**Chapter 15 Notes**

**Alex Yang**

**(5/17/2021)**

1. 15.7 References and Reference Parameters (II)

* If a function returns a reference to a variable, make sure the variable is static

#include <iostream>

using std::cout;

int& getInt(){

//static int x = 4;

int x = 4; //Create warning

x++;

return x;

}

int main(void) {

int a,b;

a=getInt();

cout<<"a="<<a<<"\n";

b=getInt();

cout<<"b="<<b<<"\n";

cout << "Hello World!\n";

return 0;

}

*Notice that static variable lift-time is the same as the whole program*

1. “using” keyword in C++

#include <iostream>

using namespace std;

int main(void){

  int age = 21;

  int age = 22; //Error: redef

age=23;

  cout<<"age="<<age<<"\n";

}

#include <iostream>

using namespace std;

namespace ns1{

int age = 21;

string name="Mike";

}

namespace ns2{

int age = 22;

}

int main(void){

  cout<<"age="<<ns1::age<<"\n"; //ns1-like address of "age"

cout<<"age="<<ns2::age<<"\n";

cout<<"name="<<ns1::name<<"\n";

ns1::name="Peter";

cout<<"name="<<ns1::name<<"\n";

}

*Notice that std is like one of namespaces with more pre-defined methods, which can be checked in the link* [*https://gcc.gnu.org/onlinedocs/gcc-4.6.2/libstdc++/api/a00911\_source.html*](https://gcc.gnu.org/onlinedocs/gcc-4.6.2/libstdc++/api/a00911_source.html)

*To avoid using pre-defined methods of std namespace in your program, do NOT suggest to use*

using namespace std;

Recommend the following statement

using std::cout;

/\*The corrected program is as follows\*/

#include <iostream>

using std::cout;

using std::string;

namespace ns1{

int age = 21;

string name="Mike";

}

namespace ns2{

int age = 22;

}

int main(void){

  cout<<"age="<<ns1::age<<"\n"; //ns1-like address of "age"

cout<<"age="<<ns2::age<<"\n";

cout<<"name="<<ns1::name<<"\n";

ns1::name="Peter";

cout<<"name="<<ns1::name<<"\n";

}

1. static\_cast<newType> (variable)

* very similar for example (int) variable as C

#include <iostream>

using std::cout;

int main(void) {

float f = 3.5;

int a = f; // this is how you do in C

int b = static\_cast<int>(f);

int c = (int)f; // this is how you do in C

cout << a<<"\n";

cout << b<<"\n";

cout << c<<"\n";

}

#include <iostream>

using std::cout;

int main() {

char c = 'a';

cout<<"c="<<c<<"\n"; //Display value

cout << "adr of c:" << static\_cast<void\*>(&c) <<"\n"; //Display adr

int\* q = (int\*)&c; // convert char c adr to int adr,

// and make q point to char

cout<<"\*q="<<\*q<<"\n"; // \*q

\*q = 65;

cout<<"c="<<c<<"\n"; // \*q

// int\* p = static\_cast<int\*>(&c); //Error

return 0;

}

1. Stream manipulators

#include <iostream> // std::cout, std::hex, std::endl

using std::cout;

using std::endl;

using std::ios;

using std::hex;

#include <iomanip> // std::setiosflags

using std::setprecision;

using std::setiosflags; //Number format

using std::setw;

int main (void) {

cout << hex;

cout << setiosflags (ios::showbase | ios::uppercase);

//Show number's base and in uppercase

cout << 100 << endl;

return 0;

}

#include <iostream>

using std::cout;

using std::endl;

using std::ios; //i/o stream

#include <iomanip>

using std::setprecision;

using std::setiosflags; // output format

using std::setw;

const double PI = 3.14159265358979;

int main(void){

const float PI = static\_cast< float >( ::PI );

// ::PI - file scope of PI

cout << setprecision( 20 )

<< " Local float value of PI = " << PI

<< "\nGlobal double value of PI = " << ::PI << endl; //double: more bits than float

cout << setw( 38 ) << "Local float value of PI = "

<< setiosflags( ios::fixed | ios::showpoint )

<< setprecision( 10 ) << PI << endl;

return 0;

}