

CS1044 – Intro. To Programming in C++

Dr. Craig A. Struble

What is Programming?

- *Planning or scheduling the performance of a task.*
- Consciously thinking about each step
- Example: Accelerating in a car
 1. Move right foot to gas pedal
 2. Apply pressure to gas pedal with right foot
 3. If speed is too high, apply less pressure.
 4. If speed is too low, apply more pressure.

Are Computers Intelligent?

- Do we really need to be involved in programming computers?
 - They have beaten world chess champions.
 - They help predict weather patterns.
 - They can perform arithmetic quickly.
- So, a computer has an IQ of _____.

3

Introduction

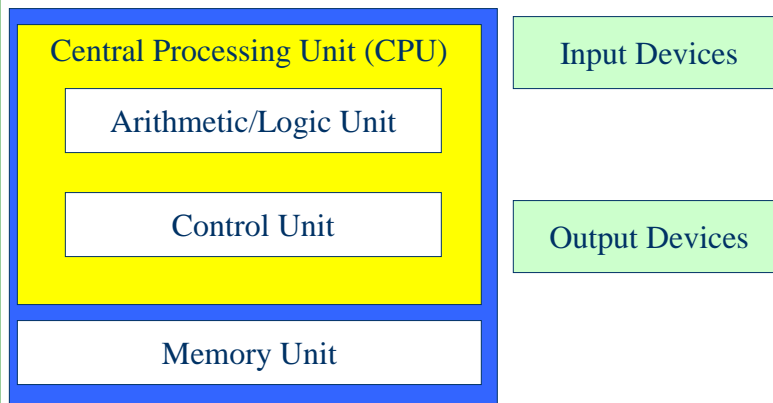
What Do We Have To Do?

- Computers cannot analyze problems and devise solutions.
- Humans (that's us) must
 - Analyze and understand a problem;
 - Devise a sequence of steps to solve the problem;
 - Translate the steps into a *computer language*.

4

Introduction

Basic Computer Components

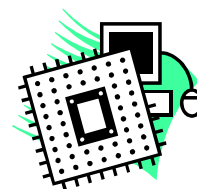


5

Introduction

Central Processing Unit (CPU)

- Executes stored *instructions*
- Arithmetic/Logic Unit (ALU)
 - Performs arithmetic and logical operations
- Control Unit
 - Controls the other components
 - Guarantees instructions are executed in sequence



6

Introduction

Memory Unit

Address

1001	
1002	
1003	
1004	
1005	
1006	
1007	
1008	
1009	

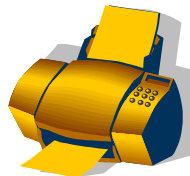
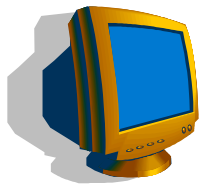
- Ordered sequence of *cells* or *locations*
- Stores instructions and data in *binary*
- Types of memory
 - Read-Only Memory (ROM)
 - Random Access Memory (RAM)

7

Introduction

Input and Output Devices

- Interaction with humans
- Gathers data (Input)
- Displays results (Output)



8

Introduction

What is Computer Programming?

- *Planning or scheduling a sequence of steps for a computer to follow to perform a task.*
- Basically, telling a computer what to do and how to do it.

9

Introduction

What Is A Computer Program?

- *A sequence of steps to be performed by a computer.*
- Expressed in a *computer language*.

10

Introduction

A Computer Program In C++

```
// This program converts miles to kilometers.
// From Problem Solving, Abstraction, & Design Using C++
// by Frank L. Friedman and Elliot B. Koffman
#include <iostream>
using namespace std;

int main() {
    const float KM_PER_MILE = 1.609; // 1.609 km in a mile
    float miles, // input: distance in miles
           kms; // output: distance in kilometers
    // Get the distance in miles
    cout << "Enter the distance in miles: ";
    cin >> miles;
    // Convert the distance to kilometers and display it.
    kms = KM_PER_MILE * miles;
    cout << "The distance in kilometers is " << kms << endl;

    return 0;
}
```

11

Introduction

Computer Languages

- A set of
 - Symbols (punctuation),
 - Special words or keywords (vocabulary),
 - And rules (grammar)used to construct a computer program.

12

Introduction

Differences In Computer Languages

- Languages differ in
 - Size (or complexity)
 - Readability
 - Expressivity (or writability)
 - "Level"
 - closeness to instructions for the CPU

13

Introduction

Machine Language

- Binary-coded instructions
- Used directly by the CPU
- Lowest level language
- Every program step is ultimately a machine language instruction

Address Contents

2034	10010110
2035	11101010
2036	00010010
2037	10101010
2038	10010110
2039	11101010
2040	11111111
2041	01010101
2042	10101101

14

Introduction

Assembly Language

- Each CPU instruction is labeled with a *mnemonic*.
- Very-low level language
 - Almost 1 to 1 correspondence with machine language
- Translated to machine language by an *assembler*.

Mnemonic	Instruction
ADD	10010011

Sample Program

```
MUL X, 10  
ADD X, Y  
STO Z, 20  
SUB X, Z
```

15

Introduction

High-Level Languages

- Closer to natural language
- Each step maps to several machine language instructions
- Provides support for *abstractions*
 - Easier to state and solve problems
 - Example: Colors

16

Introduction

Examples of High-Level Languages

Language	Primary Uses
Pascal	Learning to program
C++	General purpose
FORTRAN	Scientific programming
PERL	Web programming, text processing
Java	Web programming, application programming
COBOL	Business

17

Introduction

Basic Programming Language Structures

- Sequence
 - Execute steps or *statements* in the language one after another.

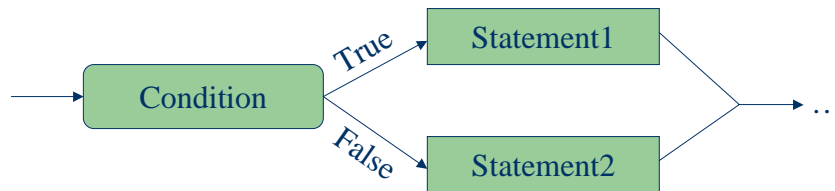


18

Introduction

Basic Programming Language Structures

- Selection (AKA *branch* or *decision*)
 - Selectively execute statements based on some condition being true or false.
 - `IF condition THEN statement1 ELSE statement2`

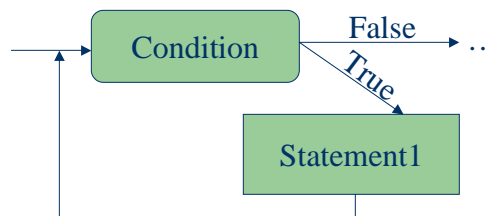


19

Introduction

Basic Programming Language Structures

- Loop (AKA *repetition* or *iteration*)
 - Repeat a statement several times until a specified condition is false.
 - `WHILE condition DO statement1`



20

Introduction

Basic Programming Language Structures

- Subprogram (AKA *procedure*, *function*, *method*, or *subroutine*)
 - A collection of the previous structures that accomplishes some smaller task.

