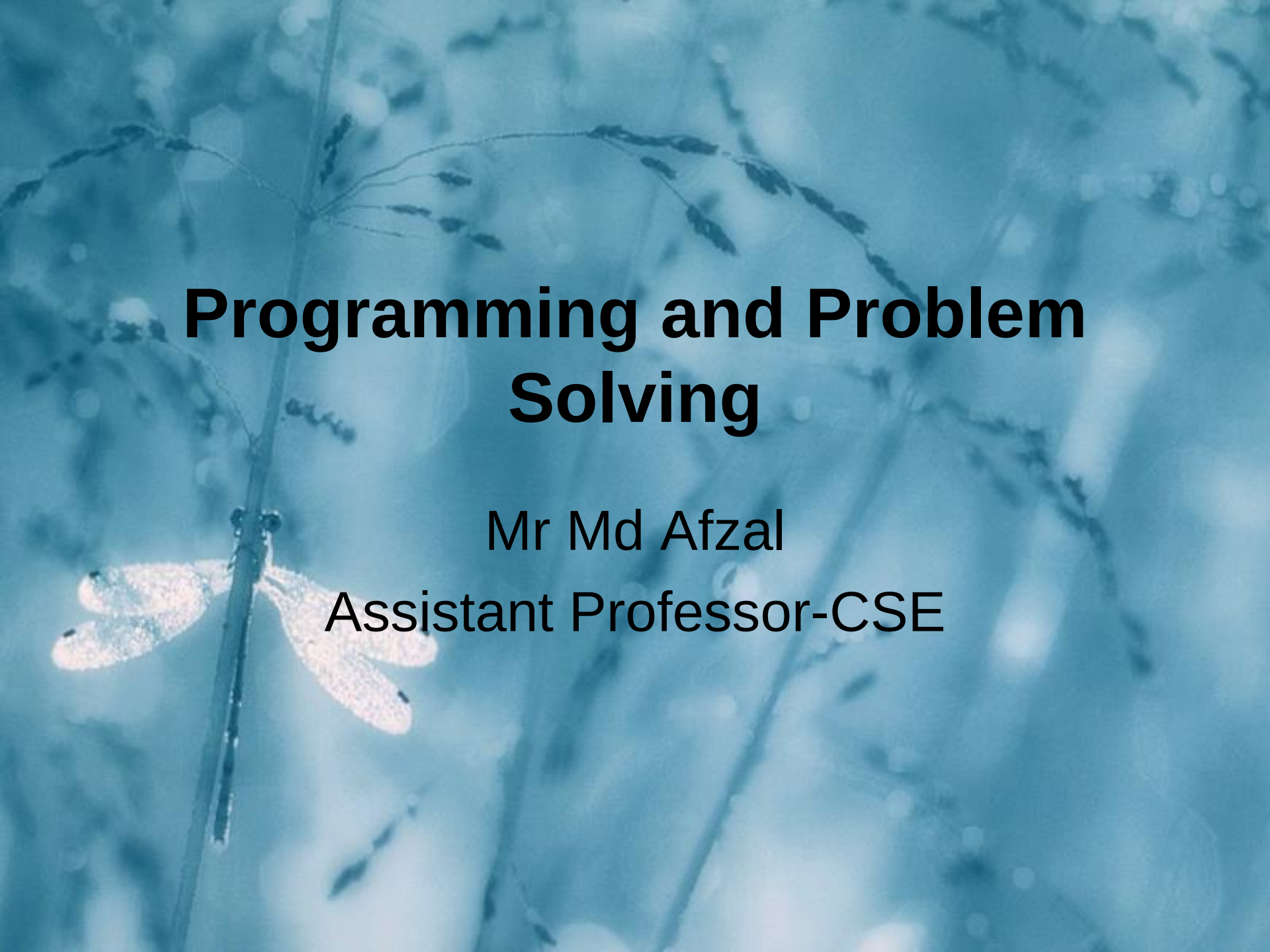


Programming and Problem Solving

Mr Md Afzal

Assistant Professor-CSE

Introduction

- In this course, we learn how to use the computer effectively to solve problems

- Let us go over the syllabus and then start the introductory topics.

Syllabus

- **Textbook:**

  Programming and Problem Solving With C++,3rd Edition

  - Nell Dale, Chip Weems amd Mark Headington, Jones and Bartlett 2002

- **Grading:**

– HW-I		10%
– HW-II		10%
– HW-III		10%
– HW-IV		10%
– Demos and class work		10%
– Exam-I	In-Class	25%
– Exam-II	In-Class	25%

  – **NOTE:** Exam-I covers all the topics until last lecture before the exam. Exam-II covers all topics from first lecture after Exam-I until the last lecture before exam week.

Topics

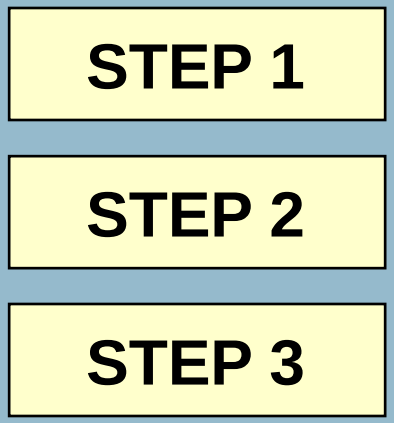
- Introduction to Computers
- Overview of C++
- Top Down Design
- Selection
- Repetition
- Function Arguments
- Formatting and Files
- Arrays and Structures
- User defined classes
- Recursion

Chapter 1 Topics

- **Computer Programming**
- **Programming Life-Cycle Phases**
- **Creating an Algorithm**
- **Machine Language vs. High Level Languages**
- **Compilation and Execution Processes**
- **C++ History**
- **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

1. **Problem-Solving**
2. **Implementation**
3. **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

A programmer needs an algorithm 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, then

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 is . . .

- **a step-by-step procedure for solving a problem 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?

- **It 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**

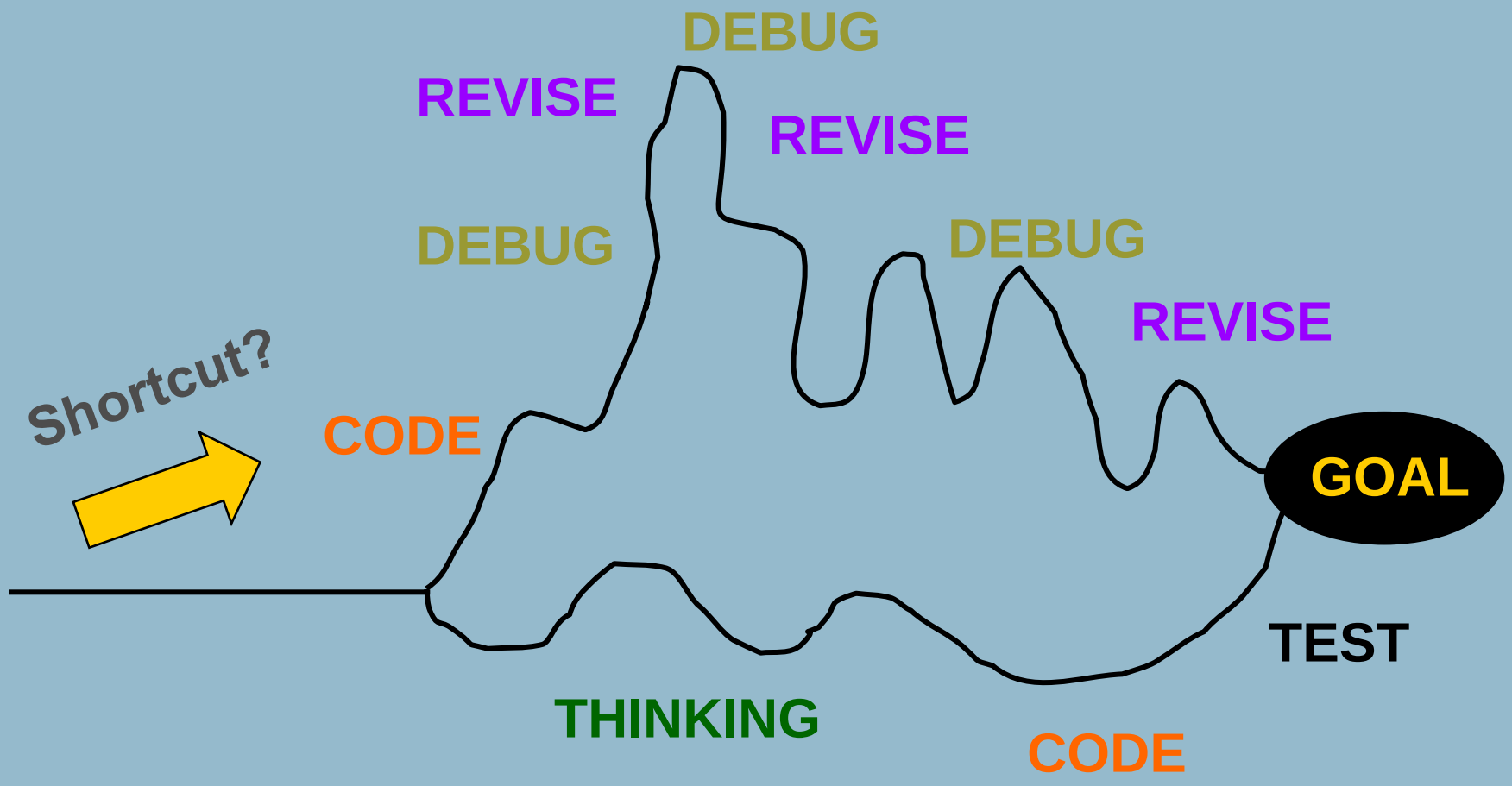
Maintenance Phase

- **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**

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

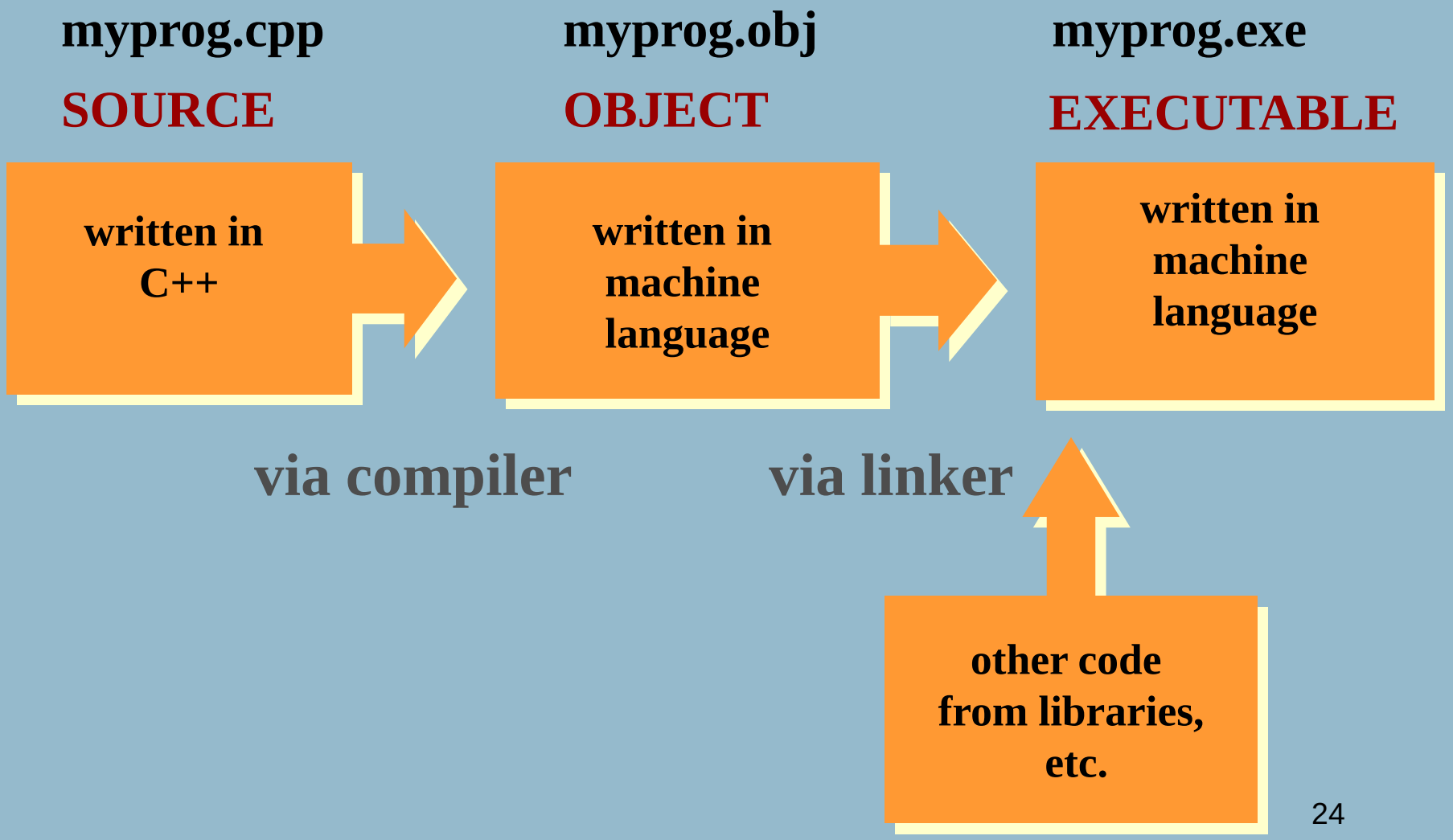
Machine Language

- is not portable
- runs only on 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

High Level Languages

- are portable
- user writes program in language similar to natural language
- 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

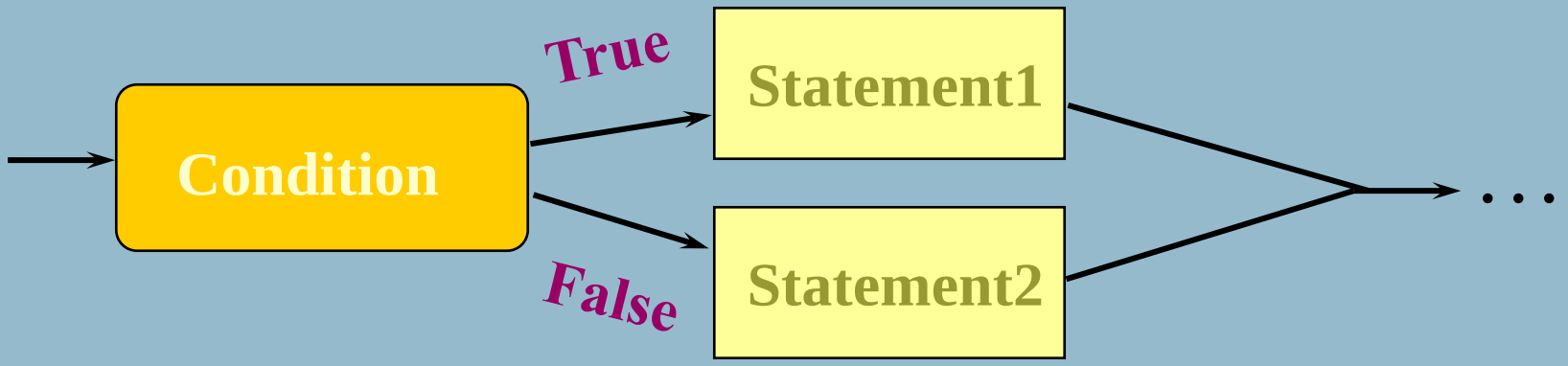
Basic Control Structures

- a sequence is a series of statements that execute one after another
- selection (branch) is used to execute different statements depending on certain conditions
- Looping (repetition) is used to repeat statements while certain conditions are met.
- a subprogram is used to break the program into smaller units

SEQUENCE

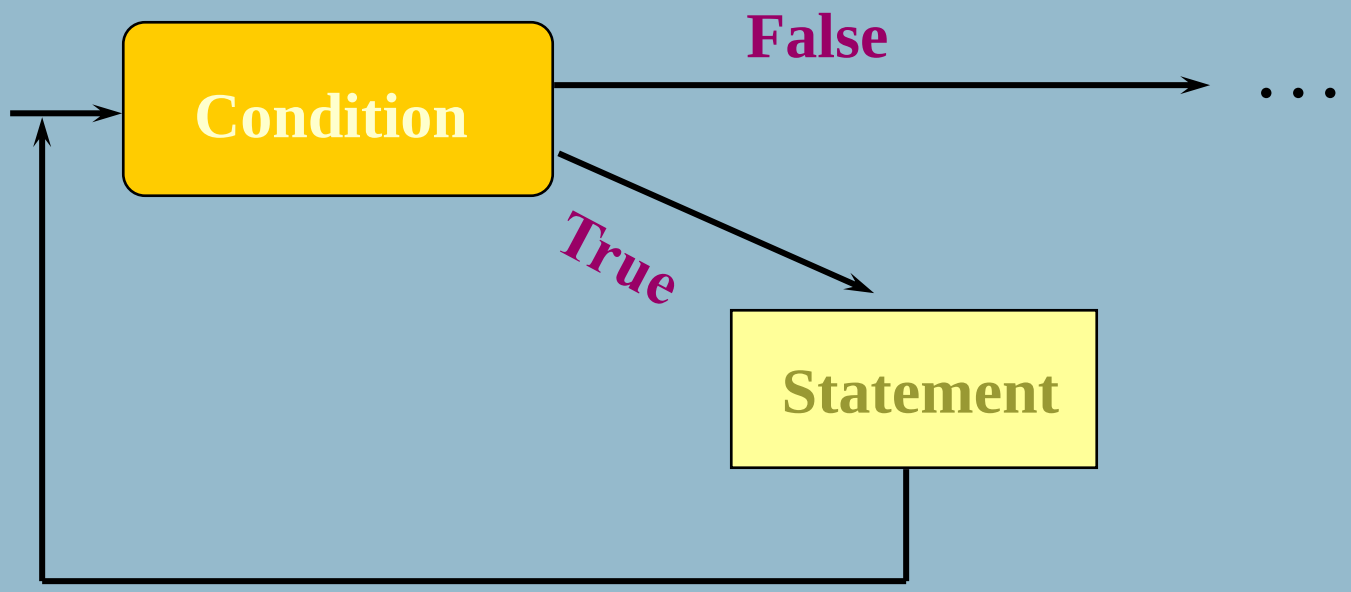
SELECTION (branch)

IF Condition THEN Statement1 ELSE Statement2

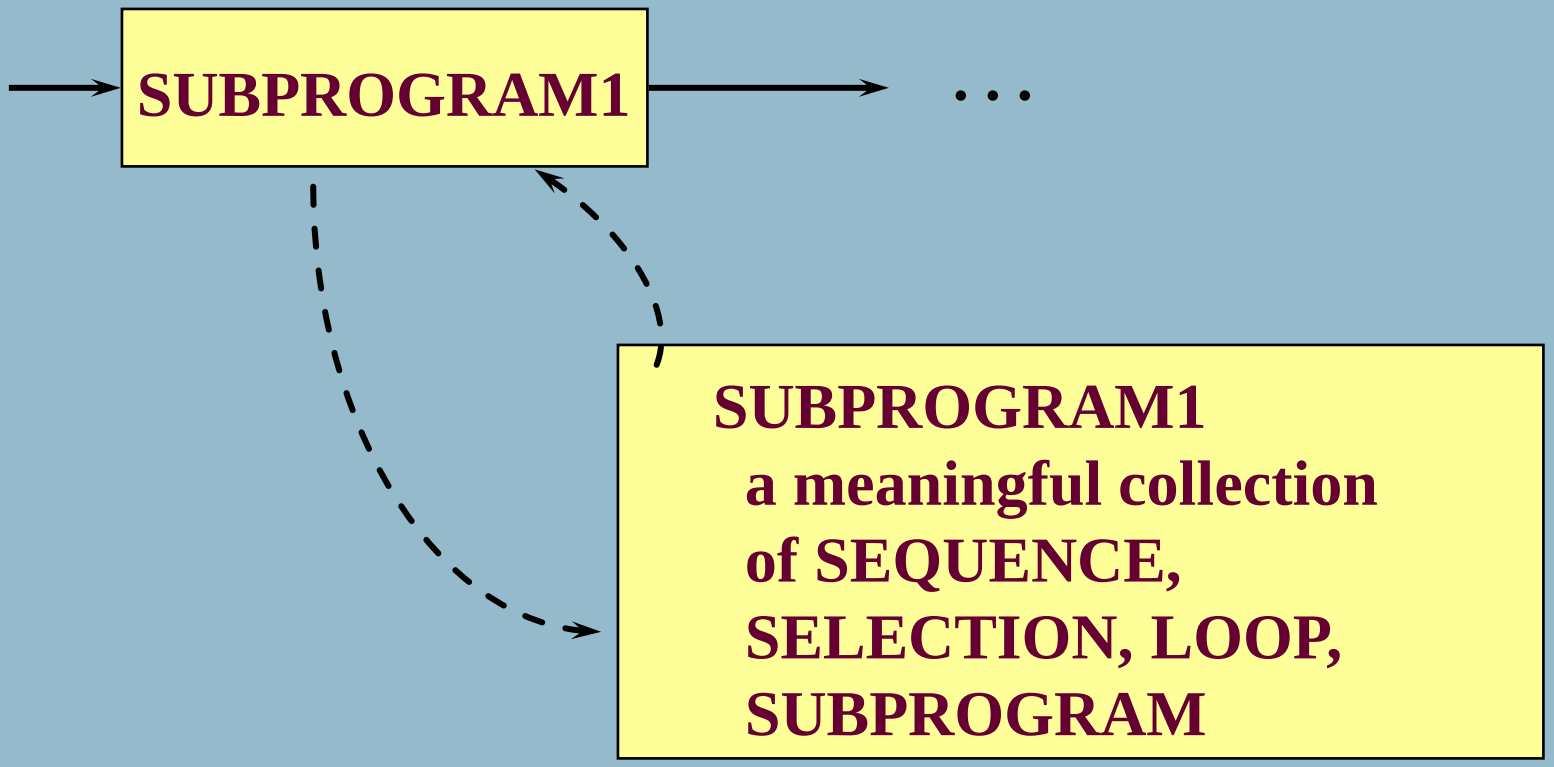


LOOP (repetition)

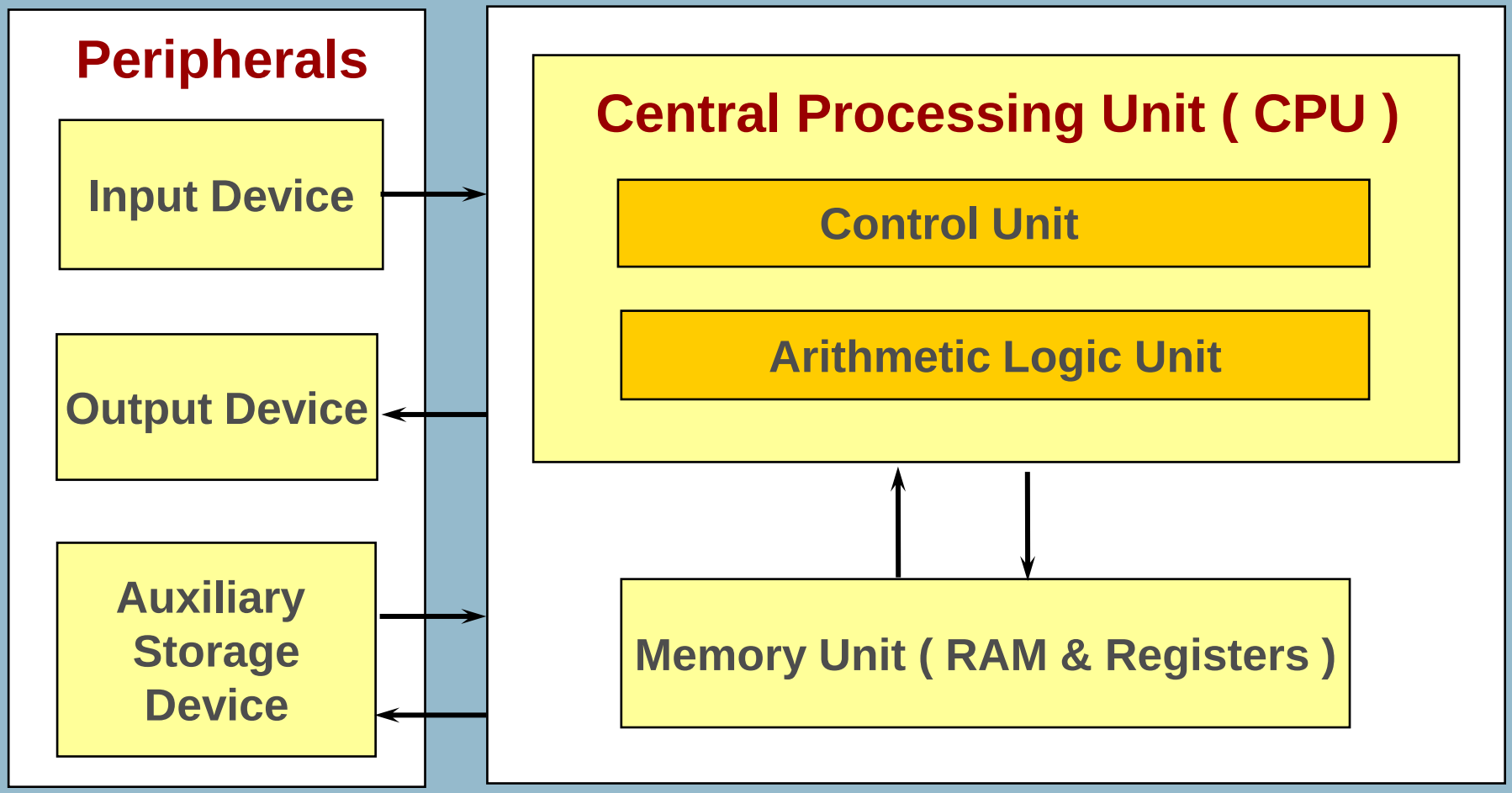
WHILE Condition DO Statement1



SUBPROGRAM (function)



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 2 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.**

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++**

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**
- **use software engineering principles to develop software free from errors**

What is 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 APPPROACH -- 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.**

Company Payroll Case Study

A small company needs an interactive program to figure its weekly payroll. The payroll clerk will input data for each employee, and each employee's wages and data should be saved in a secondary file.

Display the total wages for the week on the screen.

One Employee's Wages

In one week employee ID # 4587 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

Problem-Solving Phase

What information will be used?

INPUT DATA from outside the program

FORMULA CONSTANTS used in program

COMPUTED VALUE produced by program

OUTPUT RESULTS written to file or screen by program

Problem-Solving Phase

INPUT DATA

Employee ID Number

Hourly payRate

Hours worked

FORMULA CONSTANTS

Normal work hours (40.0)

Overtime pay rate factor (1.5)

OUTPUT RESULTS

Hourly payRate

Hours worked

Wages

COMPUTED VALUE

Wages

Week's Wages, in General

If hours are more than 40.0, then

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

Algorithm for Company Payroll Program

- initialize total company payroll to 0.0
- repeat this process for each employee:
  1. Get the employee's ID empNum
  2. Get the employee's hourly payRate
  3. Get the hours worked this week
  4. Calculate this week's wages
  5. Add wages to total company payroll
  6. Write empNum, payRate, hours, wages to file
- write total company payroll on screen

C++ Program

```
// ********************************************************
//  Payroll program
//  This program computes each employee's wages and
//  the total company payroll
// ********************************************************

#include <iostream>          // for keyboard/screen I/O
#include <fstream>           // for file I/O

using namespace std;

void  CalcPay ( float,  float,  float& ) ;

const  float  MAX_HOURS = 40.0;  // Maximum normal hours
const  float  OVERTIME = 1.5;    // Overtime pay factor
```

C++ Code Continued

```
int  main( )
{
     float     payRate;              //  Employee's pay rate
     float     hours;                //  Hours  worked
     float     wages;                //  Wages earned
     float     total;                //  Total company payroll
     int       empNum;               //  Employee  ID  number
     ofstream  payFile;              // Company  payroll file

     payFile.open( "payfile.dat" );      // Open file
     total = 0.0;                        // Initialize total
```

```
cout << "Enter employee number: ";  // Prompt
cin  >> empNum;                     // Read ID number

while ( empNum  !=  0 )             // While not done
{
    cout << "Enter pay rate: ";
    cin  >> payRate ;               // Read pay rate
    cout << "Enter hours worked: ";
    cin  >> hours ;                 // and hours worked

    CalcPay(payRate, hours, wages); // Compute wages

    total = total + wages;          // Add to total

    payFile << empNum << payRate
            << hours << wages << endl;

    cout << "Enter employee number: ";
    cin  >> empNum;                 // Read ID number
}
```

```
    cout <<  "Total payroll is  "
         <<  total  << endl;

    return 0 ;                          //  Successful completion
}

// ********************************************************

void  CalcPay ( /* in */    float   payRate ,
                /* in */    float   hours ,
                /* out */   float&  wages )

//  CalcPay computes wages from the employee's pay rate
//  and the hours worked, taking overtime into account

{
    if ( hours  >  MAX_HOURS )
        wages = (MAX_HOURS * payRate ) +
                (hours - MAX_HOURS) * payRate * OVER_TIME;
    else
        wages  =  hours * payRate;
}
```