COMPREHENSIVE EDITION

PROGRAMMING AND PROBLEM SOLVING WITH C++

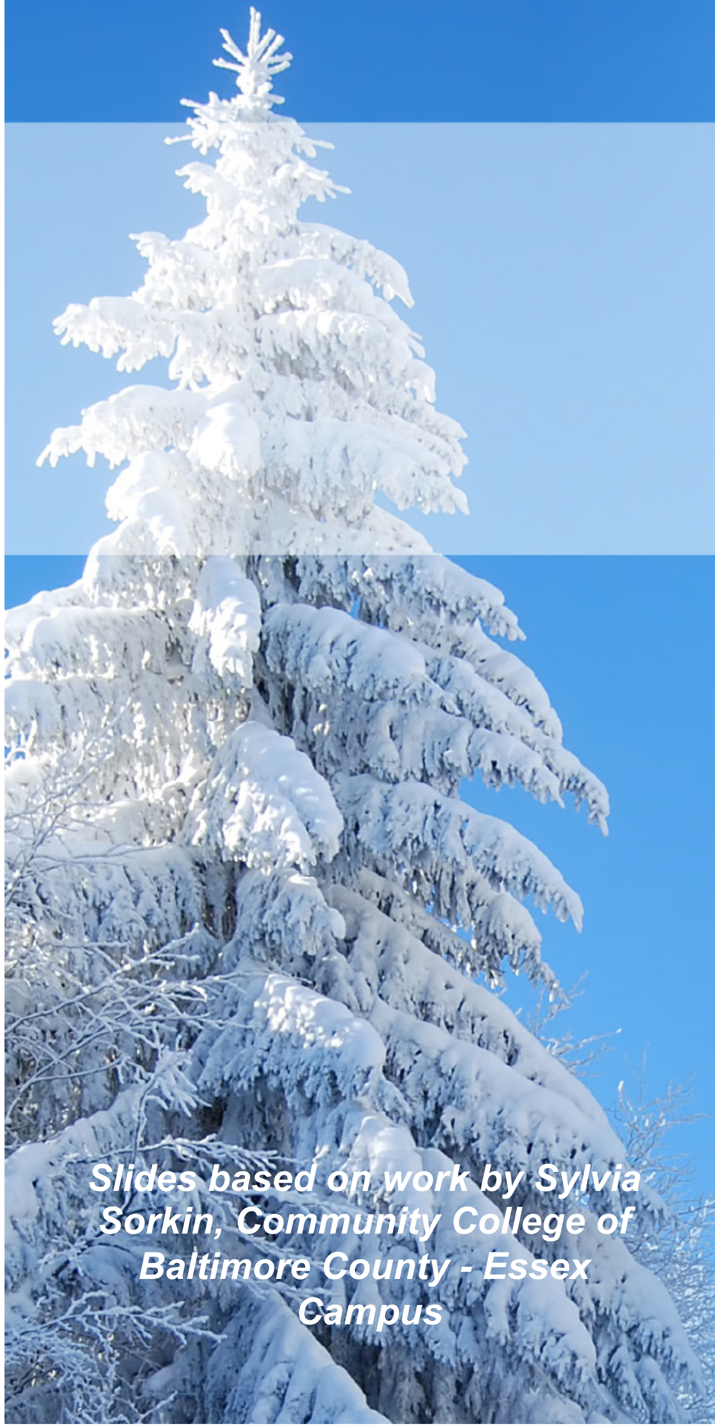
SIXTH EDITION

Nell Dale and Chip Weems

Chapter 1

Overview of Programming and Problem Solving

Slides based on work by Sylvia Sorkin, Community College of Baltimore County - Essex Campus

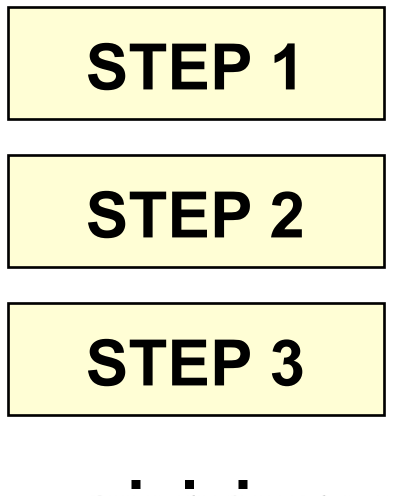


Chapter 1 Topics

- Computer Programming
- Programming Life-Cycle Phases
- Creating an Algorithm
- Machine Language vs. Assembly Language vs. High Level Languages
- Compilation and Execution Processes
- C++ History
- Basic Control Structures
- Computer Components
- Computing Profession Ethics
- Problem-Solving Techniques

What is Computer Programming?

- It is the process of planning a sequence of steps(called instructions) for a computer to follow.



Programming Life Cycle Phases

- Problem-Solving
- Implementation
- Maintenance

Problem-Solving Phase

- Analyze the problem and specify what the solution must do
- Develop a general solution(algorithm) to solve the problem
- Verify that your solution really solves the problem

Sample Problem

Suppose a programmer needs to determine an employee's weekly wages.

How would the calculations be done by hand?

One Employee's Wages

In one week an employee works 52 hours at the hourly pay rate of $24.75. Assume a 40.0 hour normal work week and an overtime pay rate factor of 1.5.

What are the employee's wages?

```
40 x $ 24.75           =   $990.00
12 x 1.5 x $ 24.75=   $445.50
                           _________
                      $   1435.50
```

Weekly Wages, in General

```
If hours are more than 40.0
        wages =
                (40.0 * payRate) +
                (hours - 40.0) * 1.5 *payRate
```

RECALL EXAMPLE

(40 x $ 24.75) +(12 x 1.5 x $ 24.75) = $1435.50

```
otherwise
        wages = hours * payRate
```

An Algorithm

- An algorithm is a step-by-step procedure for solving a problem
  - with a finite amount of data
  - in a finite amount of time

Algorithm to Determine an Employee’ s Weekly Wages

1. Get the employee’ s hourly payRate
2. Get the hours worked this week
3. Calculate this week’ s regular wages
4. Calculate this week’ s overtime wages(if any)
5. Add the regular wages to overtime wages(if any) to determine total wages for the week

What is a Programming Language?

- A programming language is a language with strict grammar rules, symbols, and special words used to construct a computer program

Implementation Phase: Program

- Translating your algorithm into a programming language is called coding
- With C++, you use

  Documentation -- your written comments

  Compiler -- translates your program into machine language

  Main Program -- may call subalgorithms

Implementation Phase: Test

- Testing your program means running(executing) your program on the computer, to see if it produces correct results
- If it does not, then you must find out what is wrong with your program or algorithm and fix it--this is called debugging

Maintenance Phase

- Use and modify the program to meet changing requirements or correct errors that show up in using it
- Maintenance begins when your program is put into use and accounts for the majority of effort on most programs
- **Wholly rewriting program with a clear design sometimes a useful alternative to modifying the existing program to meet changing requirements**

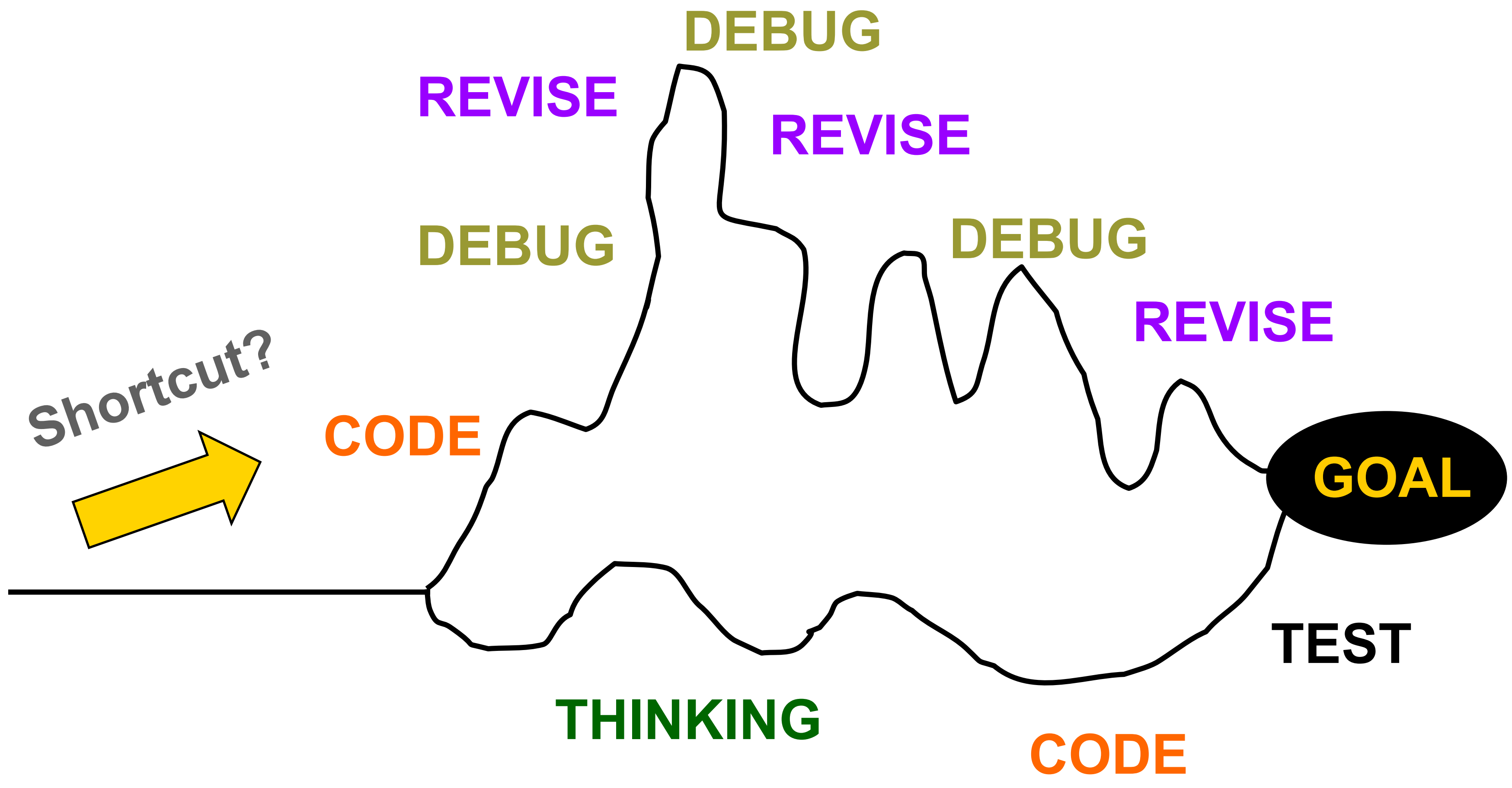
Software Maintenance Tips

- Check the existing code works as claimed
- Make changes to a *copy* of the existing code
- After acheiving desired functionality, change related aspects of the program to leave clean, consistent code for the next programmer
- Keep backup copies of current version of code to assist in developing new programs

Programming Life Cycle

1. **Problem-Solving Phase**
   - Analysis and Specification
   - General Solution(Algorithm)
   - Verify
2. **Implementation Phase**
   - Concrete Solution(Program)
   - Test
3. **Maintenance Phase**
   - Use
   - Maintain

A Tempting Shortcut?



Memory Organization

- Two circuit states correspond to 0 and 1
- Bit(short for binary digit) refers to a single 0 or 1
- Bit patterns represent both the computer instructions and computer data
- 1 byte = 8 bits
- 1 KB = 1024 bytes
- 1 MB = 1024 x 1024 = 1,048,576 bytes

How Many Possible Digits?

- **Binary**(base 2) numbers use 2 digits:
  just 0 and 1

- **Decimal**(base 10) numbers use 10 digits:
  0 through 9

Machine Language

- Is not portable
- Runs only on a specific type of computer
- Is made up of binary-coded instructions(strings of 0s and 1s)
- Is the language that can be directly used by the computer

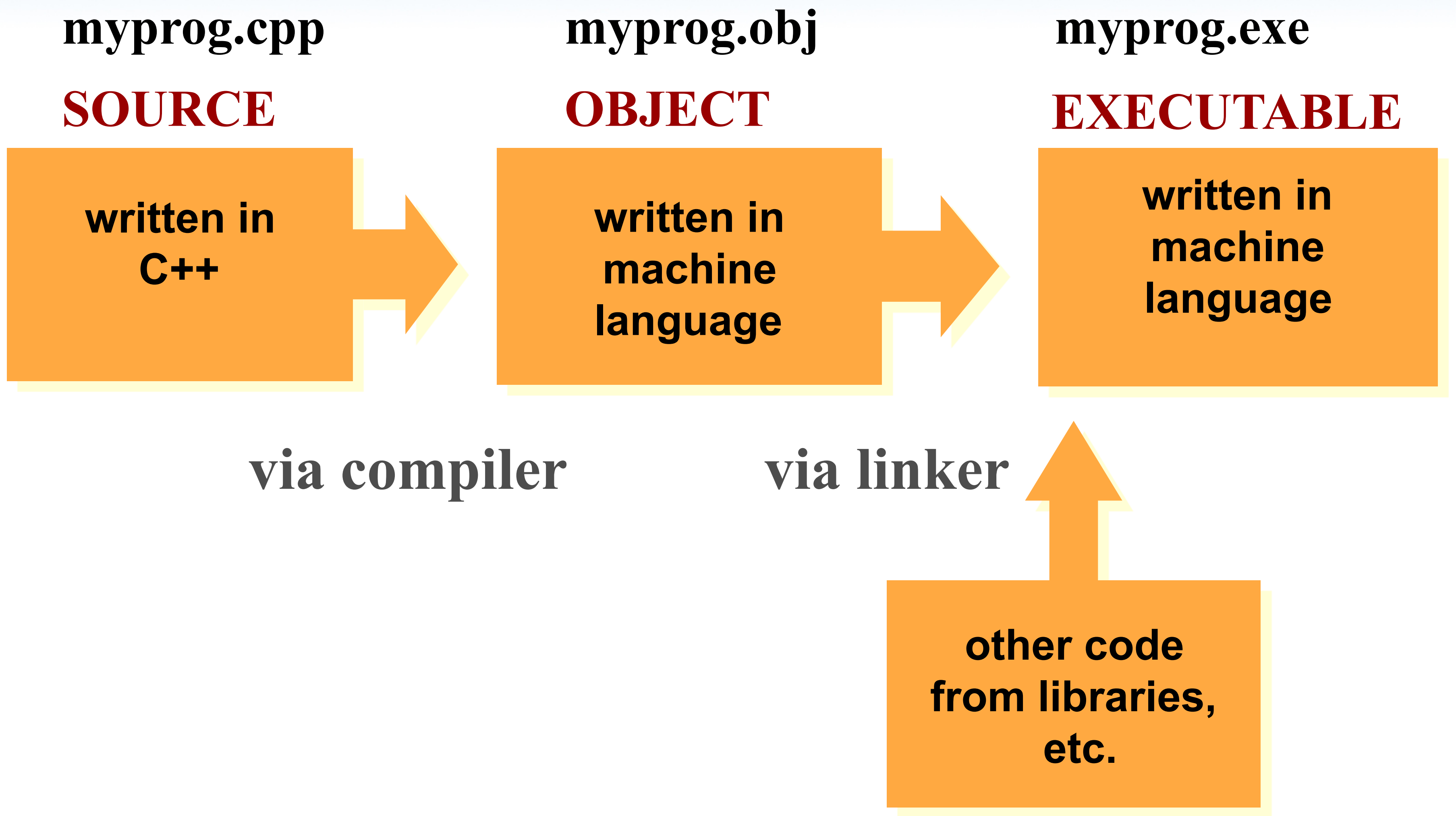
Assembly Language

- An programming upgrade from machine language
- Instructions for program are in a *mnemonic*
- Computer cannot directly execute the instructions
- An assembler program translates the assembly language instructions into machine binary code

High Level Languages

- Portable
- User writes program in language similar to natural language such as English
- Many use a compiler to translate programs written in certain high-level languages
- Examples -- **FORTRAN, COBOL, Pascal, Ada, Modula-2, C++, Java**
- Most are standardized by ISO/ANSI to provide an official description of the language

Three C++ Program Stages



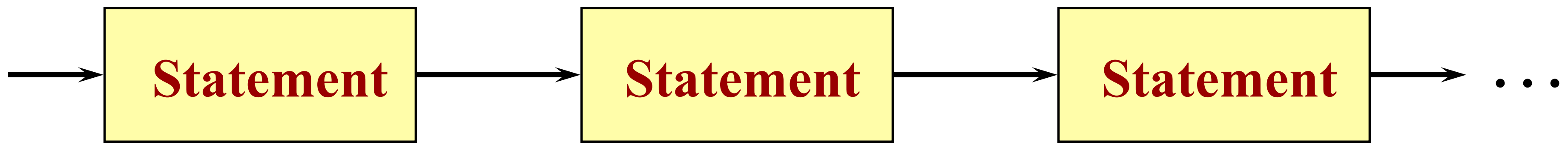
Java Programming Language

- Achieves portability by using both a compiler and an interpreter
- First, a Java compiler translates a Java program into an intermediate Bytecode--not machine language
- Then, an interpreter program called the Java Virtual Machine(JVM) translates a single instruction in the bytecode program to machine language and immediately runs it, one at a time

Basic Control Structures

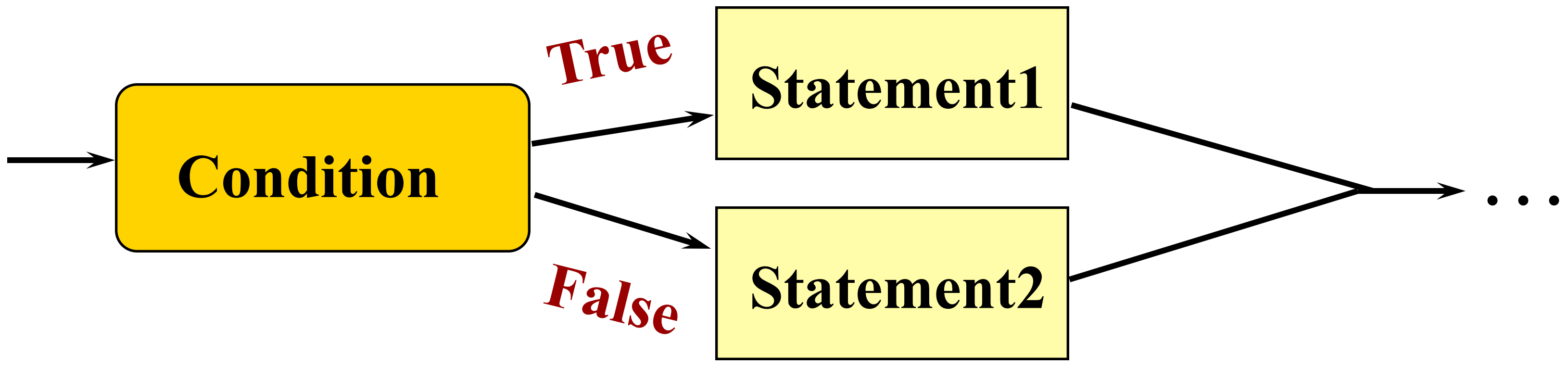
- A sequence is a series of statements (instructions) that execute one after another
- A selection(branch) statement is used to determine which of two different statements to execute depending on certain conditions
- A looping(repetition) statement is used to repeat statements while certain conditions are met
- A subprogram is a smaller part of another program; a collection of subprograms solves the original problem
- Each of these ways of structuring statements controls the order in which the computer executes the statements

SEQUENCE



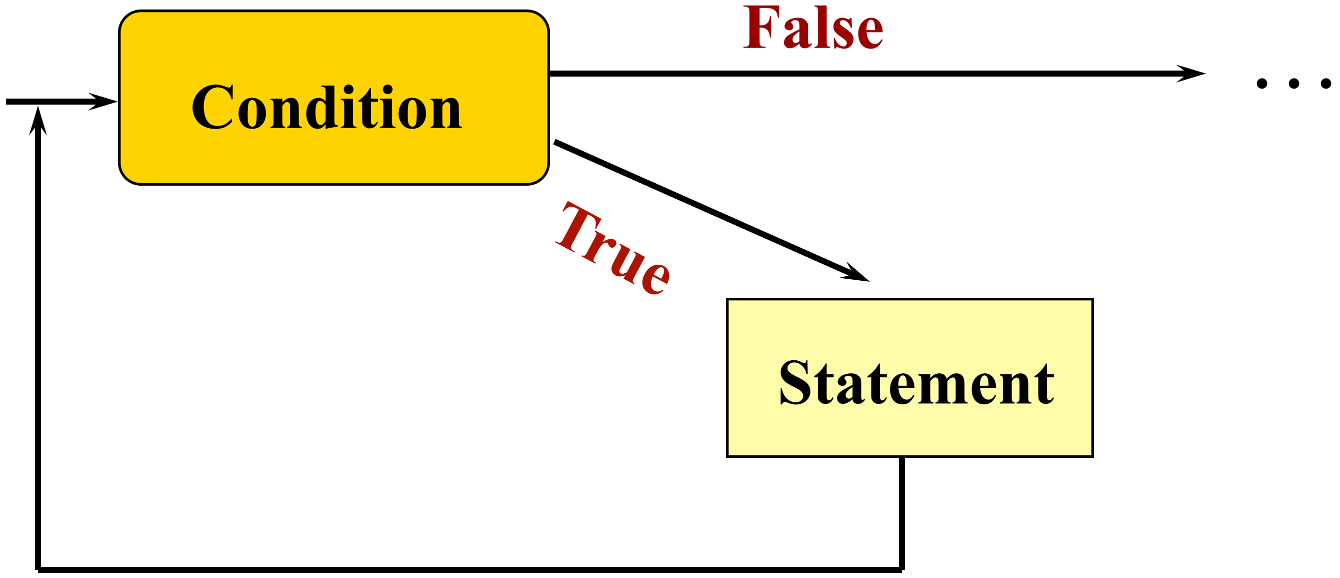
SELECTION(branch)

IF Condition THEN Statement1 ELSE Statement2

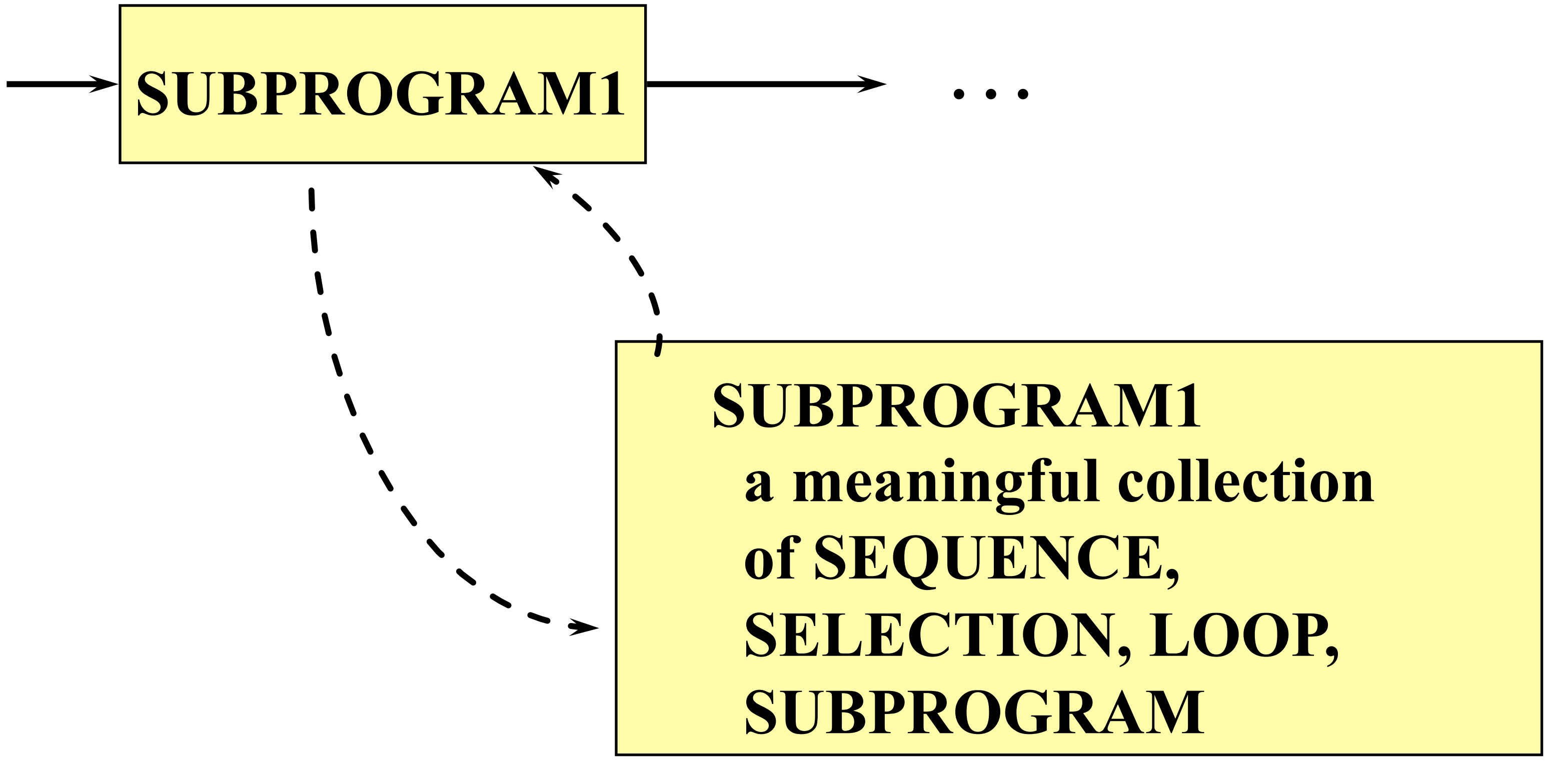


LOOP(repetition)

WHILE Condition DO Statement1



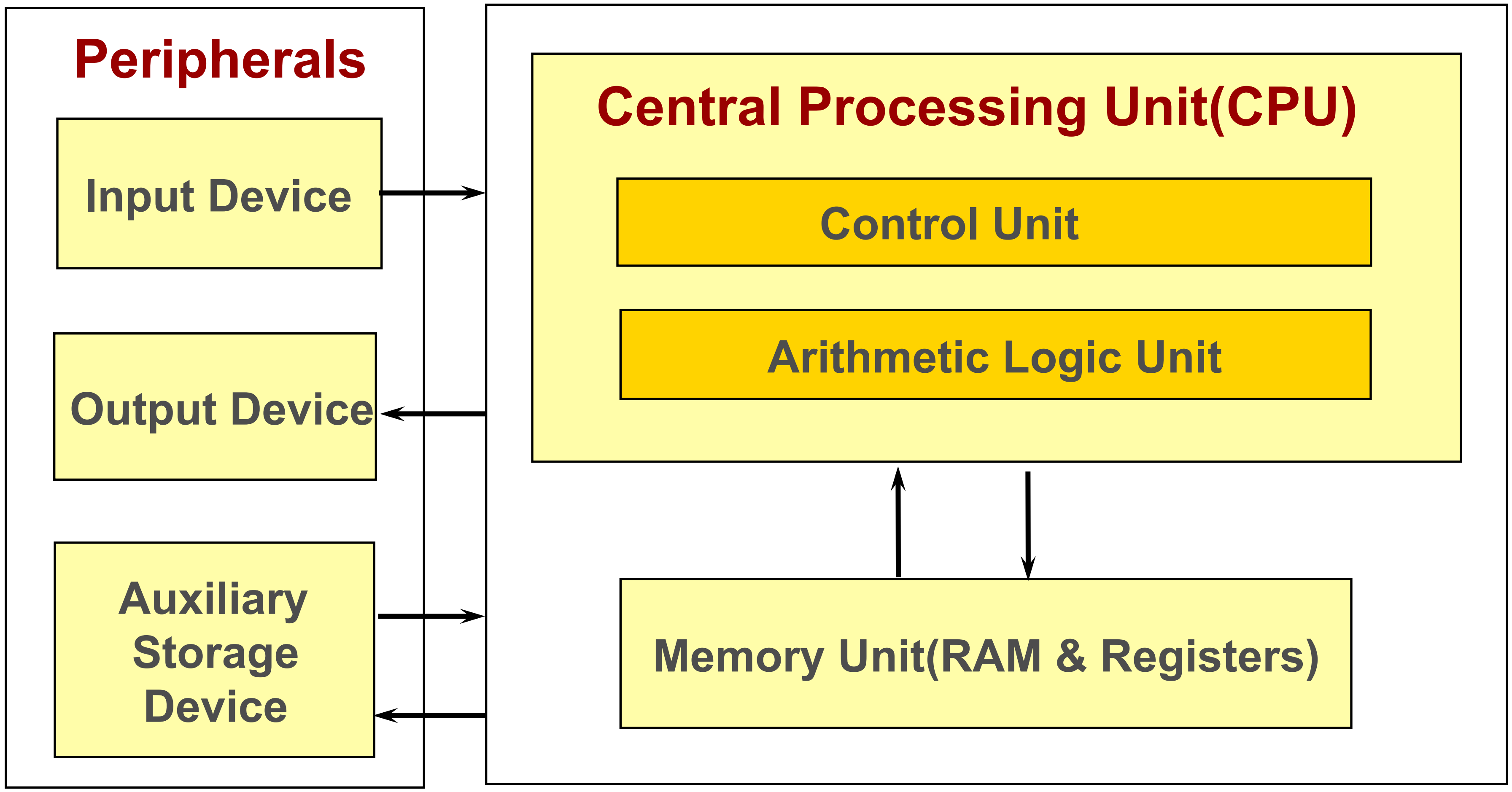
SUBPROGRAM(function)



Some C++ History

- 1972 : Dennis Ritchie at Bell Labs designs C and 90% of UNIX is then written in C
- Late 70's : OOP becomes popular
- Bjarne Stroustrup at Bell Labs adds features to C to form "C with Classes"
- 1983 : Name C++ first used
- 1998 : ISO/ANSI standardization of C++

Computer Components



Memory Unit

- Is an ordered sequence of storage cells, each capable of holding a piece of information
- Each cell has its own unique address
- The information held can be **input data**, **computed values**, or your **program instructions**

Central Processing Unit

- Has two components to execute program instructions
  - **Arithmetic/Logic Unit** performs arithmetic operations, and makes logical comparisons
  - **Control Unit** controls the order in which your program instructions are executed

Peripherals

- Are input, output, or auxiliary storage devices attached to a computer
  - Input Devices include keyboard and mouse
  - Output Devices include printers, video display, LCD screens
  - Auxiliary Storage Devices include disk drives, scanners, CD-ROM and DVD-ROM drives, modems, sound cards, speakers, and digital cameras

Computing Profession Ethics

- Copy software only with permission from the copyright holder
- Give credit to another programmer by name whenever using his/her code
- Use computer resources only with permission
- Guard the privacy of confidential data
- Protect computer resources against harmful programs, malware
- Use software engineering principles to develop software free from errors

What are the Areas of Computer Science?

The Computing Curriculum 1991(ACM/IEEE)

- Algorithms and Data Structures
- Architecture
- Artificial Intelligence and Robotics
- Database and Information Retrieval
- Human-Computer Communication
- Numerical and Symbolic Computation
- Operating Systems
- Programming Languages
- Software Engineering
- Social and Professional Context

Problem Solving Techniques

- **Ask questions** -- about the data, the process, the output, error conditions
- **Look for familiar things** -- certain situations arise again and again
- **Solve by analogy** -- it may give you a place to start
- **Use means-ends analysis** -- determine the I/O and then work out the details

More Problem Solving Techniques

- **Divide and conquer** -- break up large problems into manageable units
- **Building-block approach** -- can you solve small pieces of the problem?
- **Merge solutions** -- instead of joining them end to end to avoid duplicate steps
- **Overcome mental block** -- by rewriting the problem in your own words

Is a year a leap year?

Problem You need to write a set of instructions that can be used to determine whether a year is a leap year. The instructions must be very clear because they are to be used by a class of fourth graders, who have just learned about multiplication and division. They plan to use the instructions as part of an assignment to determine whether any of their relatives were born in a leap year.

Leap Year Algorithm

```
Prompt the user to enter a four-digit year
Read the year
If IsLeapYear
        Write "Year is a leap year"
Otherwise
        Write "Year is not a leap year"
```

IsLeapYear Algorithm

```
Divide the year by 4
If the remainder isn't zero,
    Return false(The year is not a leap year)
Otherwise divide the year by 10 and
If the remainder isn't 0,
    Return true(The year is a leap year)
    Otherwise, divide the year by 400 and
    If the remainder isn't 0
          Return false(The year is not a leap year)
          Otherwise, Return true(The year is a leap year)
```

C++ Program

```
//******************************************************
// LeapYear program
// This program inputs a year and prints whether the year
// is a leap year or not
//******************************************************
#include <iostream>        // Access output stream

using namespace std;        // Access cout, endl, cin

bool IsLeapYear(int);       // Prototype for subalgorithm


int main()
{
 …
}
```

Body of Main

```
{
    int year;                  // Year to be tested
    cout << "Enter a year AD, for example, 1997."
         << endl;              // Prompt for input
    cin >> year;               // Read year

    if(IsLeapYear(year))              // Test for leap year
        cout << year << " is a leap year."  << endl;
    else
        cout << year << " is not a leap year."  <<
   endl;
    return 0;                  // Indicates successful
                               //  completion
}
```

IsLeapYear

```
bool IsLeapYear(int year)
// IsLeapYear returns true if year is a leap year and
// false otherwise
{
    if(year % 4 != 0)          // Is year not divisible by 4?
        return false;           // If so, can't be a leap year
    else if(year % 100 != 0) // Is year not a multiple of 100?
        return true;            // If so, is a leap year
    else if(year % 400 != 0) // Is year not a multiple of 400?
        return false;           // If so, then is not a leap year
    else
        return true;            // Is a leap year

}
```