Summer School

on

Introduction to Programming

through

Python and C

Organizers

Pralay Mitra & Soumyajit Dey

Instructors

Goutam Biswas, Ranita Biswas, Pritam Prasun

Technical Support

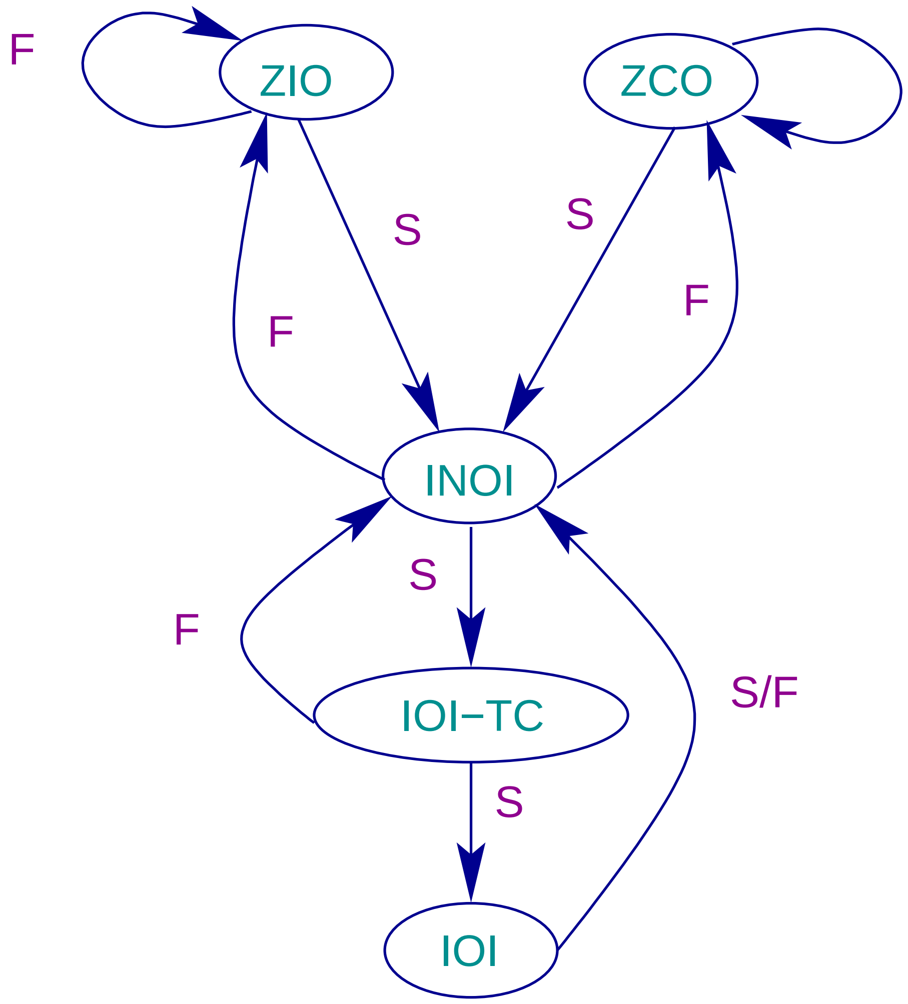
Atanu Mity & Bishnu Paria

Why Computer Programming?

- Computer is used almost everywhere.
- Computer Programming is needed in every branch of education.
- Preparation for National Informatics Olympiad.

INOI

Indian National Olympiad in Informatics


ZIO
ZCO
INOI
IOI−TC
IOI
F
F
S
S
F
F
S
F
S/F
S

ZIO and ZCO

- ZIO: Zonal Informatics Olympiad - last but one Saturday of November (e.g. 21st November, 2015). Problem solving.
- ZCO: Zonal Computing Olympiad - last Saturday of November (e.g. 28th November, 2015). Programming in C, C++, or Pascal.

http://www.iarcs.org.in/inoi/

INOI and IOI-TC

- INOI: Indian National Olympiad in Informatics - people selected through ZIO or ZCO can take this. Programming in C, C++, or Pascal.
- IOI-TC: Team selection camp for International Olympiad in Informatics. People selected through INOI.

`http://www.iarcs.org.in/inoi/`

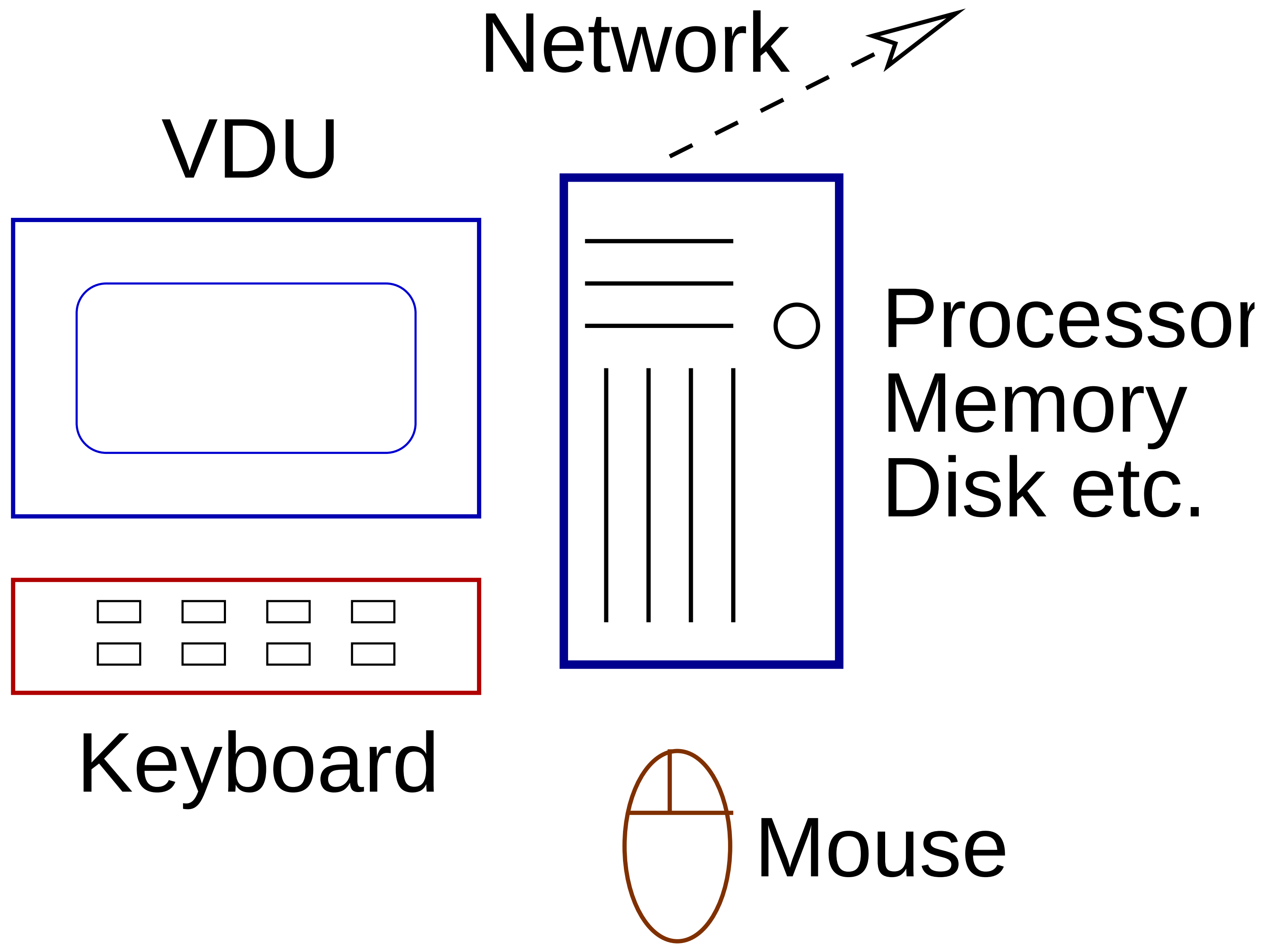
IOI

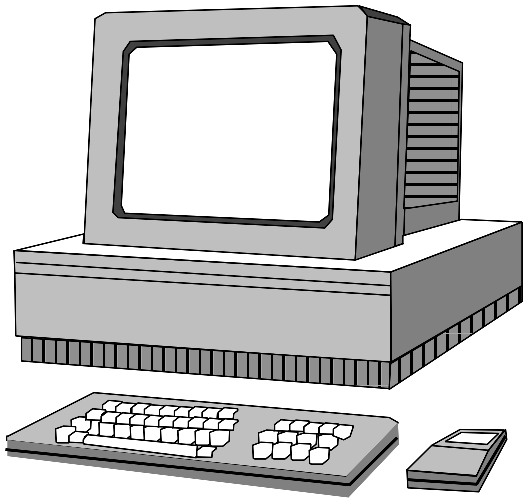
Different countries participate in IOI (International Olympiad in Informatics) with 4-member teams.

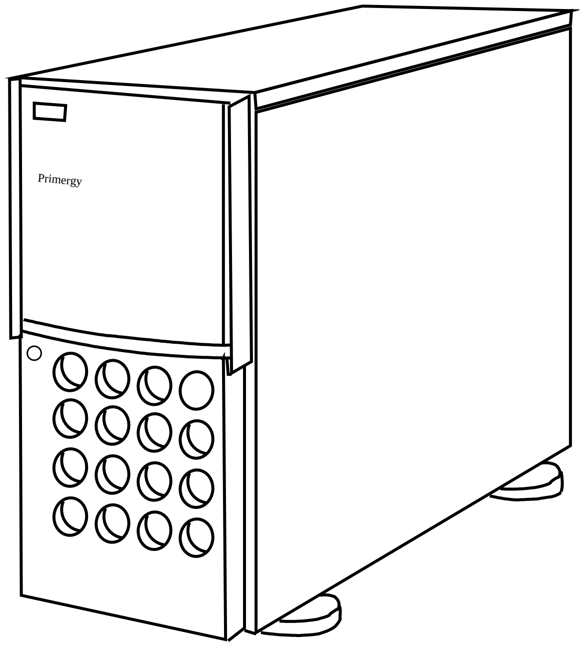
`http://www.ioinformatics.org/index.shtml`

ioi
2012
ITALY

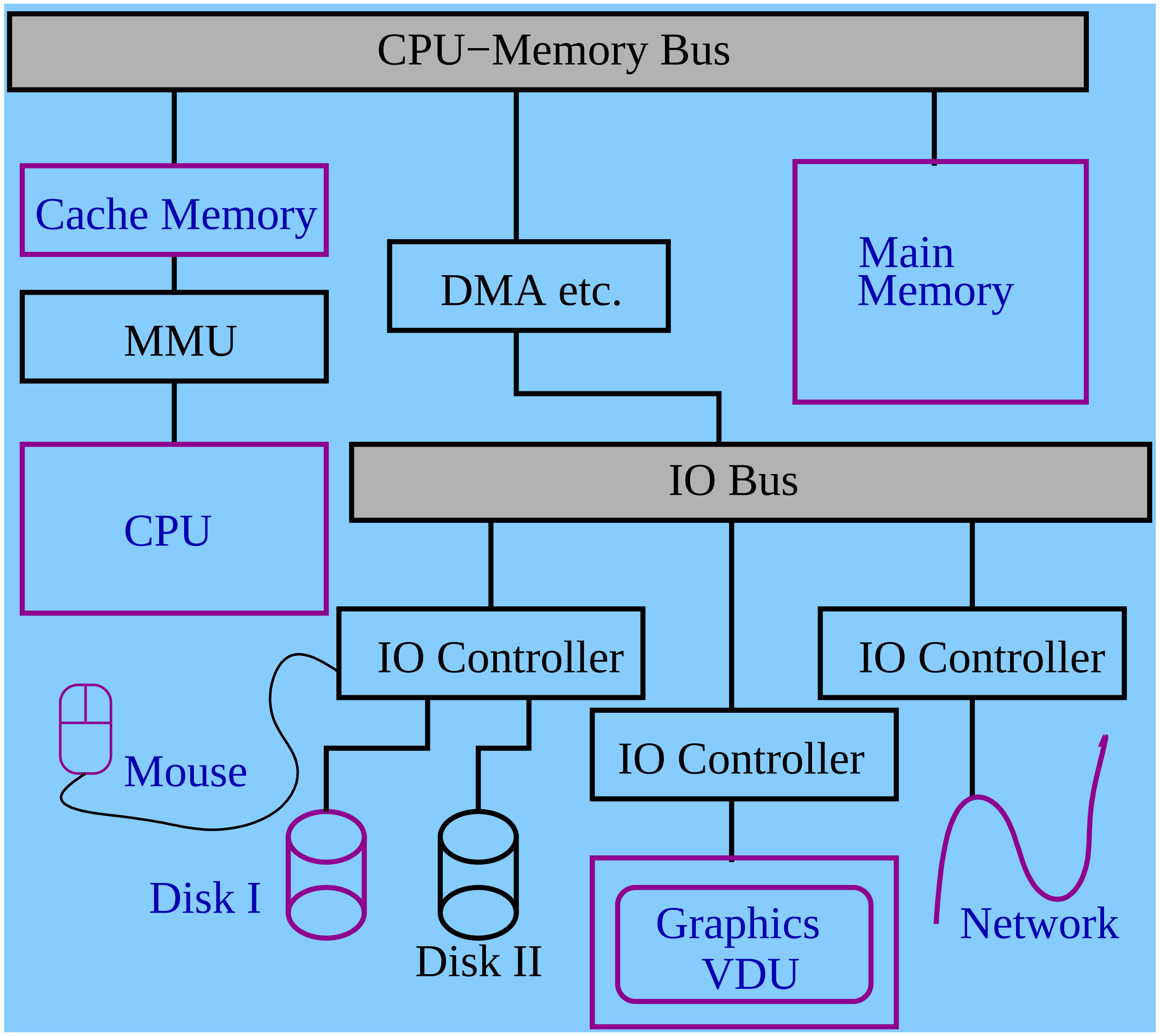
Computer System: *An Overview*


Network
VDU
Processor
Memory
Disk etc.
Keyboard
Mouse




Primergy

oxygen


CPU−Memory Bus
Cache Memory
DMA etc.
Main Memory
MMU
IO Bus
CPU
IO Controller
IO Controller
IO Controller
Mouse
Disk I
Disk II
Graphics VDU
Network

Types of Digital Computers

- General-purpose computers.
- Special-purpose computers - in washing machine, refrigerator, music system, car, mobile phone, camera, toy, instrument and there is no end.

Stored Program Computer

- A computer is used to process data.
- Computer cannot do anything on its own.
- It is given a sequence of instructions, called a program, to process any data.
- Both program and data are stored in the computer memory.

Computer Models

von Neumann Model: A computer model where both data and program are stored in the same memory is called von Neumann architecture[a]

Harvard Model: Data and program are stored in different memory in the Harvard architecture[b]

[a]The first draft of a report on the EDVAC at the Moore School of Electrical Engineering, by John von Neumann, and the design of ENIAC by J. P. Eckert and John W. Mauchly at the University of Pennsylvania.

[b]Harvard Mark I electro mechanical computer stored instructions on punched tape and data in electro-mechanical counters.

CPU Instruction Set

The set of instructions that the CPU of a computer understands is called its machine instructions.

Machine Language Program

A finite sequence of machine instructions is called a machine language program. It is stored in the main memory for execution.

Memory Location & Address

Program and data during execution are stored in the main memory.
Memory is divided into equal size blocks[a] called memory locations.
Each memory location has a unique address.
The CPU access a location by sending its address to the memory subsystem.

[a]Typical size of each block is 1, 2, 4, or 8 bytes. One byte consists of eight binary digit (8-bit).

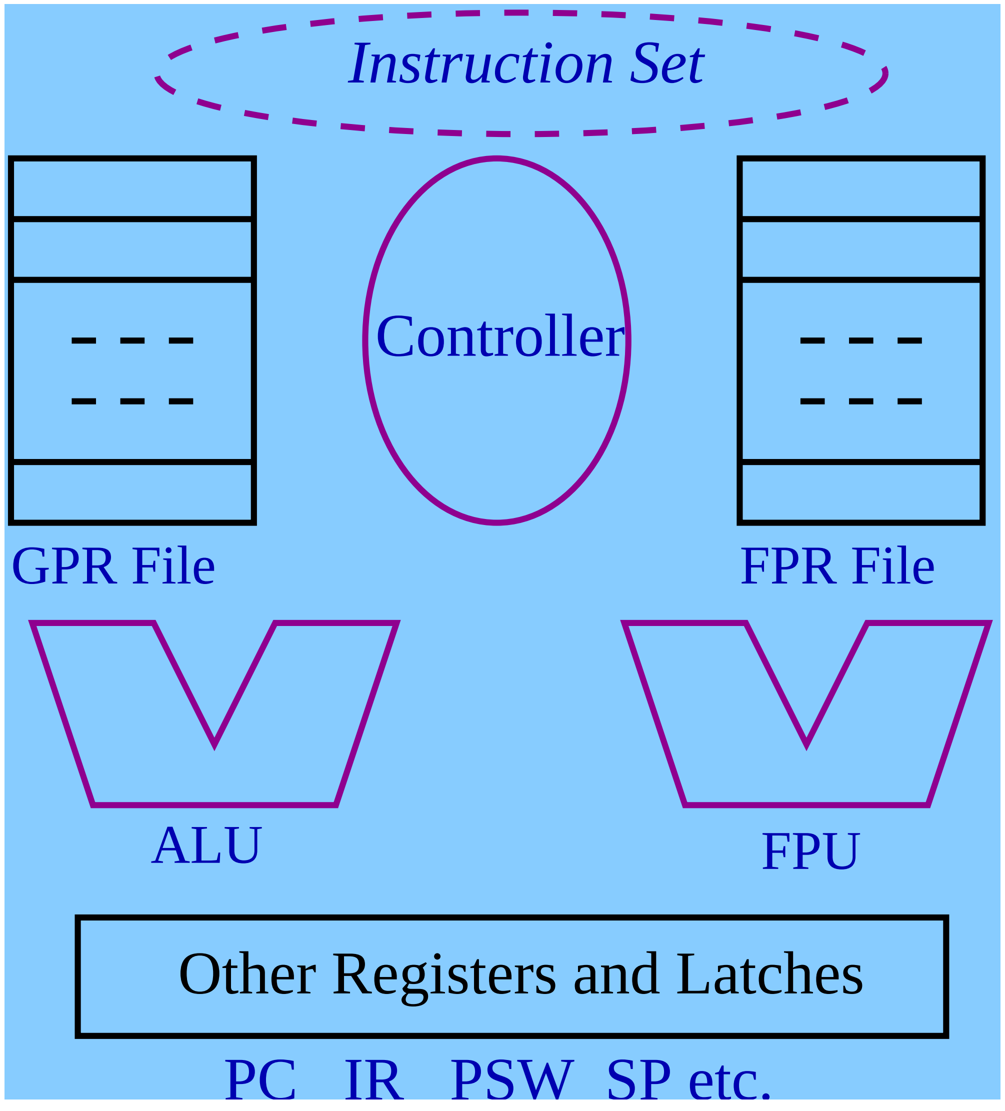
Program Execution

Fetch: The CPU fetches the next instruction from the memory and decodes (analyzes) it.

Execute: Action is taken according to the requirement of the instruction. It may be necessary to fetch data from the memory, or to do some arithmetic operation etc.

Continuation of this cycle is called program execution.

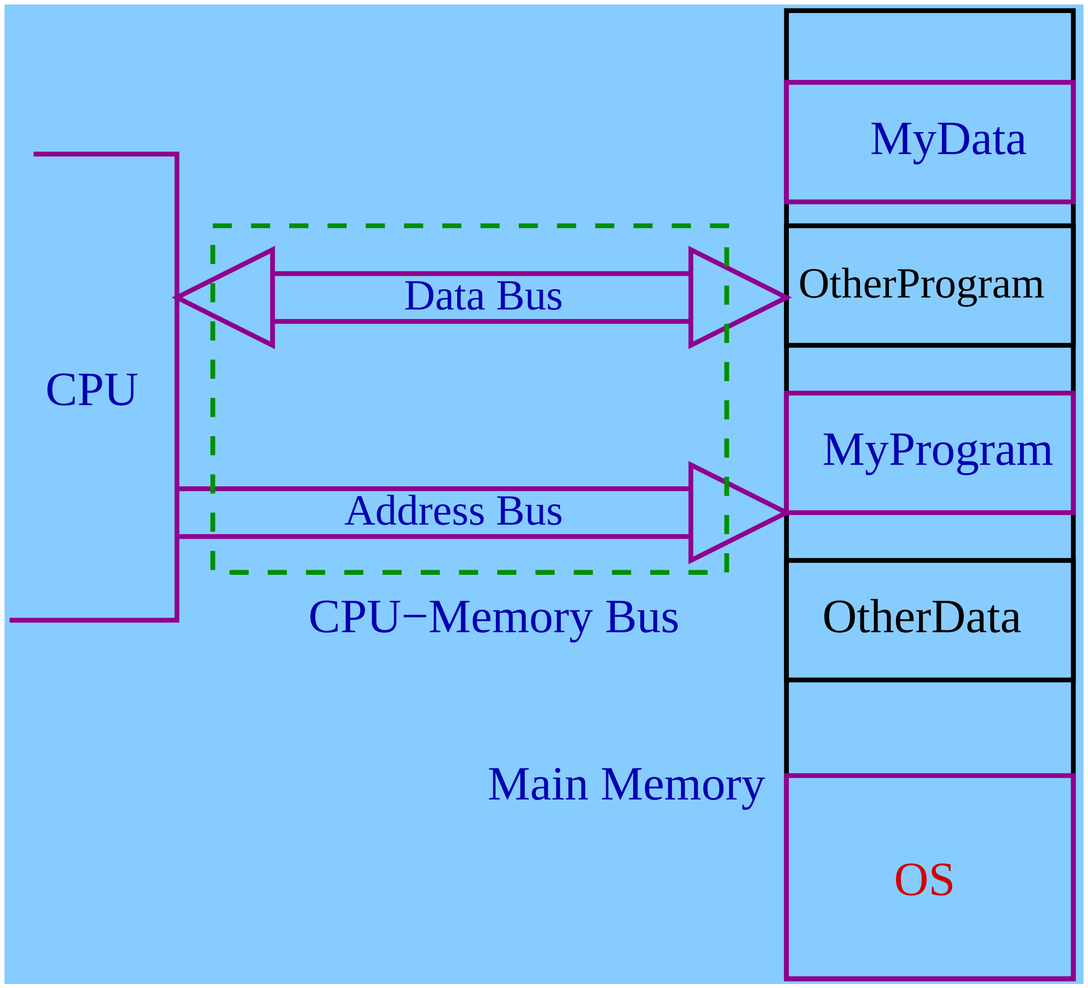
Central Processing Unit (CPU)


Instruction Set
Controller
GPR File
FPR File
ALU
FPU
Other Registers and Latches
PC IR PSW SP etc.

Central Processing Unit (CPU)

- GPRs: general purpose registers
- FPRs: floating-point registers
- PC or IP: program counter or instruction pointer
- PSW: program status word
- IR: instruction register

Program and Data in the Main Memory



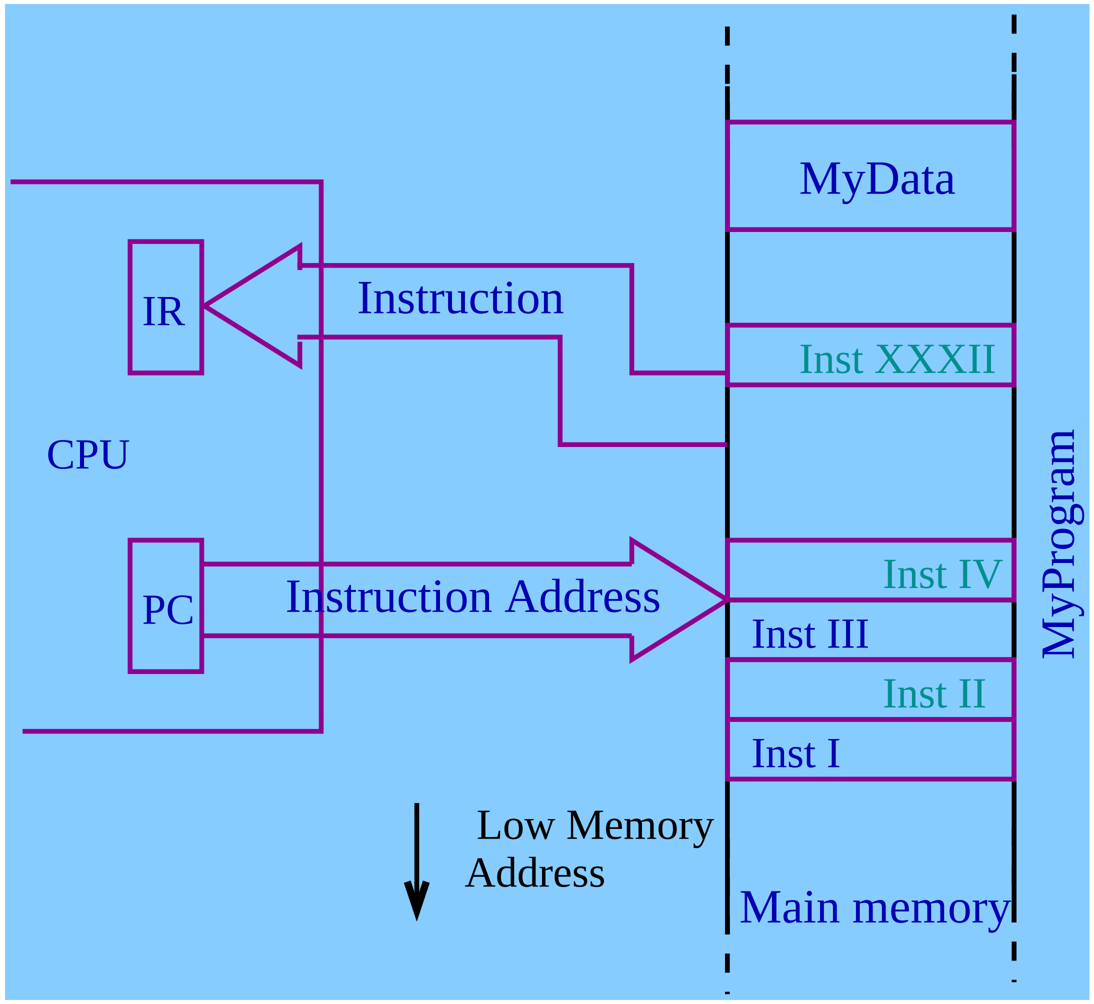
Program Execution

Instruction Fetch

The address of the next instruction is sent on the address bus from a special register of the CPU, called the program counter (PC) or the instruction pointer (IP).

Instruction Fetch

The content of the addressed location (instruction in this case) is read by the memory subsystem and sent to the CPU on the data bus.
The CPU saves the instruction in a special register called instruction register (IR).


MyData
IR
Instruction
Inst XXXII
CPU
MyProgram
PC
Instruction Address
Inst IV
Inst III
Inst II
Inst I
Low Memory Address
Main memory

Instruction Decoding & Execution

Once the instruction is available in the instruction register (IR), the CPU hardware (control circuit) decodes it, and takes a sequence of actions. The action sequence depends on the instruction.

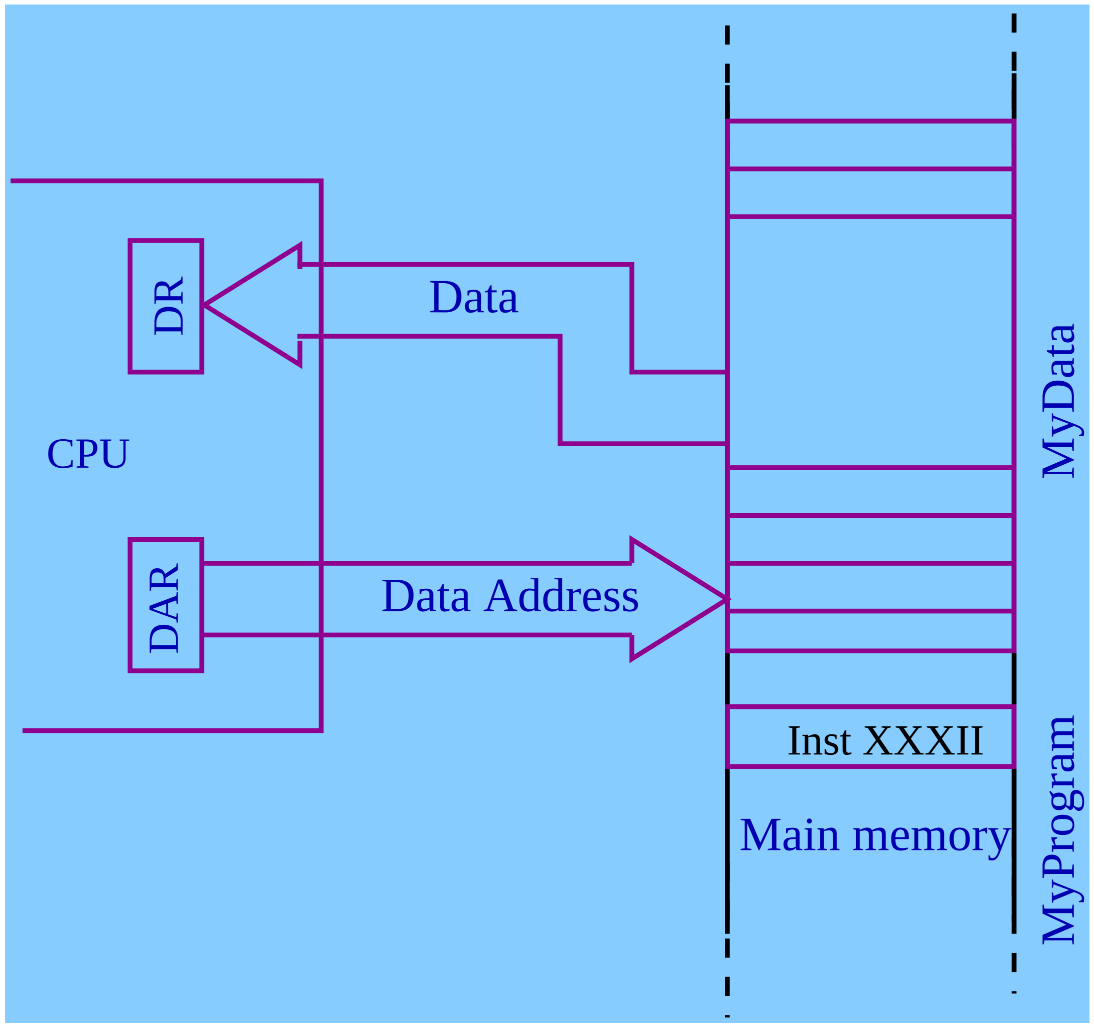
Data Fetch

If the instruction requires data from the memory, the CPU fetches it by sending the address of the data[a] on the address bus. Memory subsystem sends the data on the data bus.

[a]Data address is already available in the CPU either as a part of the instruction (IR) or in some other CPU register.

Data Fetch

The CPU may save the data in some internal register and use it for the operation.


DR
Data
CPU
DAR
Data Address
MyData
Inst XXXII
Main memory
MyProgram

Result of an Instruction

The result of an instruction execution may be stored in an internal register of the CPU or in the memory.

Memory Write

The CPU sends the address of the memory location and the data to write, on the address and the data buses respectively. It also sends a signal for memory-write.

With a Grain of Salt

The actual process is more complex

A Machine Instruction of Pentium

1000 0011 1110 1100 0000 1000

There is a register called esp in the CPU[a] of Pentium, that can store an integer value. This instruction subtracts 8 from the data stored in esp.

`esp ← esp - 8`

[a]It is a CPU and some more thing fabricated on silicon by Intel.

Binary World

World inside a computer is mostly binary - its alphabet has only two symbols 0 and 1.

But a language in binary may cause nightmare to human being.

The Charge Of The Light Brigade

“Theirs not to make reply, Theirs not to reason why, Theirs but to do & die”

The same text in binary inside a computer looks as follows

01010100 01101000 01100101 01101001 01110010 $\cdots\cdots$ *00100000 01100100 01101001 01100101 00001010*

Assembly Language Instruction

Binary machine language instructions are given human understandable symbolic representations called assembly language instructions.

1000 0011 1110 1100 0000 1000 ⇒ sub 8, esp

Assembly Language Program

A sequence of assembly language instructions forms an assembly language program.

Assembly Language to Machine Language

- The CPU does not understand assembly language instructions.
- A program is written to translate an assembly language program to the equivalent machine language program.
- This program is known as assembler.

High-Level Languages

It is also tedious to write large program in assembly language.

Moreover an assembly language program depends on a particular CPU, and also on the computing environment.

High-Level Languages

More human understandable programming languages were designed for the ease of writing program and also to make program more system independent. Following are a few names of such languages:

Fortran, Basic, Lisp, Cobol, Algol, PL/I, Pascal, SML, Prolog, C, C++, Java, Haskell, OCAML, Ada, $C^{\#}$, Perl, Python, etc.

Translation

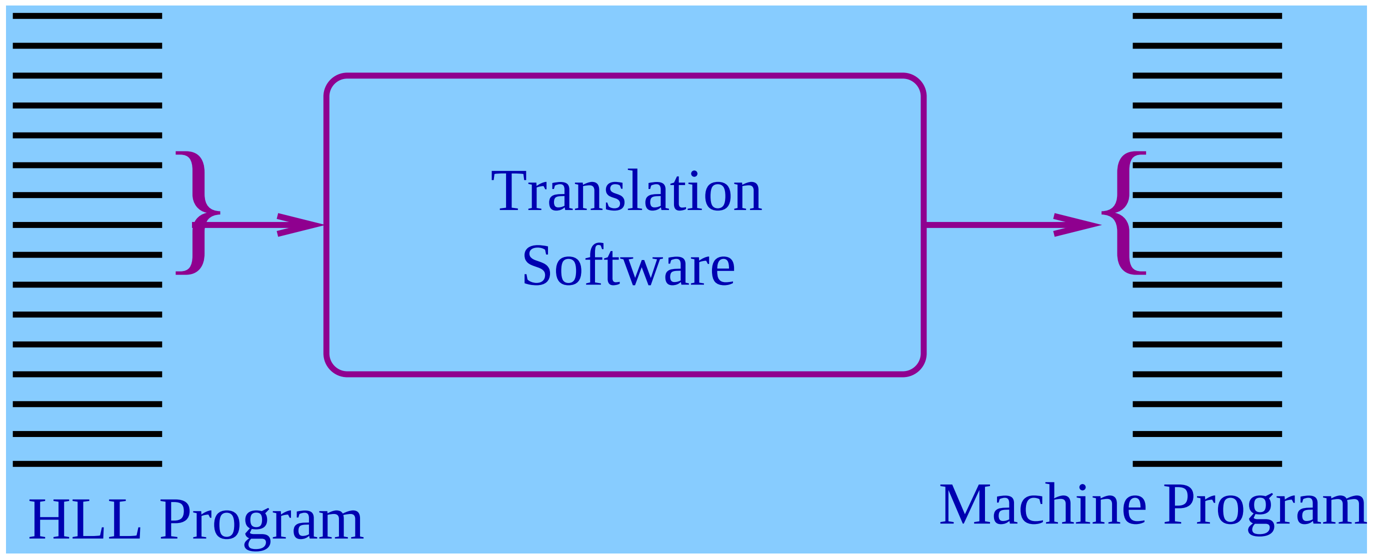
A program in a high-level language cannot directly control the CPU.
It is translated to an assembly language or to a machine language program.

Translation

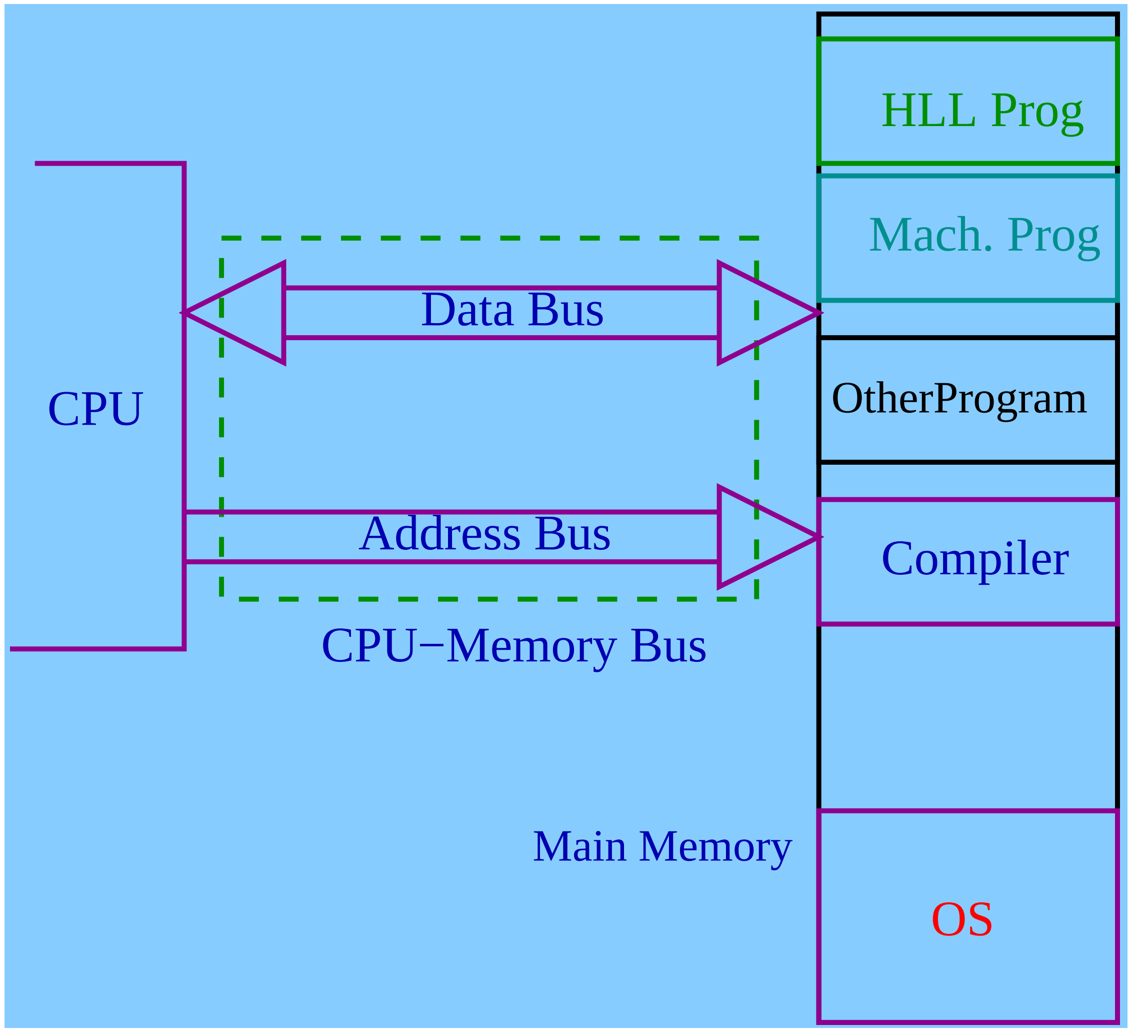
Two different types of translation schemes are generally used. One is called compilation and the other is called interpretation.
The software that *translates* a high-level program to an assembly language or a machine language program is called a compiler.
In an interpreter the translation and the execution goes hand-in-hand[a].

[a]Some high-level languages are more suitable for compilation and some other languages are suitable for interpretation. But theoretically any language can be compiled or interpreted.

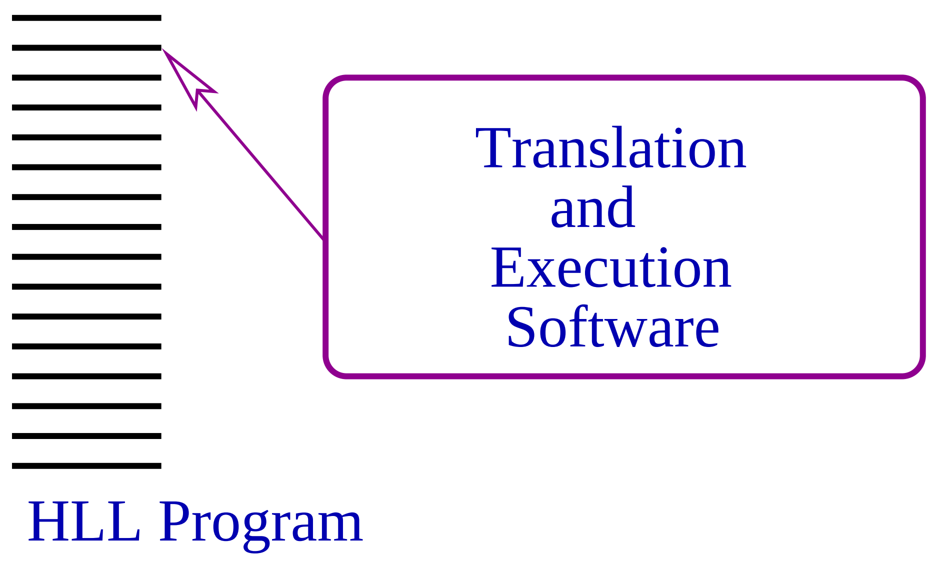
Compiler: *A Software Translator*



Compiler: *A Software Translator(cont.)*



Interpreter: *A Software Translator*



Note

We shall see how a compiler translates a C program and how a Python program is interpreted.

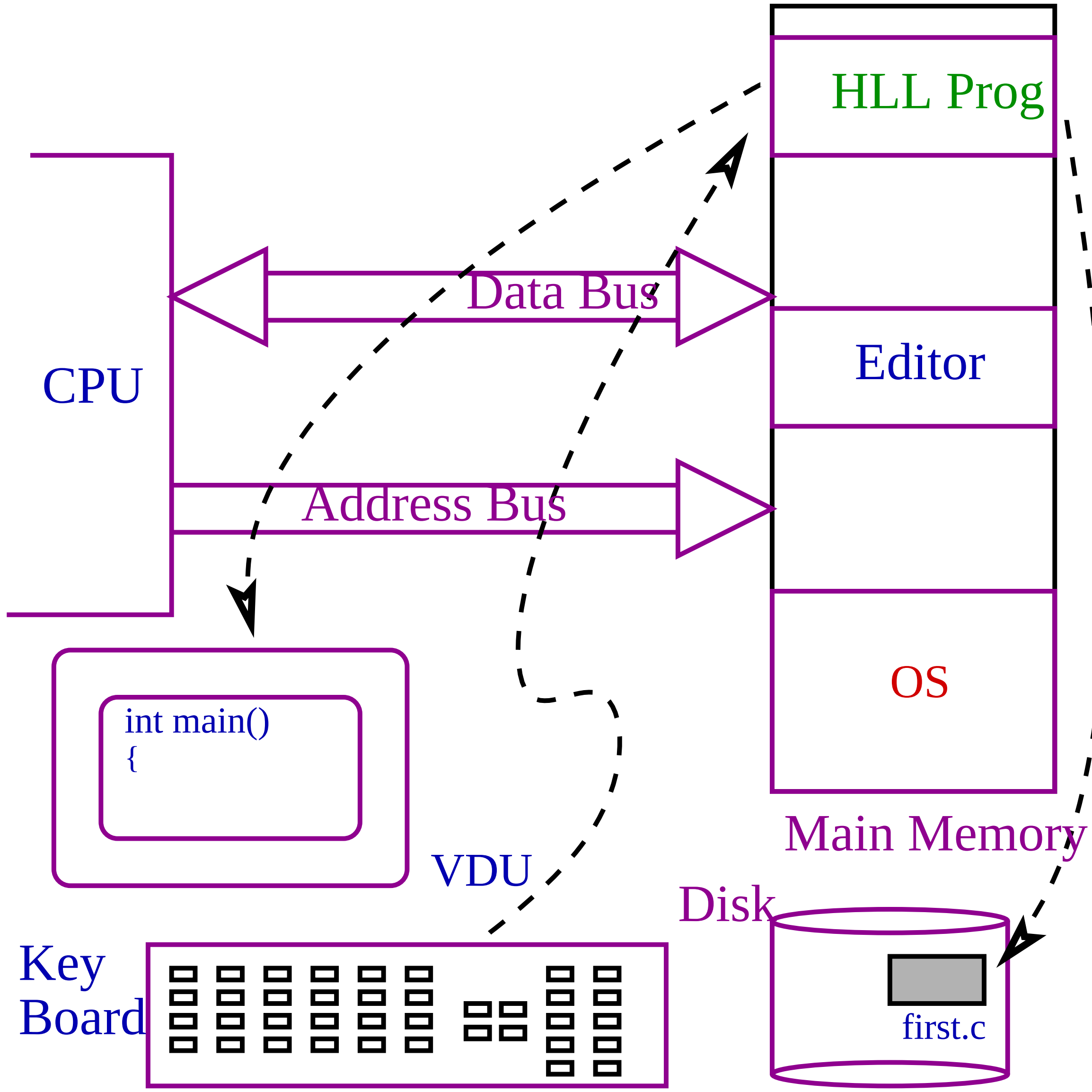
Writing and Storing a HLL Program

It is necessary to get some facility to write and store a high-level language program[a] in a computer. This is done with the help of a software called an editor e.g. gedit, emacs, vim etc.

[a]An assembly language program can also be written and saved in a similar manner.

Writing and Storing a HLL Program

These software provide facility to write/edit a text and store it in the hard disk as a named object called a file.


HLL Prog
Data Bus
CPU
Editor
Address Bus
OS
int main()
{
Main Memory
VDU
Disk
Key
Board
first.c

Operating System

A computer system would have been very difficult to use unless a master program called an operating system (OS) is running on it to present a more user friendly view.
It helps user to use the available system resources with ease. It also 'protects' one user-data from another user (and also itself).

Operating System

We shall work on an *OS* called Linux[a].

[a]The initial version of this OS was designed by Linus Torvalds in 1991. But in the current versions there is large contribution from GNU started by Richard Stallman in 1984 and may well be called GNU/Linux.

Command Interpreter

There are other system software (utilities) that are essential for the use of a computer system. One most visible and important software is the command interpreter e.g. bash. We shall talk more about it afterward.

Our Programming Environment

- PC with Intel Core 2 Duo Processor[a].
- Operating System, Linux (redhat/Mandriva/Ubuntu $\cdots$).
- Command interpreter, bash (Bourne-again shell).
- Editor, gedit.

[a]Core micro-architecture, Dual core, 2.53 GHz CPU clock, up to 6 MB L2 cache, 1.066 GHz bus etc.

Python Programming Environment

- Python is a general purpose high-level programming language designed and first implemented by Guido van Rossum[a] at CWI (Centrum Wiskunde & Informatica, Netherlands) in the late 1980s (1989-1991).
- *Python Software Foundation.*

[a]Van Rossum is Python's principal author, and his continuing central role in deciding the direction of Python is reflected in the title given to him by the Python community, Benevolent Dictator for Life (BDFL) - from Wikipedia.

Our C Programming Environment

- The C language was designed by Dennis Ritchie and Brian Kernighan in early 1970s. Subsequently it has gone through several standardizations.
- Compiler, GNU gcc for C language[a].
- Assembler, as.
- Linker (link-editor), ld.

[a] gcc is actually a C++ compiler.

Python Interpreter

```
$ python
Python 2.6.4 (r264:75706, Oct 27 ....)
[GCC 4.4.1] on linux2
Type "help", ....
>>> quit()
$
```

First Python Program

```
$ python
......
>>> print 'My first Python program'
My first Python program
>>>
```

First C Program

```
#include <stdio.h>
int main()
{
    printf("The First C Program\n");
    return 0;
} // first.c
```

Write and Execute on Linux OS

1. Use an editor e.g. ‘gedit’.
   - Run (execute) the editor program: `$ gedit &`.
   - Key-in the Python program text.
   - Save the program as a named file e.g. ‘first.py’.
2. Run python interpreter with ‘first.py’ as the argument. `$ python first.py`.

3. If there is an error, go back to the editor and fix it.

Write, Compile and Execute on Linux OS

1. Use an editor e.g. 'gedit'.
   - Run (execute) the editor program.
     `$ gedit &`.
   - Key-in the C program text.
   - Save the program as a named file e.g. 'first.c'.
2. Compile the C program to the executable file '`a.out`'. `$ cc -Wall first.c`.

3. If there is an error, go back to the editor and fix it; otherwise run the '`a.out`' file and get the output.

Typical Commands Are

- `$ gedit first.c &`
- `$ cc -Wall first.c`
- `$ ./a.out`

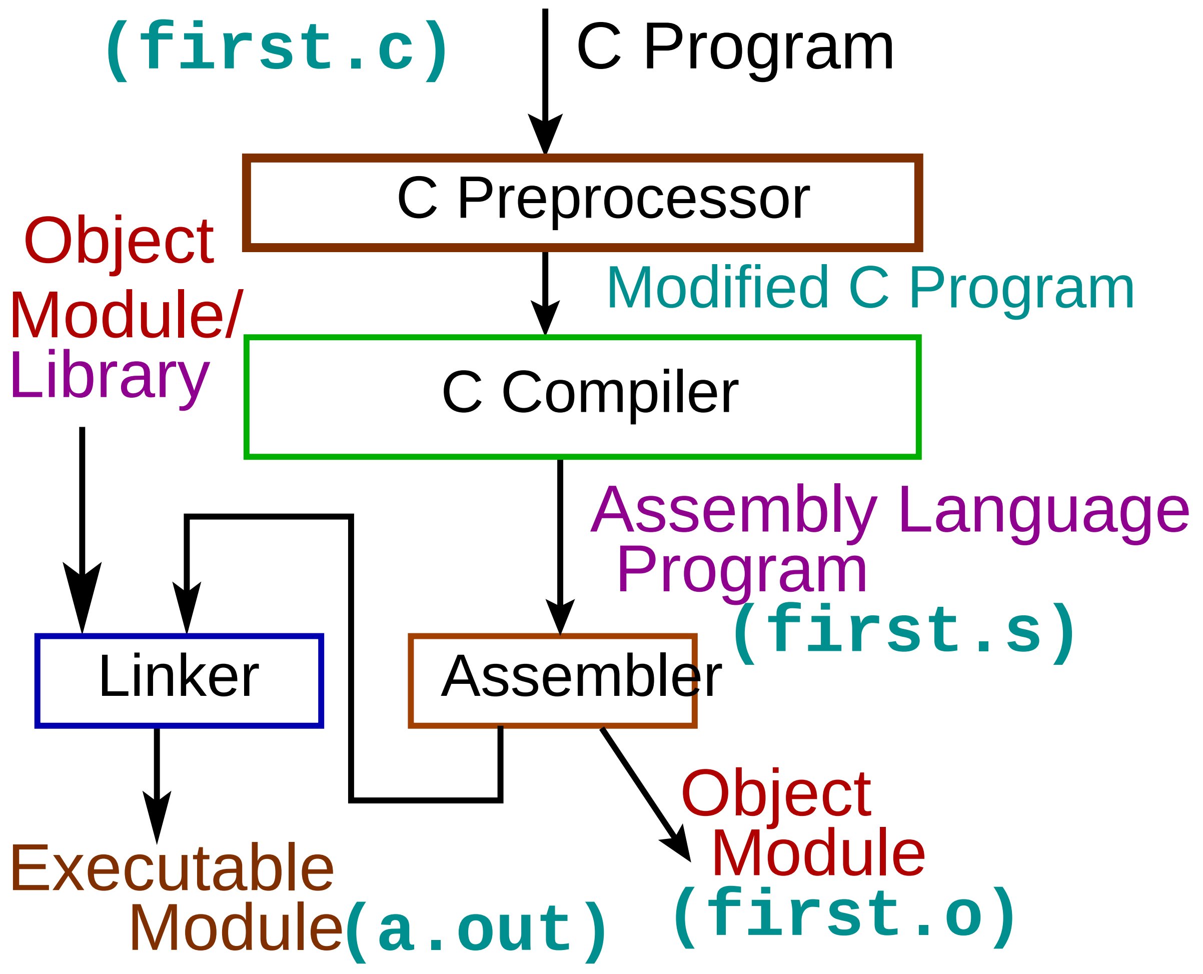
```
My first program
```

Note

- The compiler creates the executable file[a] 'a.out' (assembler output) from the C program file 'first.c'.

[a]It contains the machine code and some other data.

CPP → Compiler → Assembler → Linker



Example - II

Write a C program that reads a positive integer n and finds the sum $1 + \cdots + n = \frac{n(n+1)}{2}$.

```
#include <stdio.h>
int main()
{
    int n;

    printf("Enter a +ve integer: ");
    scanf("%d", &n);
    if(n <= 0)
       printf("Incorrect data: %d\n", n);
    else printf("1+ ... +%d = %d\n",n,n*(n+1)/2);
    return 0;
} // second.c
```

Example - IIa

Write a Python program that reads n and computes the sum $1 + \cdots + n = \frac{n(n+1)}{2}$.

Python Program File

A Python program file starts with the following line:

```
#! /usr/bin/python
```

This is the path of the Python interpreter.

second.py

```
#! /usr/bin/python
# This is the second Python program
n = input("Enter a +ve integer: ")
if n>=0:
        if n < 2: print n, '=', n*(n+1)/2
        else: print '1+...+',n,'=',n*(n+1)/2
else:
        print '-ve number'
# file: second.py
```

Execution Permission

A Python script file should have execution permission:

```
$ ls -l second.py
-rw-r--r-- 1 ... 239 ... second.py
$ chmod +x second.py
$ ls -l second.py
```

Running Python Script

```
$ python second.py
Enter a +ve integer: 10
1+...+ 10 = 55
```

or

```
$ ./second.py
Enter a +ve integer: 10
1+...+ 10 = 55
```

Laboratory Session

Let's Start At The Very Beginning

- How to Start a Computer?
- How to write a Python or C Program?
- How to Compile a C Program to an Executable Module?
- How to Run (Create a Process with) an executable file?

How to Start & Stop the Machine

1. Power On the machine (main cabinet and VDU).
2. Select the Linux option.
3. Enter the login name and then the password. The *login name* is `user` and the *password* is `user123`.
4. Open one or two windows on the VDU (use the icon).

1. One window is active at a time - use the mouse to activate an window.
2. A window may be closed by pressing Ctrl-D.

Power-off the Machine

1. Close all windows.
2. logout from the machine (use the icon).
3. Shutdown the system.
4. Power-off the VDU (power of the main cabinet will be put-off automatically).

How to Start the gedit Editor

- Select an window and enter the command gedit & .
- A gedit window is created.
- You may select the font size.
- Close the gedit window by selecting the quit drop-down menu of the File menu.

Write, Compile and Execute a Program

1. If you want to write a new file, select New drop-down of File menu.
2. For an existing file, select Open of the File menu.
3. Give a file name to a new file. C program file name is abcde.c and a Python file name is abcde.py

Starting Python Interpreter

1. $ `python`
2. The interpreter comes up with the following prompt:

```
Python 2.6.4 .....
>>>
```

Activity - I

Enter the first Python program (`first.py`). Execute the Python interpreter with the program file name as argument. The file name extension is `.py`. `$python first.py`

```
My First Python Program
```

Activity - Ia

Enter the first C program (`first.c`).
Compile it to the executable `./a.out`. Run the executable file `./a.out`.

If There is an Error

```
#include <stdio.h>
int main()
 // Syntax error
    printf("The First C Program\n");
    return 0;
} // first.c
```

Compiler Message

```
$ cc -Wall first.c
first.c:4:  error:  syntax error before
"printf"
first.c:4:  warning:  type defaults to
‘int’ ....  ‘printf’
first.c:4:  error:  conflicting types
for ’printf’
.............
```

What to do?

1. Open the file first.c in gedit.
2. Identify and fix the errors.
3. Save the modified file (same name).
4. Compile it again.

Any time you make a change in the file, save it and compile it to get the new executable module, a.out.

Note

1. If there is no more message from the compiler, the C program is syntactically correct (well-formed).
2. An well-formed C program may have logical error. Error-free compilation does not guarantee the logical correctness of a program.

Running Python Script

```
$ python second.py
Enter a +ve integer: 10
1+...+ 10 = 55
```

Activity - II

Enter the second program (`second.py`) and interpret it. Write the corresponding C program `second.c`, compile and execute it.

Integer and Float Operators

C	Python
`+ - * / %`	`+ - * / // % **`

Activity - II

Modify the second Python program to compute $1^2 + 2^2 + 3^2 + \cdots + n^2 = \frac{n(n+1)(2n+1)}{6}$. Write the corresponding C program `sumSq.c`.

Example - III

Following python program reads a positive number r as the radius of a circle and computes its area using the formula $A = \pi r^2$.

The corresponding C program is `circleArea.c`.

Python Program

```
#! /usr/bin/python
import math                           # import math
                                      #       module
r = input("Enter the radius: ") # circleArea.py
if r>=0:
        print 'radius: ', r
        print 'circle area: ', math.pi*r*r
else: print '-ve number'
```

C Program

```
#include <stdio.h>
#include <math.h>
int main() {
    double radius, area ;

    printf("Enter the radius: ");
    scanf("%lf", &radius) ;
    area = M_PI*radius*radius ;
    printf("Circle-Area = %lf for Radius = %lf\n",
                                area, radius);
    return 0 ;
} // Area of a circle third.c
```

Example - IV

We write a Python program that reads a temperature in degree Fahrenheit and converts it to the equivalent temperature in degree Celsius. We know that $32°F \equiv 0°C$ and $212° - 32° = 180°$ of Fahrenheit scale is equivalent to $100° - 0° = 100°$ of Celsius scale. So a temperature F in the Fahrenheit scale is related to the equivalent temperature C in the Celsius scale as $\frac{F-32}{180} = \frac{C-0}{100-0}$ i.e. $C = \frac{5}{9}(F - 32)$

Python Program

```
#! /usr/bin/python
# Fahrenheit to Celsius: fahrToCel.py
f = input("Enter Temp in F: ")
print f,' F = ', 5*(f-32)/9, ' C'
```

C Program

```
#include <stdio.h>
int main(){
    float cel, fah;

    printf("Enter temp. in F: ");
    scanf("%f", &fah);
    cel = 5.0*(fah-32.0)/9.0 ;
    printf("%6.2f F = %6.2f C\n", fah, cel);
    return 0;
} // Fahrenheit to Celsius: fourth.c
```

Activity - IV

Modify the program to read a temperature in Celsius scale and convert it to the equivalent temperature in the Fahrenheit scale.

Getting the Slides

- Run Mozilla and access `http://cse.iitkgp.ac.in/~goutam`
- Follow the link `Summer School for the School Students` . . . of Summer: 2014.
- Download the correct .pdf file.

Display the Slides

- View the slides using either kpdf.
- `$ kpdf lect1.pdf &`[a]
- Select the View $\rightarrow$ Orientation to Seascape for proper view.

[a] Or, use `xpdf` or `acroread` for `.pdf` file.

Example - V

Write a Python program to read two integers and print the larger one.

Python Program

```
#! /usr/bin/python
# Larger of two: larger.py
m = input("Enter two integers: ")
n = input()         # enter in the next line
if m < n : max=n
else: max=m
print 'Max(',m,',',n,')=',max
```

C Program

```
#include <stdio.h>
int main() {
    int first, second, max ;

    printf("Enter two integers :") ;
    scanf("%d%d", &first, &second) ;
    if(first > second) max = first ;
    else max = second ;
    printf("\nMax(%d,%d) = %d\n",
           first, second, max) ;
    return 0 ;
} // Larger fifth.c
```

Another Python Program

```
#! /usr/bin/python
# Larger of two: larger1.py
x = raw_input("Enter two integers: ")
y = x.split(' ')
try:
   m, n = int(y[0]), int(y[1])
   print 'Max(',m,',',n,')=',max
except:
   print 'Not two integers'
```

Activity - V

- Enter the Python program and execute it.
- Modify the program to print the smaller one.
- Modify the program to read three integers and print the largest one.

Python Program

```
#! /usr/bin/python
# Largest among 3 data: 3largest.py
x = raw_input("Enter three integers: ")
y = x.split(' ')
try:
   m, n, p = int(y[0]), int(y[1]), int(y[2])
   if m < n: max=n
   else: max=m
   if max < p: max=p
   print 'Max:',max
except:
   print 'Not three integers'
```

Example - VI

Following Python program reads a two-digit positive integer say mn and converts it to the form nm.

Python Program

```
#! /usr/bin/python
# Larger of two: mnTonm.py
x = raw_input("Enter a 2-digit +ve integer: ")
if len(x) == 2 and x.isdigit():
   y = x[1]+x[0]
   print x,'-->',y
else: print 'Not a 2-digit integer'
```

C Program

```
#include <stdio.h>
int main()
{
    int n ;

    printf("Enter a two-digit integer: ");
    scanf("%d", &n);
    if(n < 10 || n >= 100) printf("Wrong data: %d\n", n
    else printf("%d --> %d\n", n, 10*(n%10)+n/10);
    return 0;
} // mn --> nm  ex6.c
```

Activity - VI

- Enter the program and interpret.
- Modify the program to print the sum of two digits.
- Modify the program to print the *arithmetic mean* of the two digits, $\frac{m+n}{2}$.

Python Program

```
#! /usr/bin/python
# Larger of two: arithMean1.py
x = raw_input("Enter a 2-digit +ve integer: ")
if len(x) == 2 and x.isdigit():
   y = float(x[1])+float(x[0])
   print 'Mean of digits = ', y/2
else: print 'Not a 2-digit integer'
```

Example - VII

Following Python program reads a positive integer n and then reads n numbers $a_1, a_2, \cdots, a_n$ and prints the largest among them.
Do the same thing in a C program.

Python Program

```
#! /usr/bin/python
# find largest among n data: largest1.py
#
n = input("Enter a +ve integer: ")
d=raw_input("Enter data, one per line: ")
try:
   max=int(d)
   for i in range(n-1):
       d=raw_input("Enter data, one per line: ")
       try:
          data=int(d)
          if max < data: max = data
```

```
        except: print 'Not int (2)'
    print 'Max:', max
except: print 'Not int (1)'
```

C Program

```
#include <stdio.h>
int main()
{
    int n, i;
    float data, max;

    printf("Enter a positive integer: ");
    scanf("%d", &n);
    if(n < 1) {
       printf("Wrong count: %d\n",n);
       return 0;
    }
    printf("Enter %d data:\n",n);
    scanf("%f", &max);
    for(i=2; i<=n; ++i){
        scanf("%f", &data);
        if(max < data) max = data;
    }
    printf("The largest data is: %f\n", max);

    return 0;
} // largest among n data ex7.c
```

Activity - VII

- Enter the program, compile and execute.
- Modify the program to print the smallest among n data.
- Modify the program to print the *arithmetic mean* of n data $AM = \frac{a_1+a_2+\cdots+a_n}{n}$.
- Modify the program to find the second largest among $n (\geq 1)$ data.