EASTERN WASHINGTON UNIVERSITY
start something big

Introduction to C

Computer Science Department
Eastern Washington University
Yun Tian (Tony) Ph.D.

Topic for today

- Why to learn C?
- What is C?
- Generic phases to execute a C program
- Compile and Run with GCC on Linux

Why learn C programming?

- C was initially developed by Dennis Ritchie between 1969 and 1973 at AT&T Bell Lab.
  - Unix OS is written in C.
- Many other languages borrow from C,
  - Including C#, D, Java, JavaScript, Limbo, LPC, Objective-C, Perl, PHP, Python, Verilog (hardware description language) and Unix's C shell.

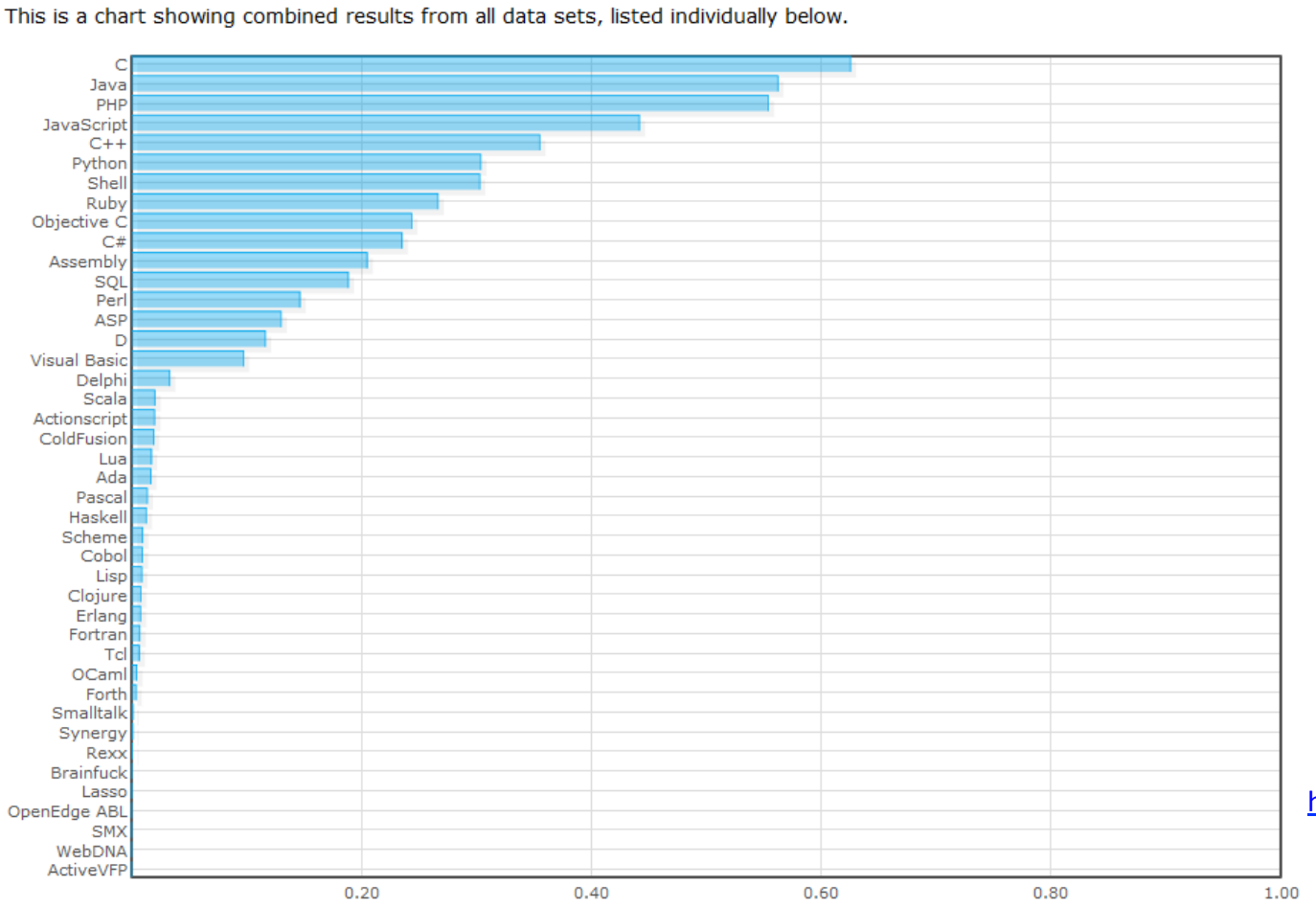
Why learn C programming?

- C is cross-platform.
  - A standards-compliant and portably written C program can be compiled for a very wide variety of computer platforms.
    - from embedded microcontrollers to supercomputers.
- C is versatile.
  - Suitable for embedded systems, device drivers, OS kernels, small command-line utilities, large desktop applications, DBMS's, implementing other programming languages, and so on.

Why learn C programming?

- C is fast.
  - Widely used to develop systems that demand performance.
    - OS, embedded and real-time systems, communication systems.
  - Most C implementations compile directly to machine code, and the programmer has full power over what happens at the machine level.
  - Even part of Java Virtual Machine is implemented in C/C++.

Why learn C programming?



http://langpop.com/

Intro to C

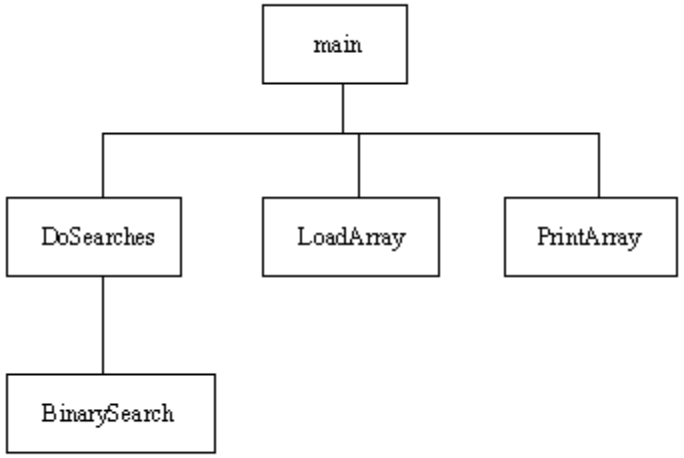
- C is an imperative (procedural) language.
  - Programmers provide steps about how to solve a problem.
  - Programming task is broken into a collection of variables, data structures, and subroutines (functions).
- Whereas, in OO programming like Java,
  - We break down a programming task into objects that expose behavior and data using interfaces.

Typical C Program Design

```
int main() // this is only pseudo code to illustrate
{
    ……
    data d = LoadArray(filename);
    result = DoSearches(d);
    PrintArray(Result);
}

result DoSearches(data d)
{
        ……
        BinarySearch(d);
        ……
}
```

Data are stored in data structures, such as arrays or linked lists. Data structures are passed into different functions as parameters to get processed.



Example Structure Chart, the main program has to call three different functions.

Picture from http://cis.stvincent.edu/html/tutorials/swd/arrays/arrays.html

C Standard Library

- C programs consist of pieces called **functions**.
- You can program all the functions you need to form a C program,
- But most C programmers take advantage of the rich collection of existing functions called the ***C Standard Library***.
  - Reference to C standard Library

  http://www.tutorialspoint.com/c_standard_library/

C Standard Library

- When programming in C you'll typically use the following building blocks:
  - C Standard Library functions.
  - Functions you create yourself.
  - Functions other people (whom you trust) have created and made available to you.
    - Bundled as Libraries.

Typical C Program Development Environment

- In this class we use Linux/Unix-based C system.
  - gcc compiler, make tool and gdb debug tool.
  - Already set up on cslinux machine.
- C programs typically go through six phases to be executed,
  - edit, preprocess, compile, link, load and execute.

Phase 1: Creating a Program

- Phase 1 consits of editing a file
- Three editors widely used
  - emacs, vi and nano
- Save you program with .c extension on hard disk.

hello.c

```
#include <stdio.h>
int main()
{
   printf("hello world!\n");
   return 0;
}
```

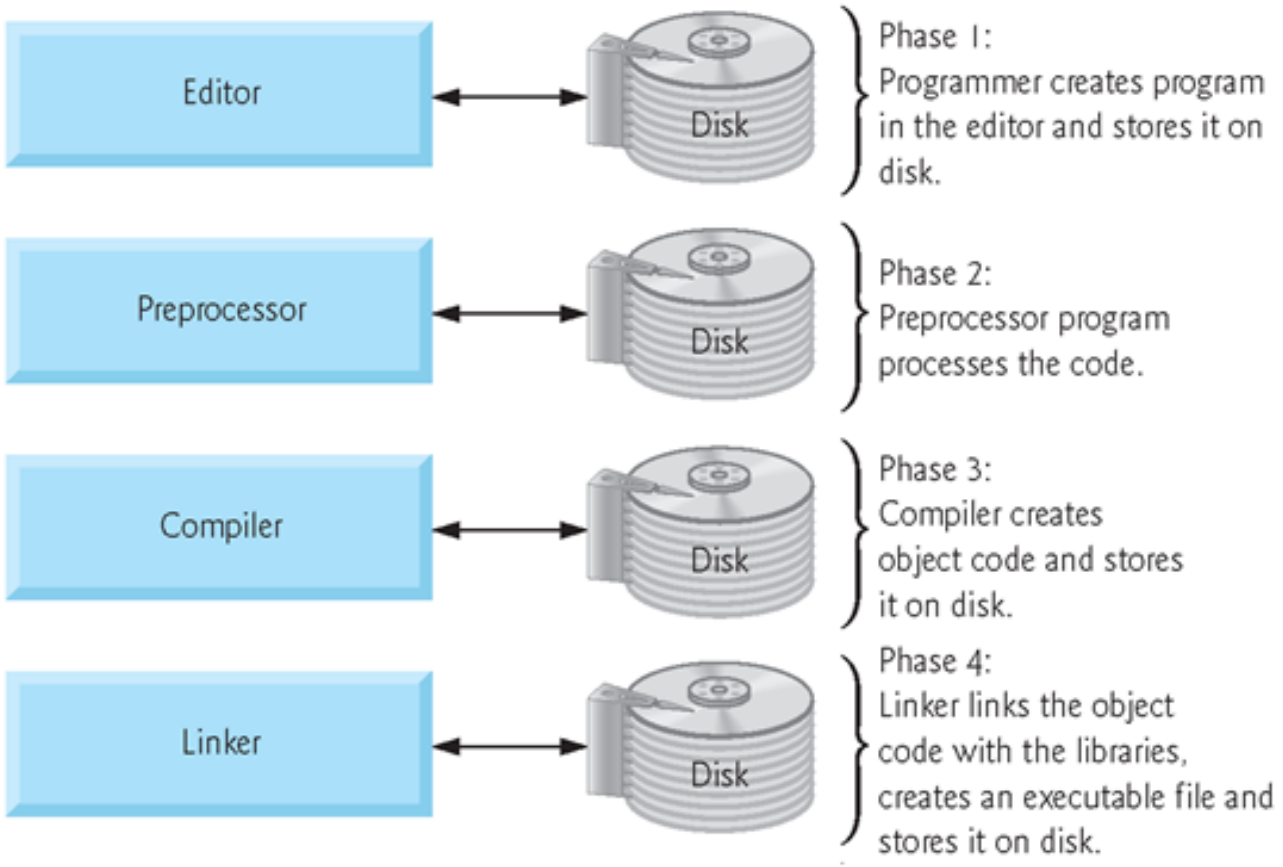
Phases 2 and 3: Preprocessing and Compiling a C Program

- C preprocessor obeys special commands (called preprocessor directives).
  - These commands do string substitutions and preprocessing.
- The compiler translates the C program into machine language-code
  - also referred to as object code, with .o extension.
  - A **syntax error** occurs when the compiler cannot recognize a statement.
    - Also called a **compile error**

Phases 4 Linking

- C programs typically contain references to functions defined elsewhere,
  - such as printf() in standard library.
- The object code produced by the C compiler typically contains “holes” due to these missing parts.
- A linker links the object code with the code for the missing functions to produce an executable image.
- **We will have more on this topic later.**

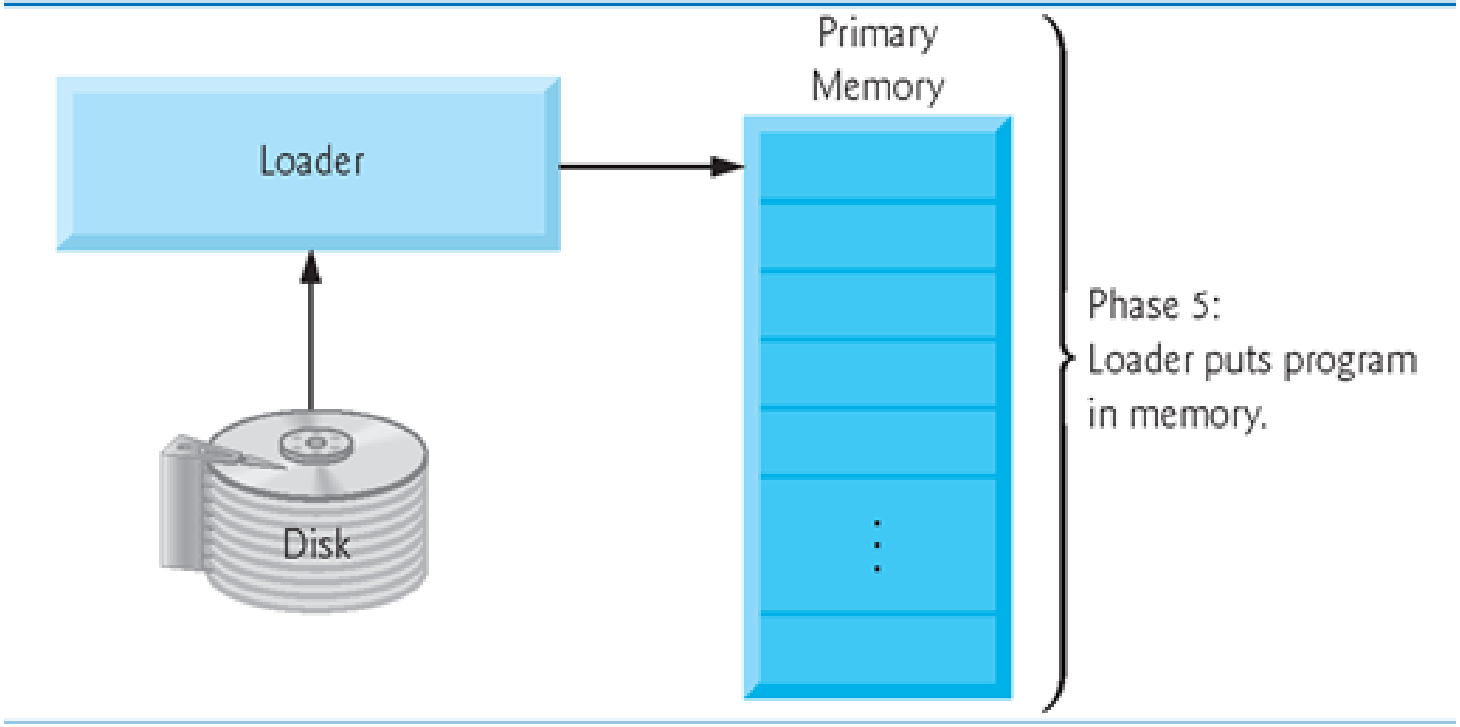
Phases 1-4 Diagram



Phase 5 Loading

- Before a program can be executed, the program must first be placed in memory.
- This is done by the loader, which takes the executable image from disk and transfers it to memory.
- Additional components from shared libraries that support the program are also loaded.

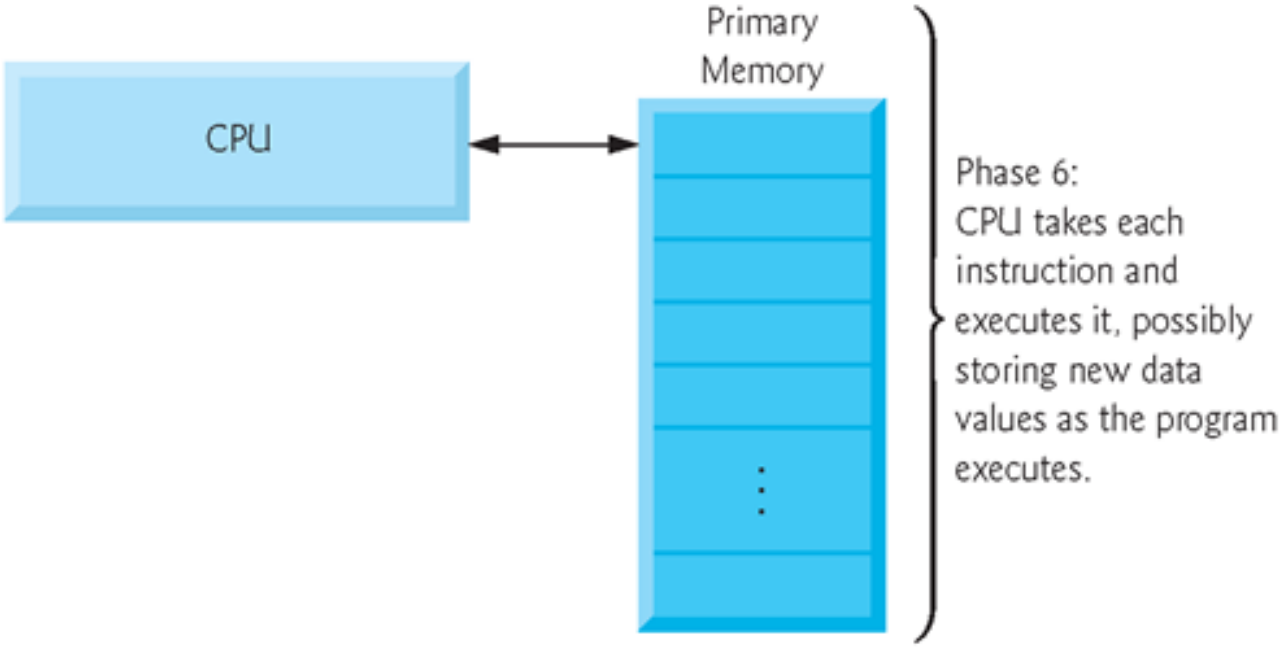
Phase 5 Loading



Phase 6 Execution

- Finally, the computer, under the control of its CPU, executes the program one instruction at a time.
- To load and execute the program on a Linux system,
  - type ./a.out at the Linux prompt and press Enter.
  - a.out is the executable produced by gcc tools.

Phase 6 Execution



Compile and Run with GCC

- Our cslinux lab has already set up gcc for you.
  - Save the file as hello.c
- To compile:

  gcc hello.c –o hello

  This command performs preprocessing, compilation and linking.(phase 2, 3 and 4)
- To Run,

  ./hello

  This command performs loading and execution.(phase 5 and 6)

hello.c

```
#include <stdio.h>
int main()
{
   printf("hello world!\n");
   return 0;
}
```

To compile:

gcc hello.c –o hello

To Run:

./hello

Summary

- Why to learn C?
- What is C?
- Generic phases to execute a C program
- Compile and Run with GCC on Linux