CS11600-1
Introduction to Programming II

- Instructor:
  Catalin Dumitrescu <cldumitr@cs.uchicago.edu>

- TAs:
  Irina Matveeva <matveeva@cs.uchicago.edu>
  Zhimin Xie <zxie@cs.uchicago.edu>

Today

- Introduction
- Tentative global schedule
- Some policies about the class
- A few words about
  - Computer organization
  - Algorithms, Programming languages, Interpreters & Compilers

Global Plan

- **Introduction** (Policies, Computer Organization, Environment)
- **non-OO Programming** (Program structure, Functions, ...)
- **OO Programming [1]** (Encapsulation, Class Members, Overloading, ...)
- **Elements of Generic Programming** (Function and Class Templates, STL)
- **OO Programming [2]** (Inheritance, Polymorphism, ...)
- **Applications & Final Review**

About the Tools

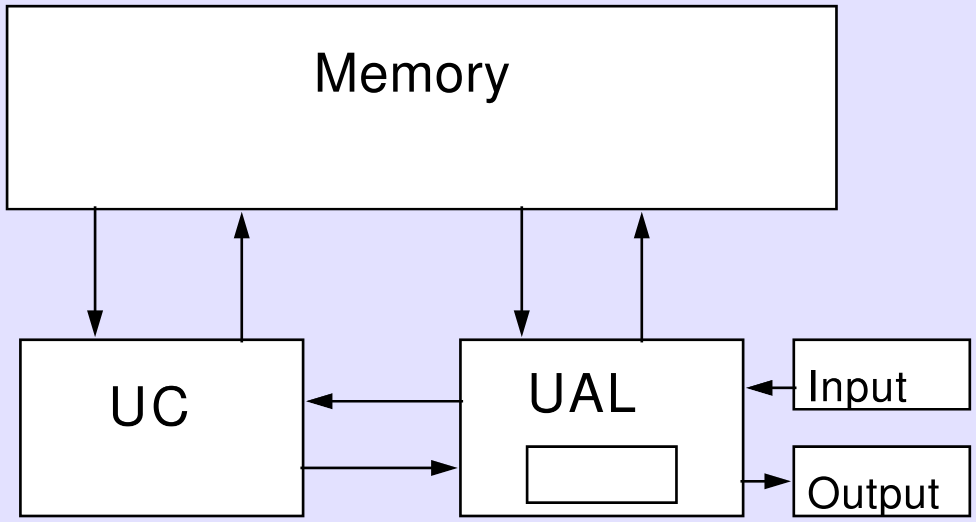
- GNU/Linux and g++ (gcc)
  - additional tools: emacs, gdb, ...
- CodeWarrior (under MacOS or Windows)
  - integrated environment
- Your programs will be graded using *g++*

Class Policies

- Regular attendance is highly recommended!
- No late assignments will be accepted without a strong motivation!
- Weekly homework problems (50%), Midterm 1 (10%), Midterm 2 (15%), Final (15%), Class Participation (10%)
- Strict format for homework problems
- Grading and Regrading

Computer Organization

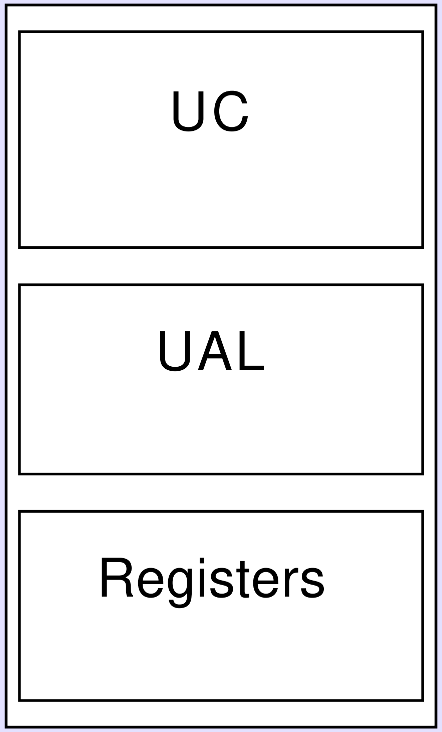
- Von Neumann's (a base for many modern computers)
  - memory
  - arithmetic and logic unit
  - control unit
  - input / output devices



CPU

- *Computer's "brain"* (?!)

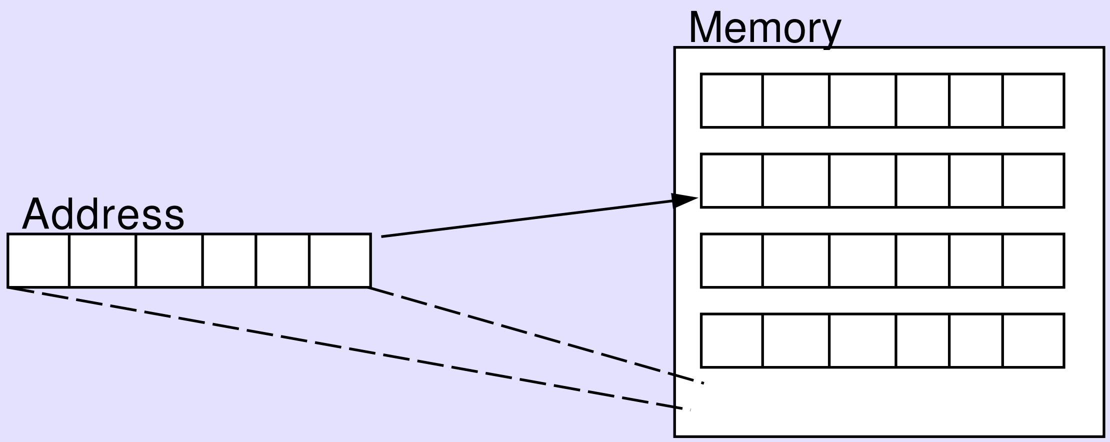
- Role - program execution
  - instruction extraction from memory
  - examination
  - execution (more or less sequentially)



Memory

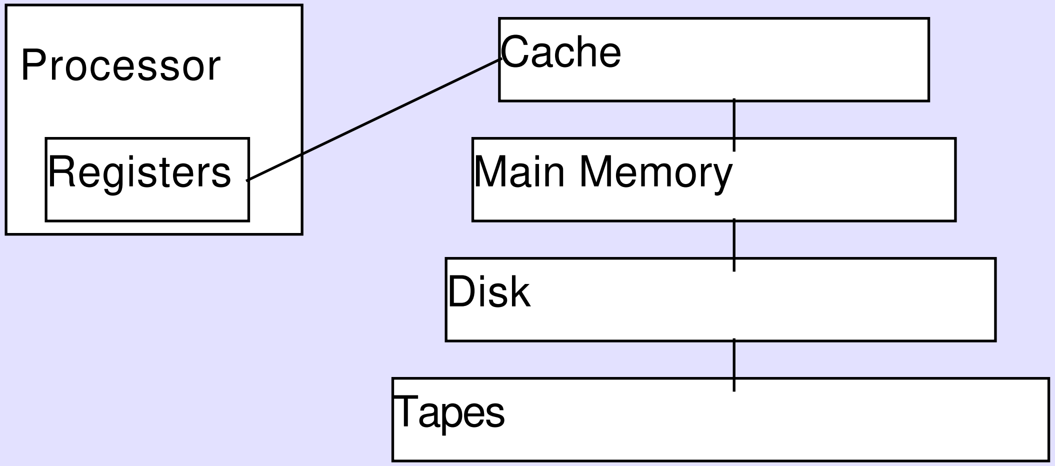
- **Role - store data and programs**

- **Elements**
  - **Bits (elementary unit): $10_{10} = A_{16} = 1010_2$**
  - **Cells (Set of bits): 16 bits store numbers from 0 to 2^16 - 1**
  - **Memory addresses: information to access cells**



Hierarchy Memory

- Cost, speed and size imposed
  - Registers
  - Caches, between:
    - processor and main memory (one, two levels)
    - main memory and disk
  - Main Memory
  - Disk Storages
  - External storages (Tapes, CDs, DVDs, On-line libraries)



Input / Output Devices

- Interaction devices with ...
  - Display
  - Keybord
  - Mouse
  - Joystick

Multi-level Organization

- Two or more levels
  - level 5: problem oriented language (compilator)
  - level 4: assembler language (assembler)
  - level 3: OS architecture language (OS)
  - level 2: architecture instruction set (direct or interpreted)
  - level 1: microarchitecture (hardware)
  - level 0: digital logic

Computer System

- Four components
  - Hardware
    - Processor, Memory, Input/Output Devices
    - Provides the basic computing resource
  - Operating System
    - intermediary between users, applications and hardware
  - Application Programs
    - ways in which hardware is used to solve problems
  - Users

Problem Solving

- Process containing many phases
  - Gaining an understanding of the problem
  - Designing a conceptual *solution*
  - Implementing the *solution*

- Solution
  - An *algorithm*
  - Ways to store data

- Algorithm
  - Any well-defined computation procedure that takes some values as input and and produces some values as output
  - Can be seen as a tool for solving a well defined computational problem

Algorithm

Example: Insert-Sort (A, N)

```
for j = 2 to N
do key = A[j]
    // Insert A[j] in the sorted sequence A[1..j-1]
    i = j-1
   while i > 0 and A[i] > key
   do A[i+1] = A[i]
       i = i - 1
    A[i+1] = key
```

Programming Language

- A language designed for programming computers
- Must be precise

- Several levels
  - Machine languages
  - Assembler languages
  - High level languages

- Why use high-level languages?
  - Allow to focus on abstract tasks
  - Compilers apply optimization procedures
  - Easy to re-use for different platforms

- Several Models: Procedural, OO, Functional, Logic

Interpreter

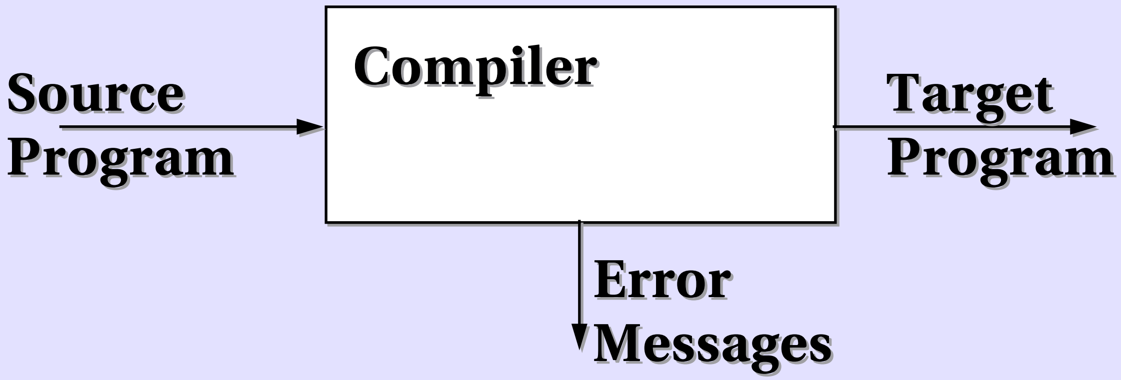
- Operations specified by the programming language are performed directly

- Used for:
  - Executing command languages (complex routines)
  - Interpreting "high-level" languages with" complex" data

Compiler

- Application that reads a program written in one language and translates it into an equivalent program in another language

- Examples



Next Time

- Introduction to C++
- Basic Statements and Notions about source files
- Preprocessing, Compilation and Linking Processes