Overview of Programming and Problem Solving Walkthrough with Figures from

"Programming and Problems Solving with C++" - Dale and Weems required textbook

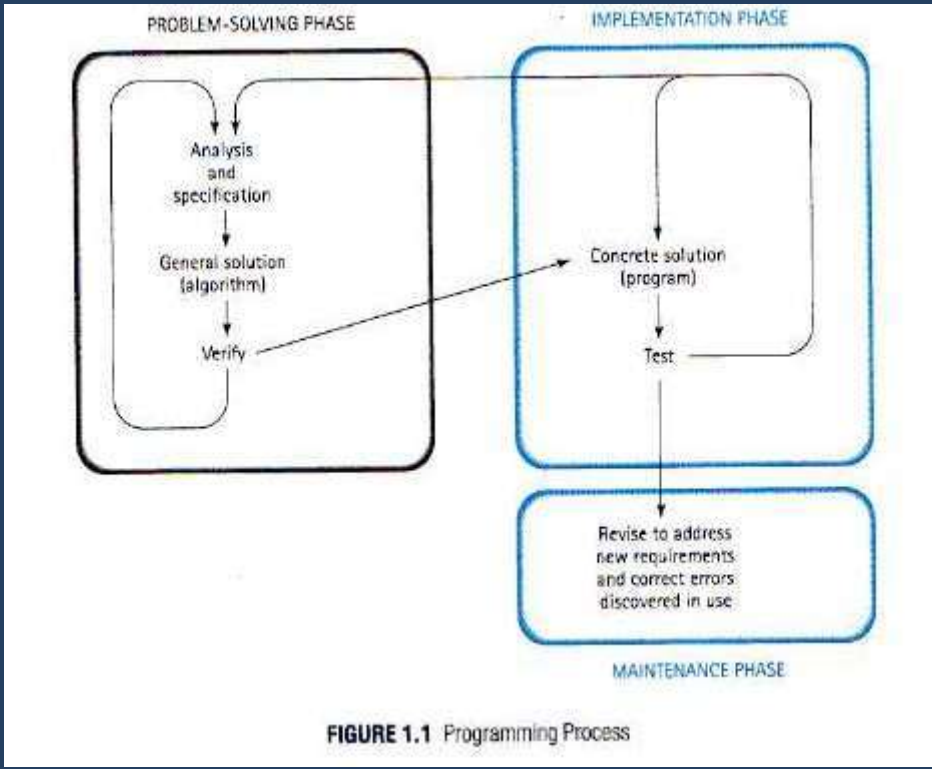


FIGURE 1.1 Programming Process

The Programming Process

When you start your car, you follow a step-by-step procedure. The algorithm might look something like this:

1. Insert the key.
2. Depress the brake pedal.
3. Make sure the transmission is in Park (or Neutral).
4. Turn the key to the start position.
5. If the engine starts within six seconds, release the key to the ignition position.
6. If the engine doesn't start in six seconds, release the key and gas pedal, wait ten seconds, and repeat Steps 3 through 6, but not more than five times.
7. If the car doesn't start, call the garage.

The most fundamental concept of computer science is that of understanding, writing and using algorithms.

Algorithm: Is a set of steps that defines how a task is performed. More precisely, an algorithm is an ordered set of unambiguous, executable steps that define a terminating activity.

The study of algorithms began as a subject in mathematics long before today's computers. A major goal of mathematicians was to find a single set of directions that described how any

problem of a particular type could be solved. The long division algorithm for finding the quotient of two multiple-digit numbers is one such example and another is the Euclidean algorithm, discovered by the ancient Greek mathematician Euclid, for find the greatest common divisor of two positive numbers.

Description: This algorithm assumes that its input consists of two positive integers and proceeds to compute the greatest common divisor of these two values.

Procedure:

Step 1. Assign M and N the value of the larger and smaller of the two input values, respectively.

Step 2. Divide M by N, and call the remainder R.

Step 3. If R is not 0, then assign M the value of N, assign N the value of R, and return to step 2; otherwise, the greatest common divisor is the value currently assigned to N.

The Euclidean algorithm for finding the greatest common divisor of two positive integers

An algorithm for perform magic tricks

Effect: the performer places some cards from a normal deck of playing cards face down on a table and mixes them thoroughly while spreading them out on the table. Then, as the audience requests either red or black cards, the performer turns over cards of the requested color.

Secret and Patter:

Step 1. From a normal deck of cards, select ten red cards and ten black cards. Deal these cards face up in two piles on the table according to color.

Step 2. Announce that you have selected some red cards and some black cards.

Step 3. Pick up the red cards. Under the pretense of aligning them into a small deck, hold them face down in your left hand and, with the thumb and first finger of your right hand, pull back on each end of the deck so that each card is given a slightly backward curve. Then place the deck of red cards face down on the table as you say, "Here are the red cards in this stack."

Step 4. Pick up the black cards. In a manner similar to that in step 3, give these cards a slight forward curve. Then return these cards to the table in a face-down deck as you say, "And here are the black cards in this stack."

Step 5. Immediately after returning the black cards to the table, use both hands to mix the red and black cards (still face down) as you spread them out on the tabletop. Explain that you are thoroughly mixing the cards.

Step 6. As long as there are face-down cards on the table, repeatedly execute the following steps:

6.1. Ask the audience to request either a red card or a black card.

6.2. If the color requested is red and there is a face-down card with a concave appearance, turn over such a card while saying, "Here is a red card."

6.3. If the color requested is black and there is a face-down card with a convex appearance, turn over such a card while saying, "Here is a black card."

6.4. Otherwise, state that there are no more cards of the requested color and turn over the remaining cards to prove your claim.

An algorithm for a magic trick

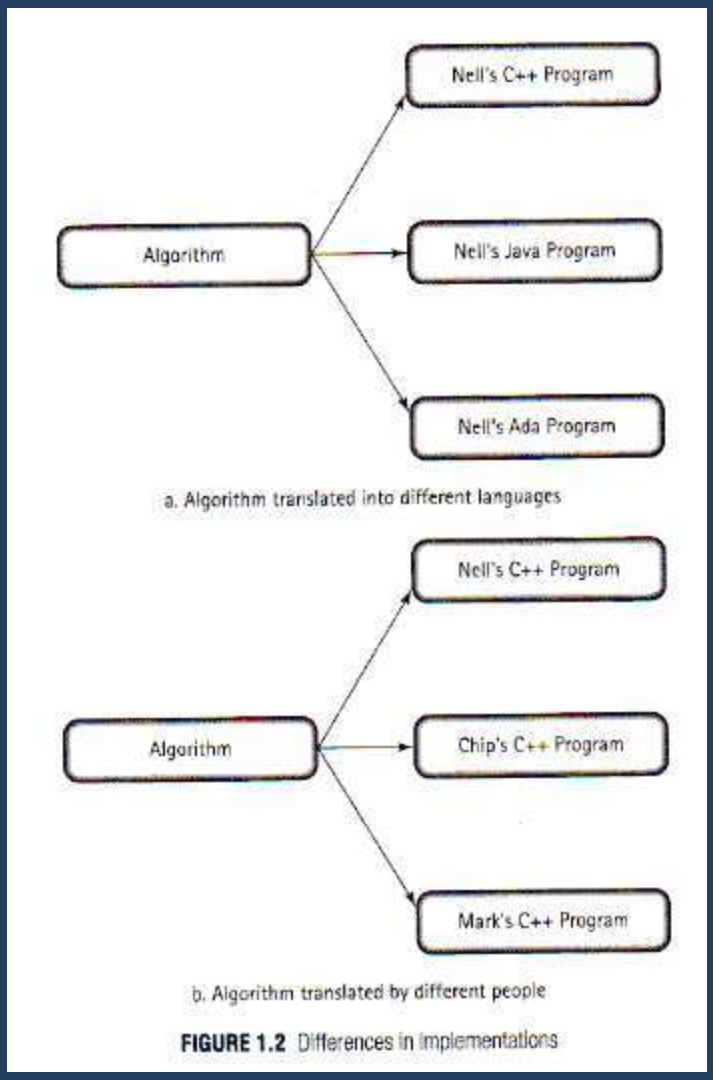


FIGURE 1.2 Differences in Implementations

Differences in implementation of algorithms (see pg. 5)

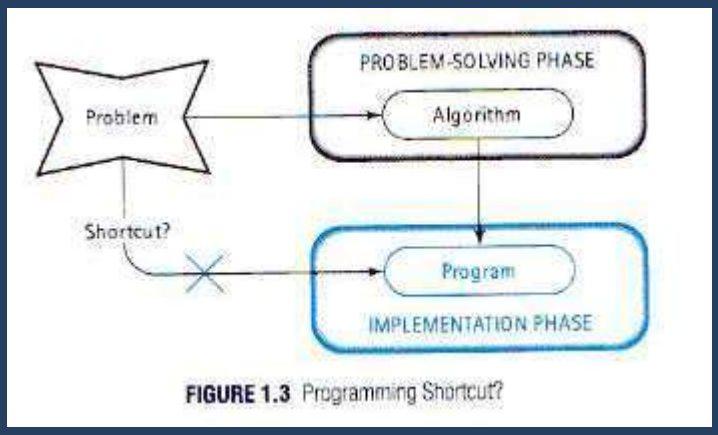


FIGURE 1.3 Programming Shortcut?

Something not to do!

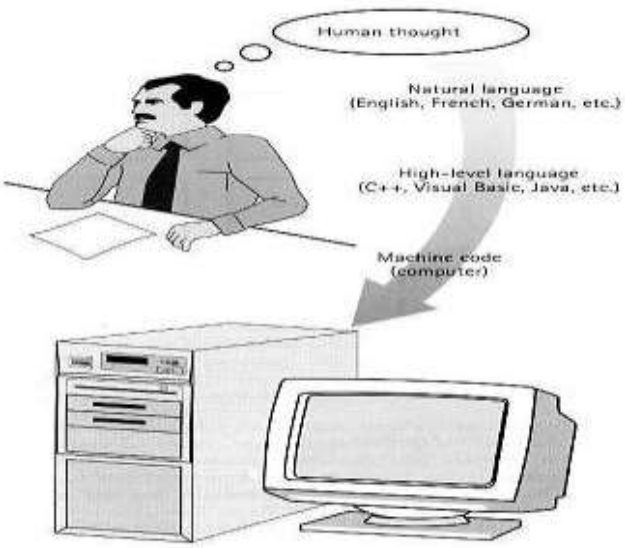


Figure 1.4 *Levels of abstraction*

Levels of abstraction

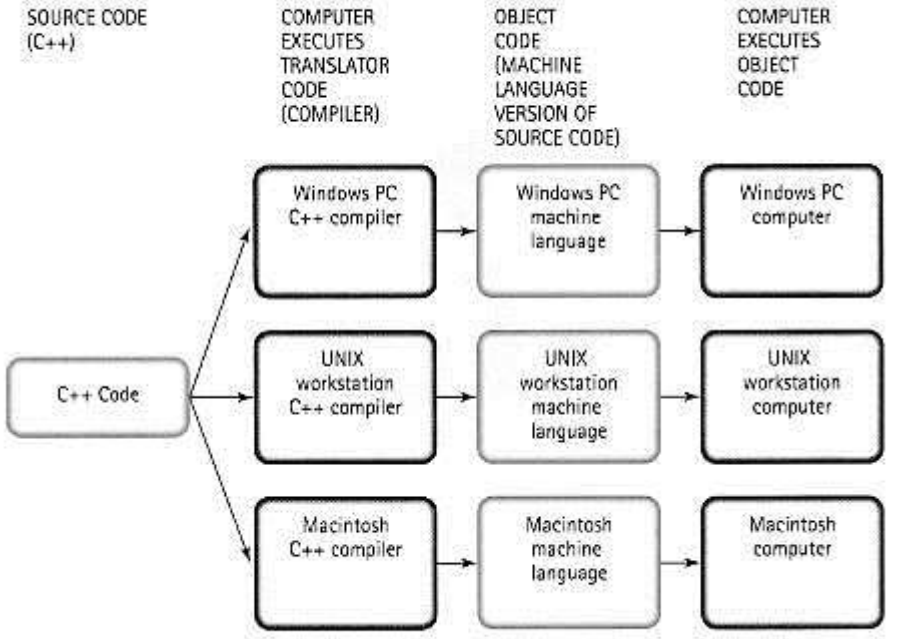


Figure 1.5 *High-level programming languages allow programs to be compiled on different systems.*

Programming languages and Operating System platforms

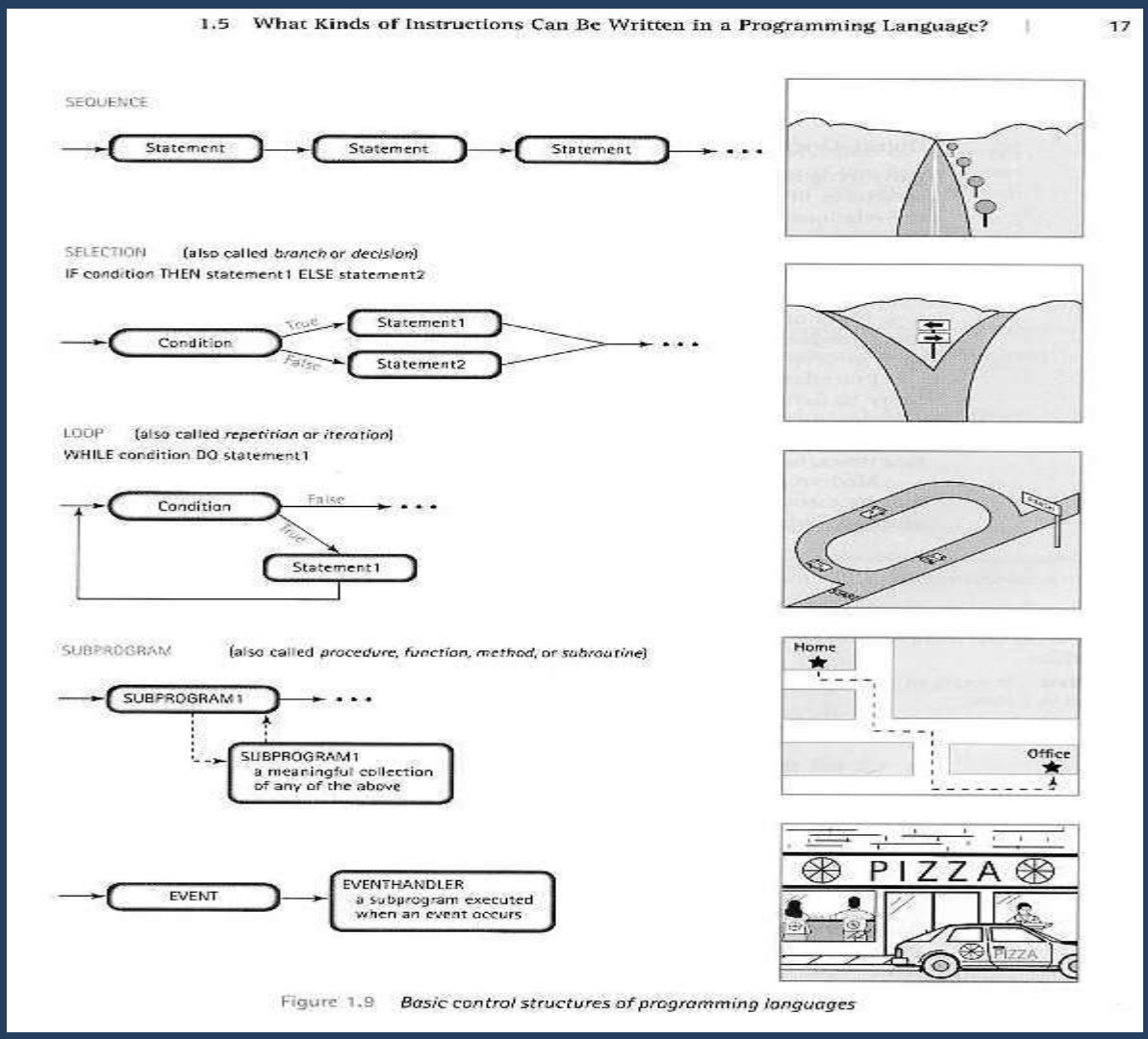


Figure 1.9 *Basic control structures of programming languages*

A visual look at control structures – you will be writing "logic routines" that address the various programming challenges that will appear in the upcoming assignments and projects.

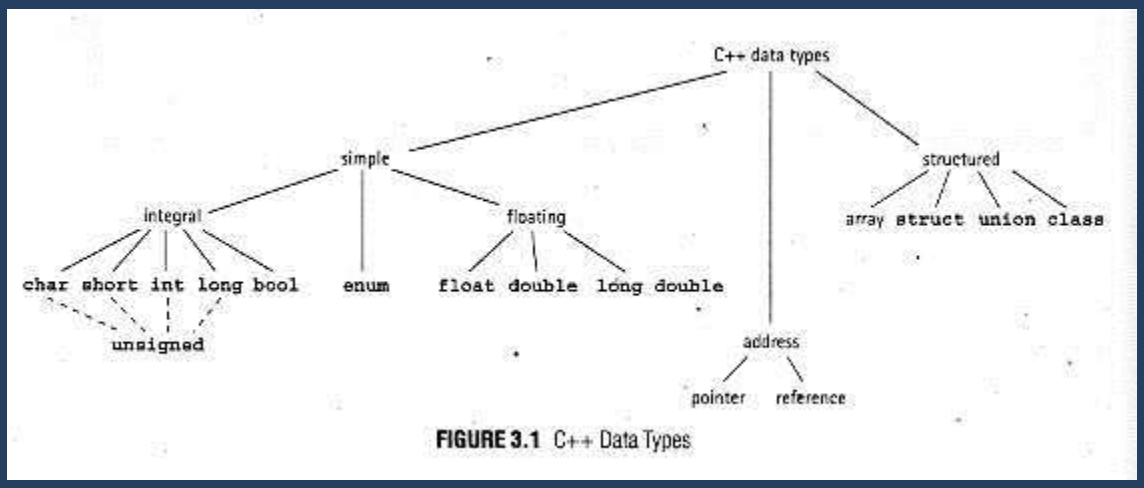


FIGURE 3.1 C++ Data Types

A Visual look at C++ Data Types (pre-defined or primitive) – You will have a basic introduction to data structures and work with both simple and structured types

Computer: Typical components, services and processes in a programmable electronic device

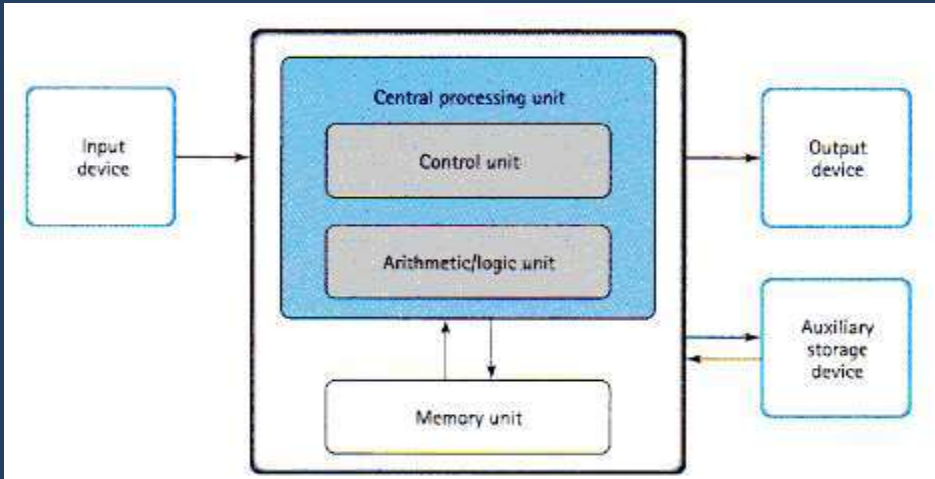


FIGURE 1.8 Basic Components of a Computer

Program code algorithms will interact with all of the above components

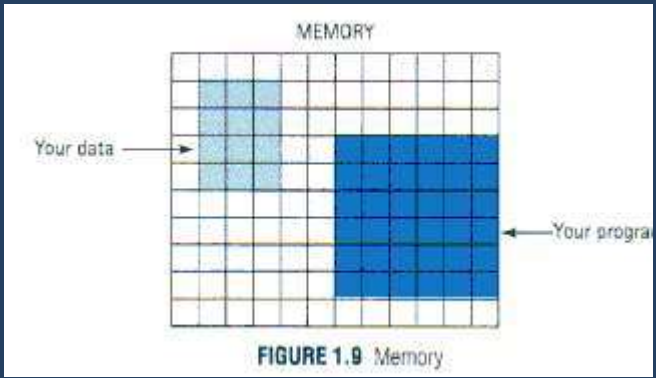


FIGURE 1.9 Memory

Program code and data is stored and allocated computer memory resources

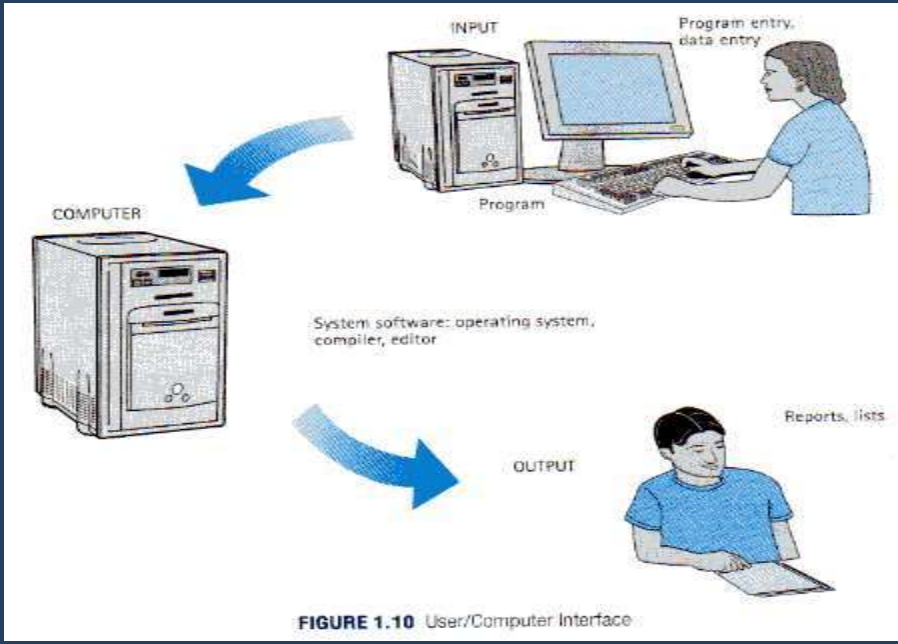


FIGURE 1.10 User/Computer Interface

Program code and data is stored and allocated computer memory resources

Problem Solving Techniques

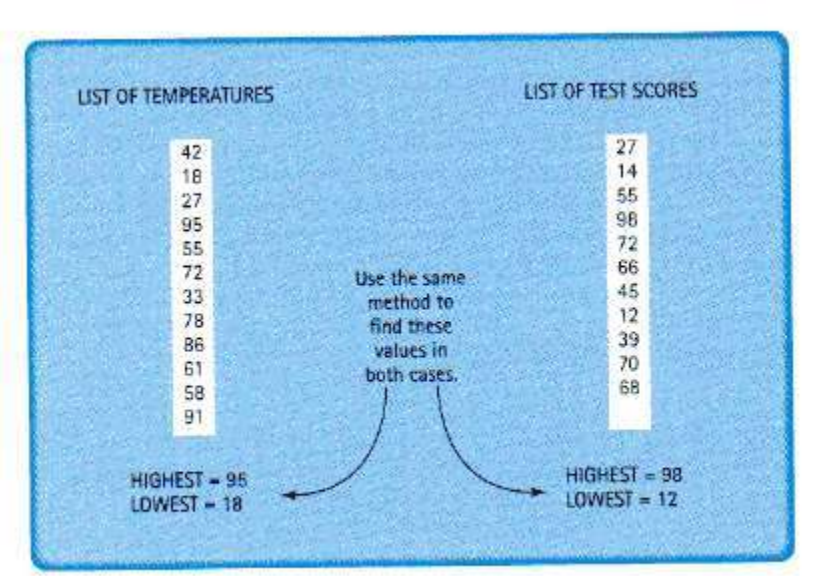
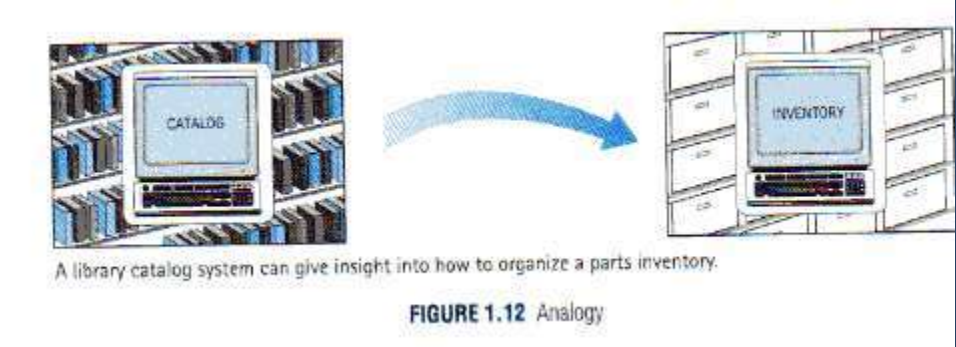


FIGURE 1.11 Look for Things That Are Familiar



A library catalog system can give insight into how to organize a parts inventory.

FIGURE 1.12 Analogy

Start: Boston Goal: Austin	Means: *Fly*, walk, hitchhike, bike, drive, sail, bus
Start: Boston Goal: Austin	Revised Means: Fly to Chicago and then Austin; *fly to Newark and then Austin*; fly to Atlanta and then Austin
Start: Boston Intermediate Goal: Newark Goal: Austin	Means to Intermediate Goal: *Commuter flight*, walk, hitchhike, bike, drive, sail, bus
Solution: Take commuter flight to Newark and then catch cheap flight to Austin	

FIGURE 1.13 Means-Ends Analysis

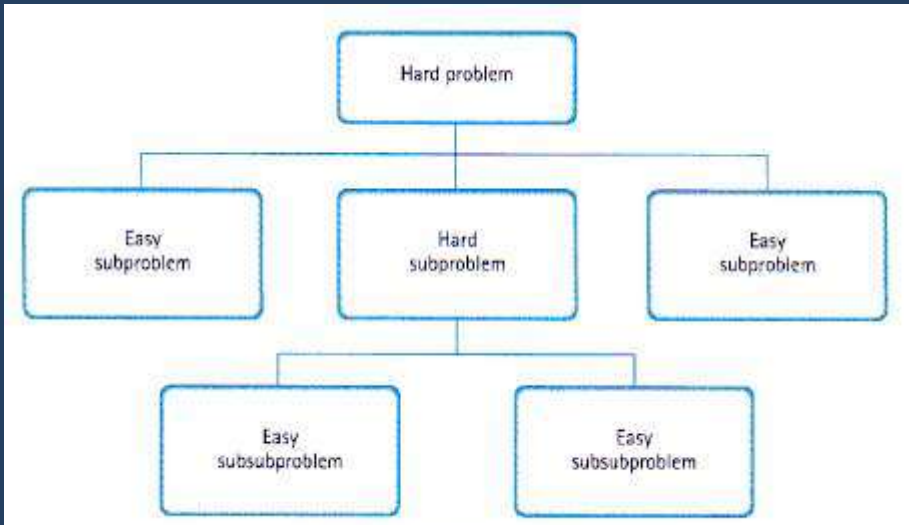


FIGURE 1.14 Divide and Conquer

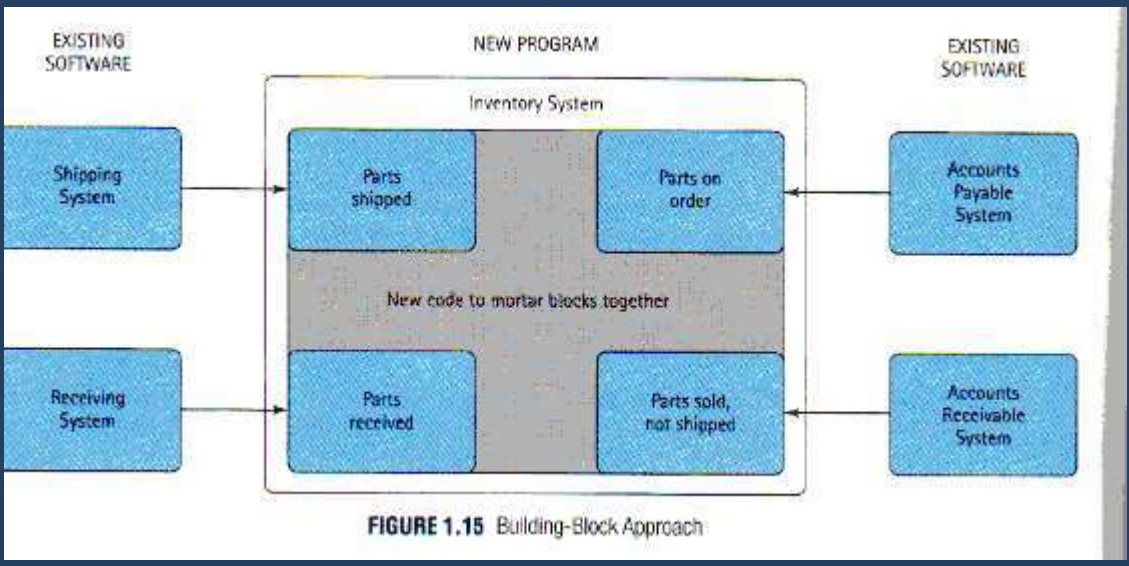


FIGURE 1.15 Building-Block Approach

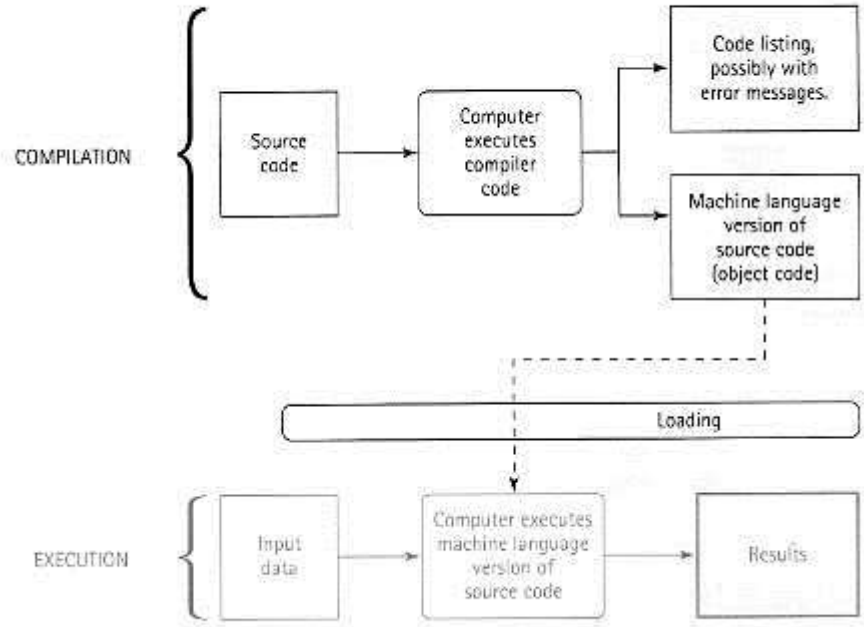


Figure 1.7 *Compilation/Execution*

The Compilation, Debugging, Execution Process – a place where you will spend most of your time

Figure 1.18 *Mental block*

The Far Side Cartoon ....MENTAL BLOCK

Remember there is life outside of C++ .....