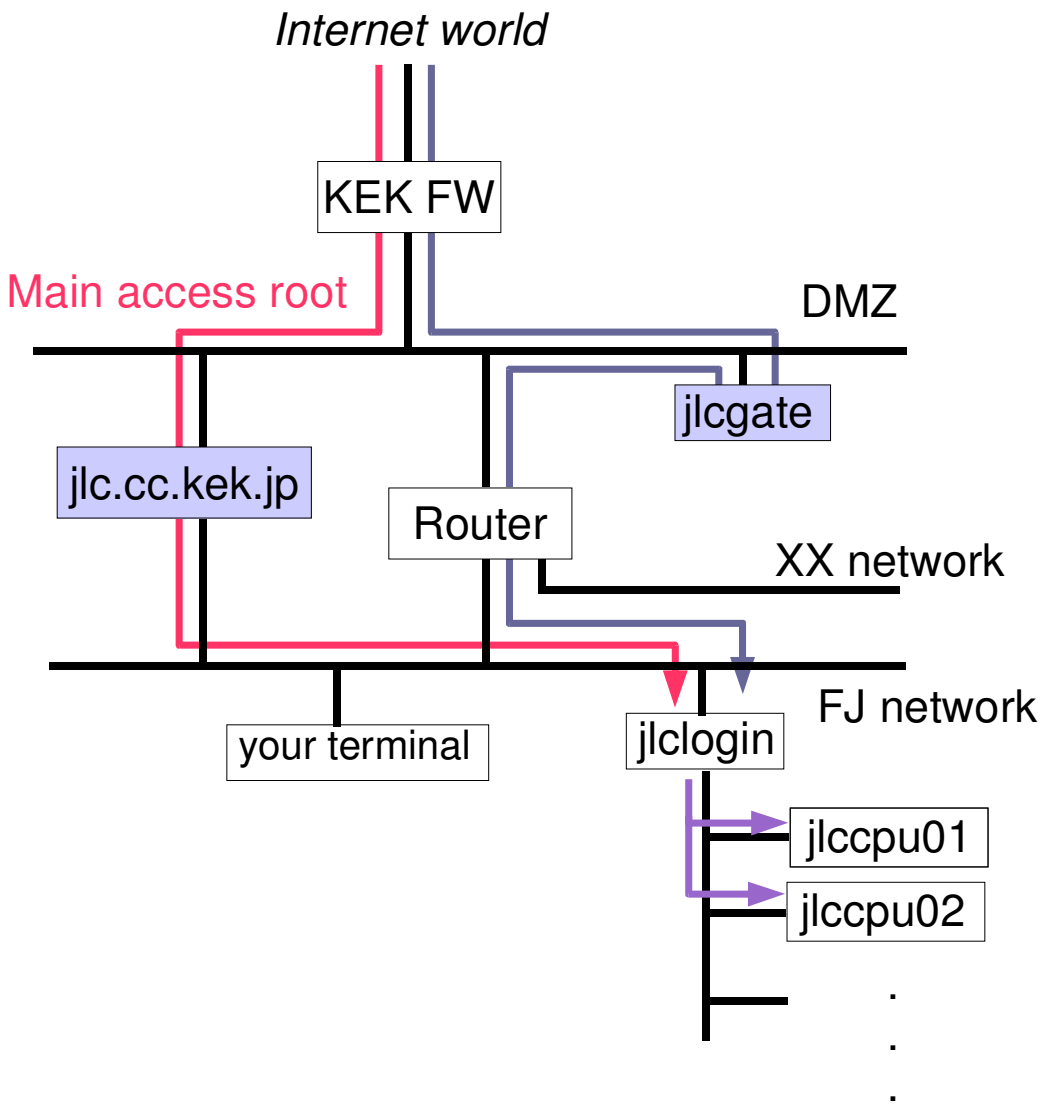
Tools for LC physics study

Akiya Miyamoto, KEK

Contents:

1. Introduction to JLC computing 9:00 - 9:30
2. Breif introduction to C++ 9:30 - 10:00
3. Introduction to ROOT 10:00 - 11:30
4. Introduction to JSF and Demo 11:30 - 12:00
5. Install JSF and other packages 13:30 -
6. Introduction to a sample program
7. Introduction to a sample analysis -15:00

Computing at KEK JLC Physics Group



Computing systems

- jlc.cc.kek.jp
  - OS: AIX
  - for CPU server, Data server, but also gateway machine for remote access by ssh
- jlcgate
  - OS: BSD
  - gateway for remote access
  - limitted use only for those who can not use jlc.cc.kek.jp
- jlclogin
  - OS: Linux (Redhat 9)
  - 2CPU server for interactive program developments
- jlccpuXX
  - OS: Linux (Redhat 9), XX=01, 02, 03, 07, 08, 09
  - For CPU heavy job

How to use jlclogin/jlccpuXX

- jlclogin:
  - to edit program, and make a test run

- Using jlccpuXX
  - submit job
    $ submit *command*
  - submit job as a detached process, save command output in *jobout*
    $ submit *command* > *jobout* 2>&1 & (bash)
    $ submit *command* >&! *jobout* & (tcsh)
  - show status of your job on cpu server
    $ showjob
  - show cpu load on each cpu server
    $ cpulookup
  - login to cpuserver
    $ rsh jlccpuXX
    - jlclogin user can login to jlccpuXX without authentification
    - use this feature to cancel jobs running on jlccpuXX
      ex. $ rsh jlccpuXX “ps -ef | grep *username*”
      $ rsh jlccpuXX “kill -15 *NNNN*”

Setup account at jlclogin

- At the first login, do

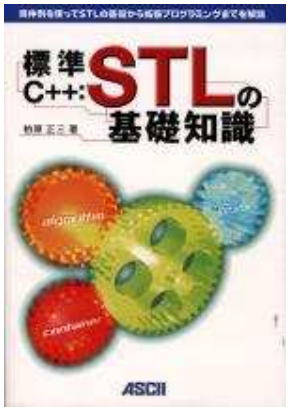
```
$ cd
$ cp ~miyamoto/ToolsLect .
$ cd ToolsLect
$ source setup.bashrc

$ root sample01.C
```

- Printer : tek3g423dbl(default), tek3g423(single-side)

```
$ lpr file.ps                  # printout
$ lpr -Ptek3g423 file.ps  # printout to tek3g423
$ a2ps  file.text              # printout text file
$ a2ps -o -  file.text > file.ps  # convert text file to ps
$ lpq                          # show print que status
$ lprm job_number      # cancel print job
```

C++ reference



1. 柏原正三著の標準C++の基礎知識シリーズ、(株)アスキー
   1)「標準C++の基礎知識」、2200円、ISBN4-7561-3121-2　お勧め
   2)「標準C++の基礎知識 実践編」、2800円、ISBN4-7561-3563-3
   3)「標準C++:STLの基礎知識」、2800円、ISBN4-7561-3804-7

2. 林晴比古 著３部作、(株)ソフトバンク・パブリッシング
   1)「新C++言語入門」シニア編 上基本機能
   2)「新C++言語入門」シニア編 下クラス機能 3600円　ISBN4-7973-1661-6
   お勧め

3. M.T.Skinner著、春木良且 訳のシリーズ、(株)インプレス発行
   1)「C++基礎講座」2400円
   2)「C++実用講座」2600円、ISBN4-8443-4647　→ 少し古い

C++ basic element

Structure of C++ source code

```
prototype_declaration
definition_of_global_variable
return_type function(arguments)
{

  function body

}
```

- Declare type of arguments and return type of functions.
- Prototype of all function must be declared before its use.

- Variable defined outside a function has a file scope: valid at all functions

```
// : comment line
/*
... : commented block
*/
```

C++: Hello world !

example.cxx

```
#include <iostream>
main() {
  cout <<  “Hello world”  << endl;
}
```

#include “file.h”
Include file.h before compile

#include <....>
Included from /usr/include

see “info cpp”

compile and run

```
$ g++ example.cxx
$ a.out
Hello world
$
```

cout << ..... << endl;
..... part is output
endl = \n
; : end of line

Basic type of variables

```
int          i=3;      // integer
float        f=3.4;   // float
double   d=4.5;  // double
char         c='a';   // 1 byte character
bool         b=true; // boolean variable

int   ia[20];       // integer array of size 20.
       ib[0] = 0;   // substitute 0 to the first element if ib array
       ib[19]=190;      //                  190 to the last element
         Note: array index is 0 to n-1
int   ka[40][5];  // equivalent to INTEGER KA(5,40) in FORTRAN
         Note: index ordering is opposite compare to FORTRAN
```

new and delete

new : dynamically allocate memory area
This area won't released unless delete command is used.
delete: release memory area allocated by new

* : pointer variable first address of array

```
int    *ib=new int[20]; // allocate integer array of size 20, dynamically
        ib[0] = 0;          // substitute 0 to the first element if ib array
        ib[19]= 190;       //                  190 to the last element
       *(ib+19)= 190;      // other way to substitute to the last element
delete   ib;      // release dynamically allocated area.
```

```
char *str="This is a pen";
```

String is always an array of char.
End of string is \00. " and ' is different

str[13]='n'
str[14]=\00

C++ minimum - 1

- Compile and link
  - ✗ g++, make, Imakefile, xmkmf -a
- C++ source program
  - ✗ semi colon (;) : end of line
  - ✗ // and /* */ : comment line and block
  - ✗ Can write a “Hello world” program
  - ✗ cpp (C pre-processor) macro
    - #include, #if, #ifdef, #define, #endif
- Basic type
  - ✗ int, float, double, bool, char, void
  - ✗ const, static
- Operator
  - ✗ +, -, *, /, %, <<, >>, | , & (no exponentiate)
  - ✗ Logical condition : <, >, ==, !=, >=, <=, &&, ||
  - ✗ Addressing operator : *, &
  - ✗ ++, +=, --, -=
  - ✗ cast operator : ( )

Example: sample01.C

```
#ifndef MY_FLAG
#define MY_FLAG
#include “my_file.h”
#else
void func() {
  int a=0;
  const int size=5;
  double b[size]={1.0,2.0,3.0,4.0,5.0};
   b[a++]=3.2;   // b[0]=3.2
   b[++a]=5.4;   // b[1]=0, b[2]=5.4;
   int x=(int)b[2];
   cout << “  b[0]=” << b[0];
   cout << “  b[1]=” << b[1];
   cout << “  b[2]=” << b[2];
   cout << “  x=” << x;
   cout << endl;
}
#endif
```

C++ minimum – 2

- 0x12(hex), 012(oct), 12(dec)
- 3.0e12 (no 3.0d12)
- Array
  - ✗ int a[20] ; a[0] to a[19]
  - ✗ int b[8][4][2]; Dimension b(2, 4, 8)
  - ✗ int *c=a; c[3]
  - ✗ &b[0][0][0] == b ;
  - ✗ char *str="This is a pen";
    - ➢ str[12]='n'; str[13]=0x00;
- if (condition) { }
  else if ( ) { }
  else { }
- int c = a == b ? 0 : 1 ;
- switch (i) {
  case 1: ..... ; break;
  case 2: ..... ; break;
  default: ..... ;
  }
- for(int i=0;i<20;i++) { }
- while (i<10) { .... }

Example: example02.C

```
#include <iomanip>
void sample02() {
   int ihex=0x12;
   int ioct=012;
   int idec=12;
   double xpi=3.12345678901234567890 ;
   cout << " ihex=" << ihex << " ioct=" << ioct;
   cout << " idec=" << idec << endl;
   cout << " def. xpi=" << xpi;
   cout << " 9 prec. " << setprecision(9) << xpi << endl;

  for(int i=0;i<50;i++){
     if( i==2 ) continue;
     if( i==4 || i == 20 ) break;
   }
}
```

C++ minimum - 3

- #include <stdio.h>
  #include <iostream>
  #include <iomanip>
- printf("format",val);
  - %d, %g, %f, %lg, %lf, %s
- scanf("format",&val);
- cout << ..... << endl;
  - cout << precision(15);
  - cout << width(15);
  - cout << scientific , fixed
- cin >> val ;
- Output to file
  - ofstream fout(file_name);
    fout << x << endl;
    fout.close();
- Input from a file
  - ifstream fin(file_name);
    fin >> i ;
    fin.close();

Example: example02.C

```
using namespace std;
void sample03()
{
  int i=10;
  float x=3.141592654;
  printf(" int i=%d  float x=%g\n",i,x);
  double y;    printf("Enter y=");
  scanf("%lg",&y);  printf("  y=%g\n",y);


  char filename[100];
  cout << "Enter filename" ;
  cin >> filename ;

  ofstream fout(filename);
  fout << " y= " << y << endl;
  fout.close();


  TDatime da;
  strstream sout;
  sout << "Today is " << da.GetMonth() ;
  sout << "-" << da.GetDay() << ends;
  TString  ts;
  ts.GetLine(sout);
  cout << ts << endl;
}
```

C++ minimum - 4

- prototype (function call)
- scope
- call by value, address, reference
  - ✗ int sub (float a);
  - ✗ int sub (float *a) or sub (float a[]) ;
  - ✗ int sub (float &a);
- enum MyID { kHeight, kWeight, kSize };
- typedef int Int_t;
- struct {
    int a;
    float b;
  } abc;

Example: example04.C

```
void sample04(){
 int i=1;
 for(int k=0;k<10;k++){
       int i=k+2;
 }
 cout << " i after loop is " << i << endl;
}


void swap_by_address(int *a, int *b){
   int tmp=*a;    *a = *b ;   *b = tmp;
}
void swap_by_reference(int &a, int &b){
   int tmp=a;   a=b;   b=tmp;
}


void swap_test(){
  int a=20;    int b=50;
  swap_by_address(&a, &b);
 cout << " a=" << a << " b=" << b << endl;
 swap_by_reference(a,b);
 cout << "a=" << a << "b=" << b << endl;
}
```

C++ minimum - Class

MyClass.h

```
class MyClass : public Basic
{
  private:
      Int_t  fData;
  public:
      MyClass::MyClass();
      MyClass::MyClass(Int_t data):fData(data) {}
      virtual ~MyClass(){}
      inline Int_t GetData(){ return fData; }
      void SetData(Int_t data=10);
};
```

- class proto-type, implementation
- private, protected, public
- inheritance
- constructor, default constructor, destructor
- default argument
- inline function
- operator over loading

MyClass.cxx

```
MyClass::MyClass()
{
    fData=20;
}
void SetData(Int_t data)
{
     fData=data;
}
```

Class の継承

■派生クラス(Derived class)は基底クラス(base class)から
1.データメンバーとメンバー関数は全て継承する。
2.コンストラクター、デストラクター、代入演算子は継承されない。
3.基底クラスの仮想関数(virtual と宣言された関数)はオーバーライトできる。

■派生クラスのコンストラクターは、基底クラスのコンストラクターを自動的に呼出し、
基底クラスのようその初期化を行う。このとき、派生クラスの初期化リストが、
1.ある場合、
初期化リストに従って、基底クラスのコンストラクターを呼び出す.
2.ない場合
基底クラスの引数無しコンストラクターを呼び出す。

```
class MyObject : public TObject
{
   protected:
        int  fData;
   public:
        MyObject(char *name, char *title): TNamed(name, title) ;
}
```

■基底クラスの型を持つポインタ変数(pb)に派生クラスへのポインターを代入できる。
delete pb;
を実行したときに、基底クラスのデストラクターが、
■virtual の時には、派生クラスのデストラクターが呼び出される。
■virtual でないときには、基底クラスのデストラクターが呼び出される。

■スコープ解決演算子::を用いると、派生クラスから基底クラスの関数を明示的に呼び出せる。

Class の継承 - 2

■呼び出し順序：
- コンストラクタ：
  1. 基底クラスのコンストラクタの呼び出し。
  2. 派生クラスのデータメンバーのコンストラクターの呼び出し。
  3. 派生クラスのコンストラクタの呼び出し。
- デストラクタ：
  1. 派生クラスのデストラクターの呼び出し。
  2. 派生クラスオブジェクトのデータメンバーのデストラクタの呼び出し
  3. 基底クラスのデストラクターの呼び出し

■仮想関数：
- 仮想関数の呼び出し元と呼び出される関数本体の結合は実行時に行われる。
- 仮想関数（基底クラスで virtual と指定された関数）が派生クラスで、
  1. 再定義されると、派生クラスの関数が呼び出される。
  2. 再定義されていないと、（派生クラス型の変数からでも）基底クラスの関数が呼び出される。

■純粋仮想関数

C++ Advanced Feature

- Operator overloading
- Template class
- Standard Template Library
  - string, vector
  - iterator

- FORTRAN interface
  - Call Fortran subroutines
  - Passing character variables
  - Access Fortran common