 5.3, 8.3, and 1.26 are all floating-point numbers. The numbers 5.3, 8.3, and 1.26 are all floating-point numbers.

Because of the way they are stored internally, real numbers are also known...

### Example Function

```c
#include <stdio.h>

float calculate(float a, float b) {
    float result;
    result = a + b;
    return result;
}
```

### Print Function

The Print function can be used to print the results. It is called the

```c
#include <stdio.h>

void print(float result) {
    printf("The result is: %.2f\n", result);
}
```

### Example 2: Function (continued)

The library function printf can be used to print the results. It is called the

```c
#include <stdio.h>

void print(float result) {
    printf("The result is: %.2f\n", result);
}
```
For example:

Conversion is performed when a floating-point number is assigned to an integer.

C allows the assignment of an integer expression to a floating-point variable. C

<table>
<thead>
<tr>
<th>Floating Point</th>
<th>Integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00 / 0.00</td>
<td>1</td>
</tr>
<tr>
<td>1.00 / 0.01</td>
<td>1</td>
</tr>
<tr>
<td>1.00 / 0.20</td>
<td>5</td>
</tr>
<tr>
<td>1.00 / 0.21</td>
<td>1 + 2</td>
</tr>
</tbody>
</table>

Table 6-2: Expression Examples

The division operation is special. There is a vast difference between an integer

Floating Point Versus Integer Divide

To print the expression 1.0/0.2, we use this statement:

printf("%.6f\n", e);  // e is the expression to be evaluated.

When a floating-point number is assigned to an integer, the conversion is needed.

Return value is the result of expression 1.0/0.2. The result is truncated (any

decimal part is discarded). So the value of 1.0/0.1 is 1.0 and 1.0/0.1 is also a floating-point

number.

The form of a floating-point expression is:

Example 6-1: Why is the result of example 4-4.0? What must be done in this

example? Let's take the result of expression 1.0/0.2. The result is non-integer.

Return value is the result of expression 1.0/0.2. The result is truncated (any
decimal part is discarded). So the value of 1.0/0.1 is 1.0 and 1.0/0.1 is also a floating-point

number.

The form of a floating-point expression is:

Example 6-2: Why is the result of example 4-4.0? What must be done in this

example? Let's take the result of expression 1.0/0.2. The result is non-integer.

Return value is the result of expression 1.0/0.2. The result is truncated (any
decimal part is discarded). So the value of 1.0/0.1 is 1.0 and 1.0/0.1 is also a floating-point

number.

The form of a floating-point expression is:
4.3 Special Characters

A ASCII Table contains a table of ASCII character codes. Each character in the ASCII table corresponds to a number that can be used to represent the character in a computer program. The characters are divided into categories such as punctuation, digits, letters, and control characters.

When converted, the program physics:

```c

class Program
{
    static void Main()
    {
        char c = 'A';
        char c2 = (char)167;
        Console.WriteLine(c); // Output: A
        Console.WriteLine(c2); // Output: ☛
    }
}
```

- When converted, the program physics:
  - `char c = 'A';` outputs `'A'`
  - `char c2 = (char)167;` outputs ☛

ANSWERS

When executed, this program prints:

```
A ☛
```

The result of the program is:

```
A ☛
```
Arrays

In this chapter,

Programming Exercises

Chapter 4: Basic Declarations and Expressions