Chapter 1

Introduction to C++ Programming

Algorithms

- Algorithm
  - A sequence of precise instructions that leads to a solution
- Program
  - An algorithm expressed in a language the computer can understand

Example

An Algorithm

Algorithm that determines how many times a name occurs in a list of names:

1. Get the list of names.
2. Get the name being checked.
3. Set a counter to zero.
4. Do the following for each name on the list:
   Compare the name on the list to the name being checked, and if the names are the same, then add one to the counter.
5. Announce that the answer is the number indicated by the counter.

Program Design

- Programming is a creative process
  - No complete set of rules for creating a program

- Program Design Process
  - Problem Solving Phase
    - Result is an algorithm that solves the problem
  - Implementation Phase
    - Result is the algorithm translated into a programming language

Problem Solving Phase

- Be certain the task is completely specified
  - What is the input?
  - What information is in the output?
  - How is the output organized?

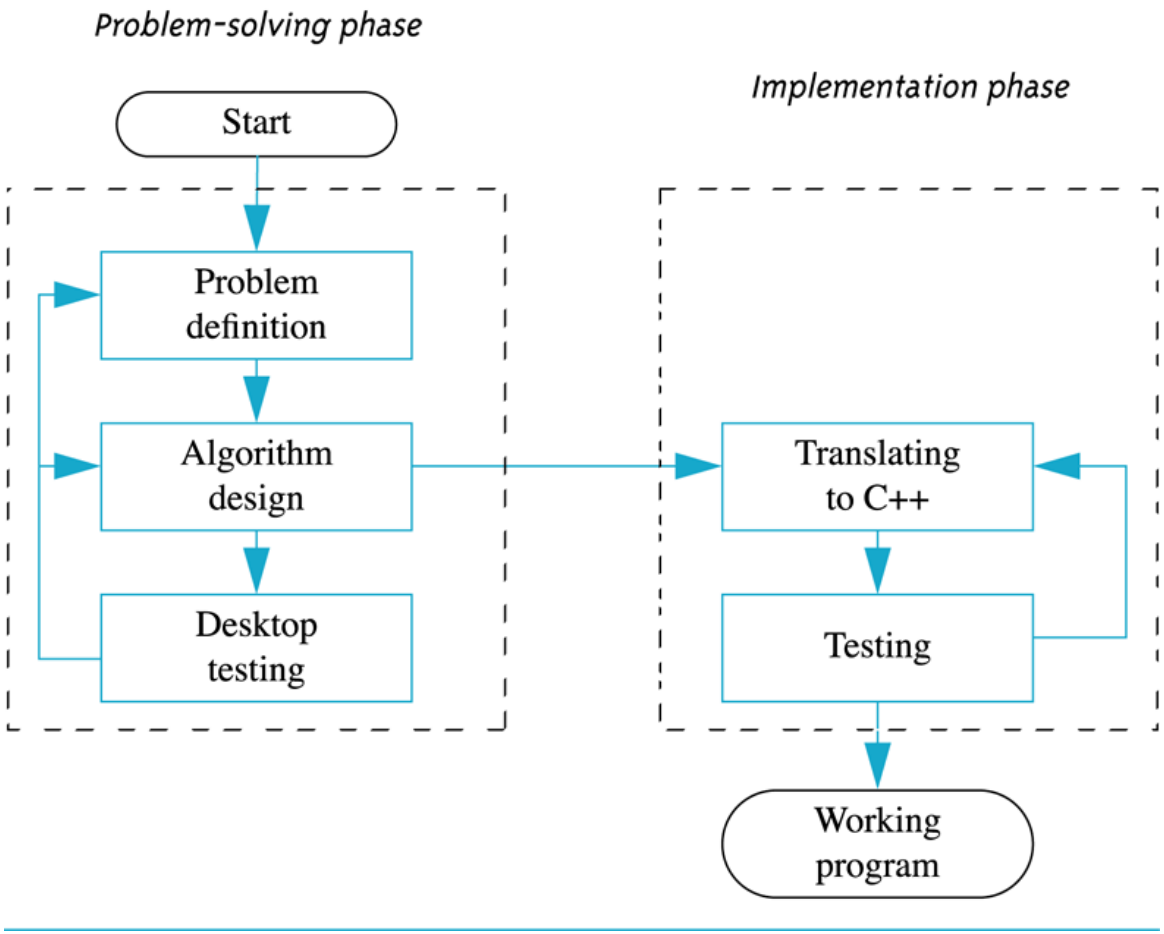
- Develop the algorithm before implementation
  - Experience shows this saves time in getting your program to run.
  - Test the algorithm for correctness

Implementation Phase

- Translate the algorithm into a programming language
  - Easier as you gain experience with the language
- Compile the source code
  - Locates errors in using the programming language
- Run the program on sample data
  - Verify correctness of results
- Results may require modification of the algorithm and program

Problem Solving & Programming

Program Design Process



Object Oriented Programming

- Abbreviated OOP
- Used for many modern programs
- Program is viewed as interacting objects
  - Each object contains algorithms to describe its behavior
  - Program design phase involves designing objects and their algorithms

OOP Characteristics

- Encapsulation
  - Information hiding
  - Objects contain their own data and algorithms
- Inheritance
  - Writing reusable code
  - Objects can inherit characteristics from other objects
- Polymorphism
  - A single name can have multiple meanings depending on its context

Software Life Cycle

- Analysis and specification of the task (problem definition)
- Design of the software (object and algorithm design)
- Implementation (coding)
- Maintenance and evolution of the system
- Obsolescence

Exercise

- Can you…
  - Describe the first step to take when creating a program?
  - List the two main phases of the program design process?
  - Explain the importance of the problem-solving phase?
  - List the steps in the software life cycle?

Introduction to C++

Introduction to C++

- Where did C++ come from?
  - Derived from the C language
  - C was derived from the B language
  - B was derived from the BCPL language
- Why the '++'?
  - ++ is an operator in C++ and results in a cute pun

C++ History

- C developed by Dennis Ritchie at AT&T Bell Labs in the 1970s.
  - Used to maintain UNIX systems
  - Many commercial applications written in c
- C++ developed by Bjarne Stroustrup at AT&T Bell Labs in the 1980s.
  - Overcame several shortcomings of C
  - Incorporated object oriented programming
  - C remains a subset of C++

A Sample C++ Program

- A simple C++ program begins this way

```
#include <iostream>
using namespace std;

int main()
{
```

- And ends this way

```
        return 0;
}
```

A Sample C++ Program

```
#include <iostream>
using namespace std;

int main( )
{
    int number_of_pods, peas_per_pod, total_peas;

    cout << "Press return after entering a number.\n";
    cout << "Enter the number of pods:\n";
    cin >> number_of_pods;
    cout << "Enter the number of peas in a pod:\n";
    cin >> peas_per_pod;

    total_peas = number_of_pods * peas_per_pod;

    cout << "If you have ";
    cout << number_of_pods;
    cout << " pea pods\n";
    cout << "and ";
    cout << peas_per_pod;
    cout << " peas in each pod, then\n";
    cout << "you have ";
    cout << total_peas;
    cout << " peas in all the pods.\n";

    return 0;
}
```

Sample Dialogue

```
Press return after entering a number.
Enter the number of pods:
10
Enter the number of peas in a pod:
9
If you have 10 pea pods
and 9 peas in each pod, then
you have 90 peas in all the pods.
```

Explanation of code (1/5)

- Variable declaration line

  int number_of_pods, peas_per_pod, total_peas;

  - Identifies names of three variables to name numbers
  - int means that the variables represent integers

Explanation of code (2/5)

- Program statement

  cout << “Press return after entering a number.\n”;

  - cout (see-out) used for output to the monitor
  - “<<“ inserts “Press…a number.\n” in the data bound for the monitor
  - Think of cout as a name for the monitor
    - “<<“ points to where the data is to end up
  - ‘\n’ causes a new line to be started on the monitor

Explanation of code (3/5)

- Program statement

  cin >> number_of_pods;

  - cin (see-in) used for input from the keyboard
  - “>>” extracts data from the keyboard
  - Think of cin as a name for the keyboard
    - “>>” points from the keyboard to a variable where the data is stored

Explanation of code (4/5)

- Program statement

```
total_peas = number_of_pods * peas_per_pod;
```

  - Performs a computation
  - '*' is used for multiplication
  - '=' causes total_peas to get a new value based on the calculation shown on the right of the equal sign

Explanation of code (5/5)

- Program statement

```
cout << number_of_pods;
```

  - Sends the value of variable number_of_pods to the monitor

Program Layout (1/3)

- Compiler accepts almost any pattern of line breaks and indentation
- Programmers format programs so they are easy to read
  - Place opening brace ‘{‘ and closing brace ‘}’ on a line by themselves
  - Indent statements
  - Use only one statement per line

Program Layout (2/3)

- Variables are declared before they are used
  - Typically variables are declared at the beginning of the program
  - Statements (not always lines) end with a semi-colon

- Include Directives
  #include <iostream>
  - Tells compiler where to find information about items used in the program
  - iostream is a library containing definitions of cin and cout

Program Layout (3/3)

- using namespace std;
  - Tells the compiler to use names in iostream in a “standard” way

- To begin the main function of the program

```
int main()
{
```

- To end the main function

```
        return 0;
}
```

  - Main function ends with a return statement

Running a C++ Program

- C++ source code is written with a text editor
- The compiler on your system converts source code to object code.
- The linker combines all the object code into an executable program.

C++11

- C++11 (formerly known as C++0x) is the most recent version of the standard of the C++ programming language.
  - Approved on August 12, 2011 by the International Organization for Standardization.
- C++11 language features are not supported by older compilers
- Check the documentation with your compiler to determine if special steps are needed to compile C++11 programs
  - e.g. with g++, use extra flags of –std=c++11

Simple C++ program

Testing Your C++ Setup

```
#include <iostream>
using namespace std;

int main( )
{
    cout << "Testing 1, 2, 3\n";
    return 0;
}
```

If you cannot compile and run this program, then see the programming tip entitled "Getting Your Program to Run." It suggests some things you might do to get your C++ programs to run on your particular computer setup.

Sample Dialogue

```
Testing 1, 2, 3
```

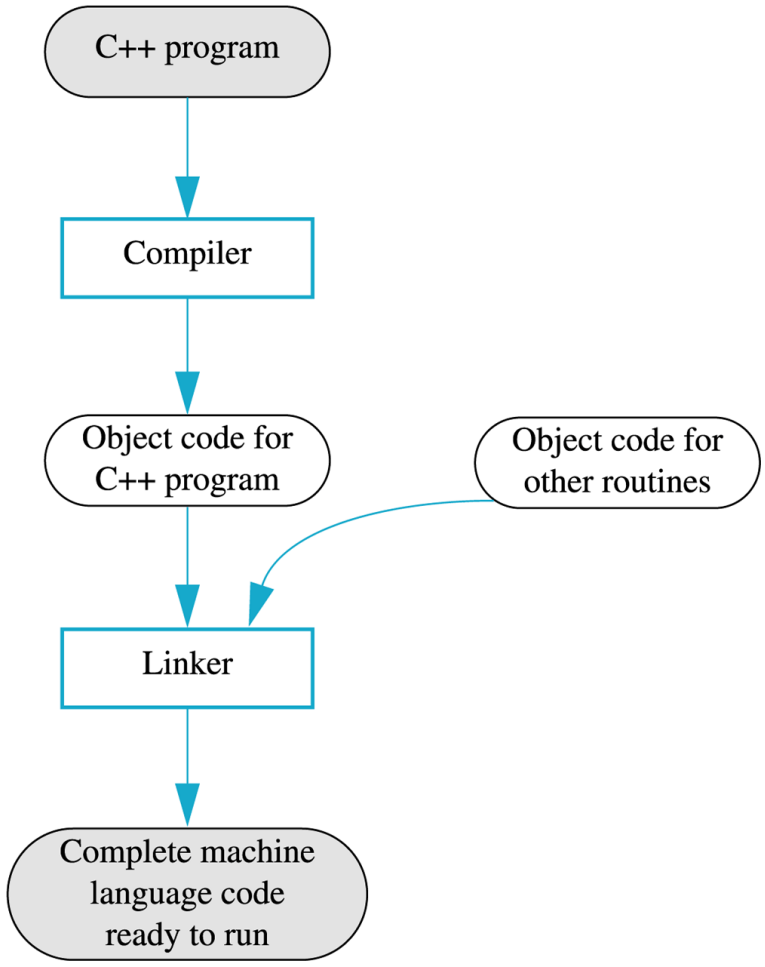
Simple C++ program

Layout of a Simple C++ Program

```
#include <iostream>
using namespace std;

int main( )
{
    Variable_Declarations

    Statement_1
    Statement_2
      ...
    Statement_Last

    return 0;
}
```

Running a C++ program

Preparing a C++ Program for Running



Exercise

- Can you…
  - Describe the output of this line?

    cout << “C++ is easy to understand.”;

  - Explain what this line does?

    cin >> peas_per_pod;

  - Explain this? #include <iostream>

Testing and Debugging

Testing and Debugging

- Bug
  - A mistake in a program
- Debugging
  - Eliminating mistakes in programs
  - Term used when a moth caused a failed relay on the Harvard Mark 1 computer. Grace Hopper and other programmers taped the moth in logbook stating:

    "First actual case of a bug being found."

Program Errors

- Syntax errors
  - Violation of the grammar rules of the language
  - Discovered by the compiler
    - Error messages may not always show correct location of errors
- Run-time errors
  - Error conditions detected by the computer at run-time
- Logic errors
  - Errors in the program's algorithm
  - Most difficult to diagnose
  - Computer does not recognize an error

Exercise

- Can you…
  - Describe the three kinds of program errors?
  - Tell what kind of errors the compiler catches?
  - What kind of error is produced if you forget a punctuation symbol such as a semi-colon?
  - Tell what type of error is produced when a program runs but produces incorrect results?

Chapter 1 -- End