TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN

----------o0o----------

BÀI THỰC HÀNH MÔN HỌC

Tên môn học: **Lập trình nâng cao**

Số tin chỉ/ĐVHT: **3TC (2LT: 1TH)**

Hệ đào tạo: **Đại học** Ngành**: Các ngành**

Số tín chỉ thực hành: **1 TC**

Giảng viên biên soạn:

1. **Nguyễn Hải Minh**
2. **Nguyễn Tuấn Anh**
3. **Hà Mạnh Hùng**
4. **Nguyễn Quang Hiệp**
5. **Trần Lâm**

Bộ môn: **Công nghệ lập trình và Ứng dụng**

Thái nguyên 2016

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN

----------o0o----------

BÀI THỰC HÀNH MÔN HỌC

LẬP TRÌNH NÂNG CAO

KHOA PHÊ DUYỆT **BỘ MÔN PHÊ DUYỆT** **GIÁO VIÊN PHỤ TRÁCH**

Thái nguyên, Năm 2016

MỤC LỤC

BÀI THỰC HÀNH 1: CĂN BẢN VỀ C++

1.1. Mục đích

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kĩ năng cơ bản sau:

- Hiểu được cấu trúc chung của một chương trình c++
- Sử dụng các cấu trúc điều khiển, cấu trúc lặp, rẽ nhánh
- Biên dịch và sử dụng môi trường lập trình thành thạo

1.2. Yêu cầu

- Chuẩn bị trước các bài tập thực hành
- Hoàn thành các bài tập trong chương

1.3. Tóm tắt lý thuyết

1.4. Nội dung thực hành

Các bước trong bài học này được trình bày chi tiết, rõ ràng và cẩn thận. Điều này giúp ta hiểu rõ về công cụ lập trình. Thực hiện theo các bước sau thật cẩn thận.

1.4.1. Bài thực hành mẫu

1.4.2. Các bài thực hành cơ bản

Bài 1. Viết chương trình có dùng hàm kiểm tra năm nhuận.

```
#include <iostream.h>
#include <conio.h>
int isLeapYear(int y)
{ return y % 4 == 0 && (y % 100 != 0 || y % 400 == 0);
}
void main()
{
        int n;
        do
        {
            cin >> n;
            if (isLeapYear(n)) cout << n << " la nam nhuan.\n";
            else cout << n << " Khong phai nam nhuan.\n";
        }
```

```
        while (n>1);
}
```

Bài 2. Viết chương trình có dùng hàm kiểm tra số nguyên tố.

```
#include <iostream
#include <conio.h>
#include <math.h>
int isPrime(int p)
{
        float sqrtp = sqrt(p);
        if (p < 2) return 0;
        if (p == 2) return 1; //2 la so nguyen to dau tien
        if (p % 2 == 0) return 0; // 2 la so nguyen to chan duy nhat
        for (int d = 3; d <= sqrtp; d += 2)
        if (p % d == 0) return 0;
return 1;
}
void main()
{
        clrscr();
        int n;
        cout<<"\n Nhap n = ";
        cin>>n;
        for (int i = 1; i < n; i++)
              if (isPrime(i)) cout << i << ' ';
               cout << endl;
getch(); }
```

Bài 3: Xây dựng một hàm sắp xếp theo thứ tự tăng dần một mảng gồm n số thực. Viết chương trình để nhập n số thực từ bàn phím, sử dụng hàm sắp xếp nói trên, và in ra màn hình hai cột song song, một cột là mảng chưa sắp xếp, một cột là mảng đã được sắp xếp.

```
#include <iostream.h>
#include <iomanip.h>
#include <math.h>
#include <conio.h>
void NhapMang(float a[],float b[],int n)
{
   cout<<"\nNhap mang\n";
   for (int i=0;i<n;i++)
   {
     cout<<"a["<<i<<"]=";
       cin>>a[i];
       b[i]=a[i];
} }

void InMang(float a[],float b[],int n)
{
  cout<<setw(10)<<"TT";
  cout<<setw(20)<<"Mang chua sap xep";
  cout<<setw(20)<<"Mang da sap xep";

cout<<"\n-----------------------------------------------------";
   for (int i=0;i<n;i++)
   {
       cout<<endl<<setw(10)<<i+1<<setw(20)<<a[i]<<setw(20)<<b[i];
   }
  cout<<"\n-----------------------------------------------------\n";
}
void SapXep(float a[],int n)
{
   //Sap xep tang dan
   for(int i=0;i<n-1;i++)
      for(int j=i+1;j<n;j++)
         if (a[i]>a[j])
{   float tg;

            tg=a[i]; a[i]=a[j]; a[j]=tg;
         }
}
void main(void)
{
```

```
    clrscr();
    float a[100],b[100];
    int n;
    cout<<"Nhap N=";
    cin>>n;
    NhapMang(a,b,n);
    SapXep(b,n);
    //Thiet lap dinh dang
    cout<<setiosflags(ios::showpoint|ios::fixed);
    cout<<setprecision(2);
    InMang(a,b,n);
    getch();
}
```

BÀI THỰC HÀNH 2: CON TRỎ

2.1. Mục đích

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kĩ năng cơ bản sau:

- Ý nghĩa, cách khai báo con trỏ
- Sử dụng con trỏ trong mảng, chuỗi
- Truyền mảng và chuỗi giữa các hàm qua con trỏ
- Xử lý mảng dễ dàng qua con trỏ

2.2. Yêu cầu

- Chuẩn bị trước các bài tập thực hành
- Hoàn thành các bài tập trong chương

2.3. Tóm tắt lý thuyết

2.4. Nội dung thực hành

Các bước trong bài học này được trình bày chi tiết, rõ ràng và cẩn thận. Điều này giúp ta hiểu rõ về công cụ lập trình. Thực hiện theo các bước sau thật cẩn thận.

Các biến con trỏ trong C chứa địa chỉ của một biến có bất kỳ kiểu nào. Nghĩa là, các con trỏ có thể là kiểu dữ liệu số nguyên hoặc ký tự. Một biến con trỏ số nguyên sẽ chưa địa chỉ của một biến số nguyên. Một con trỏ ký tự sẽ chứa địa chỉ của một biến kiểu ký tự

2.4.1. Bài thực hành mẫu

2.4.2. Các bài thực hành cơ bản

Bài 1: Tính giá trị của apples, *ptrApp, *ptrApp2 của đoạn mã sau:

```
int apples;
int *ptrApp = &apples;
int *ptrApp2 = ptrApp;
apples += 2;
*ptrApp --;
*ptrApp2 += 3;
```

(

Bài 2: Xác định kết quả của đoạn mã sau: (

```
int *p1 = new int;
```

```
int *p2;
*p1 = 5;
 p2 = p1;
cout << "*p1 = " << *p1 << endl;
cout << "*p2 = " << *p2 << endl;
*p2 = 10;(
cout << "*p1 = " << *p1 << endl;
cout << "*p2 = " << *p2 << endl;
p1 = new int; (*p1 = 20;(
cout << "*p1 = " << *p1 << endl;
cout << "*p2 = " << *p2 << endl; (
```

Bài 3: Hãy nêu các tình huống có thể xảy ra đối với đoạn chương trình sau (

```
int *p1 = new int;
int *p2;
*p1 = 5;
p2 = p1;
cout << *p2;
delete p1;
cout << *p2;
*p2 = 10;
```

Bài 4: Hãy nêu các tình huống có thể xảy ra đối với đoạn chương trình sau:

```
int *p1 = new int;
int *p2 = new int[10];
*p1 = 5;
p2 = p1;
cout << *p2;
```

Bài 5: Hãy nêu các tình huống có thể xảy ra đối với đoạn chương trình sau:

```
int *a = new int[10];
for (int i = 0; i <= 10; i ++) {
    print a[i];
    a[i] = i;
}
```

Bài 6: Nhập từ bàn phím vào một danh sách các số nguyên. Hãy sử dụng cấu trúc mảng động để lưu giữ dãy số nguyên này và tìm số lượng số nguyên chia hết cho số nguyên đầu tiên trong dãy. Hiện kết quả ra màn hình.

Bài 7: Phân tích sự khác biệt, nhược điểm, ưu điểm của việc sử dụng mảng động và mảng tĩnh. Những lưu ý khi khai báo, xin cấp phát và giải phóng bộ nhớ đối với mảng động.

Bài 8: Sử dụng cấu trúc mảng động để lưu giữ một danh sách tên các sinh viên nhập vào từ bàn phím. Hãy sắp xếp danh sách tên sinh viên tăng dần theo độ dài của tên. Hiện ra màn hình danh sách sau khi đã sắp xếp.

2.4.4. Các bài thực hành tự làm

Bài 9: Viết một chương trình C để nối hai chuỗi bằng cách sử dụng các con trỏ.

Để thực hiện điều này,

1. Khai báo ba biến chuỗi.
2. Khai báo ba con trỏ kiểu ký tự.
3. Nhập các giá trị của hai chuỗi.
4. Tạo ba con trỏ để trỏ đến ba biến chuỗi. Chuỗi thứ ba hiện tại không có bất kỳ giá trị gì.
5. Lặp qua chuỗi thứ nhất và sao chép nội dung của chuỗi đó vào chuỗi thứ ba. Sử dụng các biến con trỏ để sao chép các giá trị.
6. Sau khi sao chép chuỗi thứ nhất, lặp qua chuỗi thứ hai và chép nội dung của chuỗi vào cuối chuỗi ba. Sử dụng các biến con trỏ để sao chép giá trị.
7. In ra chuỗi thứ ba.

Bài 10: Viết một chương trình C để đảo một mảng ký tự bằng cách sử dụng con trỏ.

Bài 11: Viết một chương trình để cộng hai ma trận sử dụng các con trỏ

Bài 12: Sử dụng con trỏ để:

- Nhập vào một mảng a gồm n phần tử nguyên. Mảng a được gọi là hợp lệ nếu chứa ít nhất một phần tử âm. Hãy kiểm tra xem a có hợp lệ không?
- Nếu a hợp lệ, hãy sắp các phần tử của a sao cho các số âm về đầu mảng. Nếu a không hợp lệ, gọi K = Min-Max với Min và Max là các giá trị nhỏ nhất và lớn nhất của mảng, hãy chèn K vào ngay sau phần tử nhỏ nhất đầu tiên của a. Xuất mảng kết quả ra màn hình.

Bài 13: Sử dụng con trỏ để:

- Nhập vào một mảng a gồm n phần tử nguyên. Sắp mảng a sao cho các số lớn nhất về đầu mảng, sau đó là các số nhỏ nhất, các số còn lại sắp giảm dần; xuất kết quả ra màn hình.
- Mảng a được gọi là hợp lệ nếu tổng các số âm và tổng các số dương bằng nhau về giá trị tuyệt đối. Hãy kiểm tra xem a có hợp lệ không?
- Xóa mọi số vừa chẵn vừa dương trong mảng a; xuất mảng kết quả ra màn hình.

Bài 14: Sử dụng con trỏ để:

- Nhập vào một mảng a gồm n phần tử nguyên.
- Mảng a được gọi là hợp lệ nếu nó không giảm (tức phần tử đứng sau luôn lớn hơn hoặc bằng phần tử đứng trước. Hãy cho biết mảng vừa nhập có hợp lệ không?
- Sắp xếp mảng a theo chiều tăng dần; in mảng vừa sắp ra màn hình. Xóa mọi phần tử chẵn trong mảng và xuất mảng kết quả ra màn hình.

BÀI THỰC HÀNH 3: Xử lý Xâu trong C++

3.1. Mục đích

- Hiểu và làm việc với 2 kiểu xâu kí tự - xâu kiểu C là một mảng kí tự và xâu kiểu C++ là một biến của lớp string. (
- Xâu C: các hàm thư viện xử lý xâu C (<cstring> và <cctype>) (
- Vận dụng xâu C thao tác với đối số dòng lệnh (
- Xâu kiểu C++: các phương thức của lớp string trong thư viện <string> (

3.2. Yêu cầu

- Chuẩn bị trước các bài tập thực hành
- Hoàn thành các bài tập trong chương

3.3. Tóm tắt lý thuyết

- Trong C++ không có kiểu dữ liệu cơ bản để lưu các xâu kí tự

- Xâu là một mảng kí tự (mảng có kiểu char) Ví dụ:(**char str[20];(**Biến str có thể lưu một xâu kí tự với độ dài cực đại là 20 kí tự **Chú ý:** kiểu char chỉ lưu 1 kí tự đơn
Mảng kí tự có thể lưu các xâu kí tự dài không quá dài khai báo của nó ví dụ: Str có thể lưu "Hello" hay “Thai Nguyen"
Các kí tự trong xâu có địa chỉ liền kề nhau trong bộ nhớ (tính chất của mảng)
C++ quy ước để kết thúc một nội dung của một xâu kí tự bằng một kí tự null(null là kí tự có mã 0 trong bảng mã ASCII)

1. Code: **char s[100];** //khai báo s không có khởi tạo

2. Code: **char str[20]={'h','a','n','o','i','\0'};**
Xâu str có 20 phần tử, có 6 phần tử đầu được khởi tạo '\0' là kí tự null, dùng kí tự đặc biệt \ kết hợp với 0 để biểu diễn

3. **char st[]={‘S',‘V','\0'};(**
Xâu st có đúng 3 phần tử, và đã khởi tạo

4.Khai báo hợp lệ:
Code:
char mystring [] = { 'H', 'e', 'l', 'l', 'o', '\0' };
char mystring [] = "Hello";
Hai cách khai báo trên là tương đương(

5. Các lệnh sau là **không hợp lệ** cho biến mảng mystring
Code:
mystring = "Hello";

mystring[] = "Hello";

mystring = { 'H', 'e', 'l', 'l', 'o', '\0' };

6.Biến con trỏ kiểu char được hiểu như là biến xâu kí tự 6.1. Khai báo hợp lệ:

Code:

char * mystring = { 'H', 'e', 'l', 'l', 'o', '\0' };

char * mystring = “Hello”;

Hai cách khai báo trên là tương đương

7. Con trỏ kiểu xâu được phép:

char *p;(

p = "Hello Hanoi";

3.4. Nội dung thực hành

3.4.1. Bài thực hành mẫu

Bài 1: Chương trình ví dụ làm việc với đối số dòng lệnh. In ra màn hình kết quả nối đối số thứ 2 và thứ 3 nếu tổng độ dài không quá 1000

```
// Neu nguoi dung nhap "noi xau1 xau2"
// thi in ra ket qua noi xau2 vao sau xau1
// Neu nguoi dung nhap it hoac nhieu hon 3 doi so
// thi in ra huong dan su dung
#include <iostream>
#include <cstring>
using namespace std;
const int N_ARGC = 3;
const int MAX_LEN = 1001;
int main(int argc, char* argv[]){
if(argc != N_ARGC){(cout << "Cu phap: noi xau1 xau2" << endl
<< "Trong do:" << endl(<< "- noi la ten chuong trinh" << endl(<< "- xau1 la nua
dau ket qua" << endl(<< "- xau2 la nua cuoi ket qua" << endl(<< "Luu y: Ket
qua noi khong duoc vuot qua " << MAX_LEN - 1 << " ki tu." << endl;
  return -1;
}
   int n = strlen(argv[1]) + strlen(argv[2]) + 1;
   if(n > MAX_LEN){
cerr << "Ket qua noi vuot qua vuot qua " << MAX_LEN - 1 << " ki tu." << endl;
return -1;
}
   char ketqua[MAX_LEN] = "";
   strcat(ketqua, argv[1]);
   strcat(ketqua, argv[2]);
   cout << ketqua << endl;
return 0; }
```

Kết quả thực thi:

```
>noi ?
Cu phap: noi xau1 xau2 Trong do:
- noi la ten chuong trinh(- xau1 la nua dau ket qua(- xau2 la nua cuoi ket
qua(Luu y: Ket qua noi khong duoc vuot qua 1000 ki tu.
>noi prog ramming
programming
```

Bài 2: Chương trình ví dụ làm việc với xâu C

```
// Modified from(// http://www.cprogramming.com/tutorial/lesson9.html #include
<iostream> // de su dung cout(#include <cstring> // de su dung cac ham tren xau C

using namespace std;
int main()

{
   char name[50];
   char lastname[50];
   char fullname[100]; // du lon de luu ca name va lastname
cout << "Moi nhap vao ten ban: ";(cin.getline(name, 50);(if(strcmp(name, "Diep")
== 0) // so sanh 2 xau C, truong hop bang

      cout << "Ten toi giong ten ban day.\n";
   else                          // truong hop khac
      cout << "Ten toi khac ten ban.\n";
   // Tinh do dai ten vua nhap
cout << "Ten ban chua " << strlen(name) << " chu cai.\n";(cout << "Hay nhap ho cua
ban: ";(cin.getline(lastname, 50);(fullname[0] = '\0'; // strcat tim den '\0' de noi' vao
phia sau strcat(fullname, lastname); // sao lastname vao fullname strcat(fullname, "
"); // them dau cach de tach ho va ten strcat(fullname, name); // sao name vao cuoi
fullname

cout << "Ho ten cua ban la: "<< fullname << "\n"; cin.get();(return 0;

}
```

Kết quả thực thi:

```
Moi nhap vao ten ban: Mai
Ten toi khac ten ban.
Ten ban chua 3 chu cai.
Hay nhap ho cua ban: Tran
Ho ten cua ban la: Tran Mai
```

Bài 3: Chương trình ví dụ làm việc với các hàm thành viên của lớp string

```
// Note: This is modified from CS Dep C++ manual
#include <iostream>
#include <string>
using namespace std;
int main()

{
   string string1("cat");
   string string2;
   string string3;
string2 = string1; // gan string1 vao string2 string3.assign(string1); // gan string1 vao
string3(cout << "string1: " << string1 << "\nstring2: " << string2

      << "\nstring3: " << string3 << "\n\n";
   // chinh sua string2 va string3
string2[0] = string3[2] = 'r';(cout << "Sau khi chinh sua xau thu 2 va xau thu 3:\n"

      << "string1: " << string1
      << "\nstring2: " << string2 << "\nstring3: ";
   // thuc thi ham thanh vien at()
   for(int i = 0; i < string3.length(); i++)
      cout << string3.at(i);
   // khai bao string4
   string string4(string1 + "apult"); // ghep xau
// phep += nap chong(string3 += "pet"; // tao ra xau "carpet"
string1.append("acomb"); // tao ra xau "catacomb"

   string string5;
// chen string1 vao string5 sau khi loai bo 4 ky tu dau cua string1 // de tao ra xau
"comb" (string5 da duoc khoi tao rong luc ban dau) string5.append(string1, 4,
string1.length() - 4);

cout << "\n\nSau khi ghep:\nstring1: " << string1(<< "\nstring2: " << string2 <<
"\nstring3: " << string3(<< "\nstring4: " << string4 << "\nstring5: " << string5 <<
endl;

cin.get();

   return 0;
} // ket thuc ham main
```

Kết quả thực thi:

```
string1: cat
string2: cat
string3: cat
Sau khi chinh sua xau thu 2 va xau thu 3:
string1: cat
```

string2: rat
string3: car
Sau khi ghep:
string1: catacomb
string2: rat
string3: carpet
string4: catapult
string5: comb

3.4.2. Các bài thực hành cơ bản

1.Gán giá trị cho xâu ví dụ:(

Code:

```
mystring[0] = 'H';
mystring[1] = 'e';
mystring[2] = 'l';
mystring[3] = 'l';
mystring[4] = 'o';
mystring[5] = '\0' ;
```

rõ ràng cách này hợp lệ nhưng không khả thi

2.Gán giá trị cho xâu sử dụng hàm **strcpy** của thư viện **string.h**

Cú pháp:nguyên mẫu của hàm: **char *strcpy(char *dest, const char *src);**

Tác dụng:

Copy xâu src (nguồn) sang dest (xâu đích)(Xâu src (nguồn) không bị biến đổi (const char *src) Hàm này cũng trả về địa chỉ của đầu của dest(

ví dụ:(Code:

```
#include <iostream.h>
#include <string.h>
void main(){
char myName [20];
(strcpy (myName, "Nguyen Van A");
cout<<myName; //sẽ in ra: Nguyen Van A }
```

3. Nhập xâu kí tự từ bàn phím dùng lệnh **cin.getline** của thư viện **iostream.h**

Cú pháp:

Khai báo thư viện iostream.h□nguyên mẫu của hàm: **istream& getline(char*str , int len, char d= '\n');**

Tác dụng:(Gán chuỗi kí tự nhập từ bàn phím (với độ dài tối đa là len) vào biến xâu str

Chú ý: Không nên dùng cin>>str để nhập xâu(

Ví dụ:(Code:

```
#include <iostream.h>
void main(){
char s [100];
cin.getline(s,100);
cout<<s;
}
```

1. Nhập xâu từ bàn phím dùng lệnh **gets** của thư viện **stdio.h**

Cú pháp:(Khai báo thư viện **stdio.h(**nguyên mẫu của hàm: **char *gets(char *str);**

Tác dụng:

Gán chuỗi kí tự nhập từ bàn phím vào biến xâu str, và trả về địa chỉ đầu của str

Ví dụ Code:

```
#include <iostream.h>
#include <stdio.h>
void main(){
char str[100];
gets(str);
cout<<str;
}
```

Lưu **ý:** Trong lời giải của câu 1-4, bạn chỉ được dùng xâu C. Câu 5 phải dùng lớp string.

Câu 1. [reverse.cpp]

Hãy viết hàm đảo ngược nội dung của 1 xâu. Đọc từ bàn phím không quá 50 xâu (xâu có thể chứa dấu trắng ở giữa), đảo ngược nội dung của từng xâu rồi ghi ra màn hình.

Ví dụ:	Input:	Output:(
	nhap xau: ik auig iht	thi giua ki

Câu 2. [stdname.cpp]

Nhập vào từ bàn phím họ tên một người. Chuẩn hóa xâu này với ba bước sau:

- Loại bỏ dấu cách ở hai đầu xâu.
- Loại bỏ dấu cách thừa ở giữa các từ.
- Viết hoa các chữ cái đầu từ.

Ví dụ:

Input:

Nhap mot xau: nguyen Viet anh

Output:

Sau khi bo dau cach o hai dau:
nguyen Viet anh
Sau khi loai bo dau cach giua cac tu:
nguyen Viet anh
Sau khi viet hoa cac chu cai dau cac tu:
Nguyen Viet Anh

Câu 3. [count.cpp]

Nhập vào một xâu kí tự chỉ gồm các chữ cái và dấu cách. Thống kê số lần xuất hiện của từng chữ cái (không phân biệt viết hoa viết thường) trong xâu, sắp xếp chúng theo thứ tự giảm dần và in kết quả.

Ví dụ: Chương trình hoạt động như sau:(

Moi ban nhap mot xau: Tan suat cua cac ky tu:

We are the world
e:3
w:2
r:2
a:1
t:1
h:1
o:1
l:1
d:1

Câu 4. [caesar.cpp]

Mã hóa Caesar được xây dựng như sau:(*Với* xâu kí tự s và một số tự nhiên k, mỗi kí tự bị mã hóa bằng cách *d*ịch về sau k vị trí trong bảng chữ *cái (n*ếu hết bảng chữ cái thì quay ngược trở lại). Việc giải mã được thực hiện ngược lại với việc mã *hóa.*

Hãy viết chương trình nhận 4 đối số dòng lệnh(

1. Tên chương trình: caesar(
2. Bằng "encoding" nếu muốn mã hóa, "decoding" nếu muốn giải mã 3. k(
4. Xâu s (đặt xâu này trong cặp ngoặc kép nếu chứa dấu cách)

Sau đó in ra xâu mã hóa hoặc giải mã Caesar của s tùy theo đối số thứ 2. Ở đây ta sử dụng bảng chữ cái tiếng Anh.

Ví dụ:(>caesar encoding 3 "Nguyen Viet Anh"
Qjxbhq Ylhw Dqk
>caesar decoding 3 "Qjxbhq Ylhw Dqk"
Nguyen Viet Anh

Câu 5. [student.cpp]

Cho cấu trúc Date biểu diễn ngày tháng và cấu trúc Student biểu diễn thông tin sinh viên như sau:

```
struct Date{
   int day;
   int month;
   int year;
};
struct Student{
int id;
string fullname;
Date birthday;
string yclass;
};
```

Viết chương trình cho người dùng chọn 1 trong 4 tính năng từ menu:(

1. Nhập sinh viên: *nh*ập *thêm vào danh sách sinh* viên đến khi người dùng không muốn nhập tiếp

2. Sắp xếp theo họ tên: *s*ắp xếp tăng dần *danh sách sinh viên theo tên, n*ếu tên trùng nhau thì sắp

*x*ếp theo họ và đệm*)*. Sau đó in ra danh sách đ*ã s*ắp.(

3. Sắp xếp theo lớp khóa học: *s*ắp xếp tăng dần *danh sách sinh viên theo l*ớp khóa học. Sau đó in

ra danh sách đ*ã s*ắp.

4. Thoát

Sau khi thực hiện xong 1 tính năng (1-3) thì in lại menu cho phép tiếp tục chọn.Yêu cầu cài đặt: Hãy sử dụng vector<Student> để lưu danh sách. Trong câu này bạn không được sử dụng hàm sắp xếp có sẵn trong các thư viện của C++.

3.4.2. Các bài thực hành nâng cao

Bài 6: Viết hàm đếm số từ xuất hiện trong xâu. Ứng dụng hàm đó viết chương trình hoàn chỉnh nhập xâu ký tự từ bàn phím, in ra màn hình số từ có trong xâu

Bài 7: Viết hàm đếm số từ bắt đầu bằng 'tr' có trong xâu. Ứng dụng hàm đó viết chương trình hoàn chỉnh nhập xâu ký tự từ bàn phím, in ra màn hình số từ bắt đầu bằng 'tr' có trong xâu.

Bài 8: Viết hàm đếm số kết thúc bằng 'ng' có trong xâu. Ứng dụng hàm đó viết chương trình hoàn chỉnh nhập xâu ký tự từ bàn phím, in ra màn hình số từ kết thúc bằng 'ng' có trong xâu.

Bài 9: Viết hàm in các từ trong xâu, mỗi từ trên 1 dòng. Ứng dụng hàm đó viết chương trình hoàn chỉnh nhập xâu ký tự từ bàn phím, in ra màn hình các từ trong xâu, mỗi từ trên 1 dòng

Bài 10: Viết hàm kiểm tra xem 1 chuỗi có phải chứa toàn ký tự số hay không? Ứng dụng hàm trên và hàm trong bài 9 viết chương trình hoàn chỉnh nhập xâu ký tự từ bàn phím để đếm xem trong xâu có bao nhiêu từ toàn là số?

Bài 11: Viết hàm kiểm tra xem 1 chuỗi có phải chứa toàn ký tự số hay không? Nếu toàn là số thì tính giá trị của số tương ứng với chuỗi đó? Ứng dụng hàm trên và hàm trong bài 9 viết chương trình hoàn chỉnh nhập xâu ký tự từ bàn phím để tính tổng các từ là số trong xâu?

Bài thực hành số 5+6: CÁC THAO TÁC TRÊN TẬP TIN

5.1. Mục đích

- Biết đọc, tạo viết và update file. (
- Chuỗi sử lý file. (
- Truy cập ngẫu nhiên file.
- Sự khác nhau giữa dữ liệu đã định dạng với sử lý file dữ liệu thô
- Xây dựng quá trình sử lý ngẫu nhiên file
- Hiểu các khái niệm liên quan.

5.2. Yêu cầu

- Chuẩn bị trước các bài tập thực hành
- Hoàn thành các bài tập trong chương

5.3. Tóm tắt lý thuyết

5.4. Nội dung thực hành

5.4.1. Bài thực hành mẫu

Bài 1: thực hiện tạo cấu trúc file output

```
#include <fstream>
#include <iostream>
#include <cstdlib> // for exit()
int main()
{
   using namespace std;

   // ofstream is used for writing files
   // We'll make a file called Sample.dat
   ofstream outf("Sample.dat");

   // If we couldn't open the output file stream for writing
   if (!outf)
   {
      // Print an error and exit
      cerr << "Uh oh, Sample.dat could not be opened for writing!" << endl;
      exit(1);
   }

   // We'll write two lines into this file
   outf << "This is line 1" << endl;
   outf << "This is line 2" << endl;

   return 0;
```

```
    // When outf goes out of scope, the ofstream
    // destructor will close the file
}
```

Bài 2: thực hiện tạo cấu trúc file input

```
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib> // for exit()

int main()
{
    using namespace std;

    // ifstream is used for reading files
    // We'll read from a file called Sample.dat
    ifstream inf("Sample.dat");

    // If we couldn't open the output file stream for reading
    if (!inf)
    {
        // Print an error and exit
        cerr << "Uh oh, Sample.dat could not be opened for reading!" << endl;
        exit(1);
    }

    // While there's still stuff left to read
    while (inf)
    {
        // read stuff from the file into a string and print it
        std::string strInput;
        inf >> strInput;
        cout << strInput << endl;
    }

    return 0;
    // When inf goes out of scope, the ifstream
    // destructor will close the file
}
```

Kết quả file:

```
This
is
line
1
This
is
```

line
2

Bài 3: Sử dụng getline

```
#include <fstream>
#include <iostream>
#include <string>
#include <cstdlib> // for exit()

int main()
{
    using namespace std;

    // ifstream is used for reading files
    // We'll read from a file called Sample.dat
    ifstream inf("Sample.dat");

    // If we couldn't open the input file stream for reading
    if (!inf)
    {
        // Print an error and exit
        cerr << "Uh oh, Sample.dat could not be opened for reading!" << endl;
        exit(1);
    }

    // While there's still stuff left to read
    while (inf)
    {
        // read stuff from the file into a string and print it
        std::string strInput;
        getline(inf, strInput);
        cout << strInput << endl;
    }

    return 0;
    // When inf goes out of scope, the ifstream
    // destructor will close the file
```

Bài 4:

```
#include <cstdlib> // for exit()
#include <iostream>
#include <fstream>

int main()
{
    using namespace std;

    // We'll pass the ios:app flag to tell the ofstream to append
    // rather than rewrite the file.  We do not need to pass in ios::out
    // because ofstream defaults to ios::out
```

```
    ofstream outf("Sample.dat", ios::app);

    // If we couldn't open the output file stream for writing
    if (!outf)
    {
        // Print an error and exit
        cerr << "Uh oh, Sample.dat could not be opened for writing!" << endl;
        exit(1);
    }

    outf << "This is line 3" << endl;
    outf << "This is line 4" << endl;

return 0;
    // When outf goes out of scope, the ofstream
    // destructor will close the file
}
```

Bài 5: Sử dung open

```
ofstream outf("Sample.dat");
outf << "This is line 1" << endl;
outf << "This is line 2" << endl;
outf.close(); // explicitly close the file

// Oops, we forgot something
outf.open("Sample.dat", ios::app);
outf << "This is line 3" << endl;
outf.close();
```

5.4.2. Các bài tập cơ bản

BÀI 1: Nhập vào từ bàn phím một danh sách sinh viên. Mỗi sinh viên gồm có các thông tin sau đây: tên tuổi, ngày tháng năm sinh, nơi sinh, quê quán, lớp, học lực (từ 0 đến 9). Hãy ghi thông tin về danh sách sinh viên đó ra tệp văn bản student.txt (

BÀI 2: Sau khi thực hiện bài 1, hãy viết chương trình nhập danh sách sinh viên từ tệp văn bản student.txt rồi hiển thị ra màn hình:

Thông tin về tất cả các bạn tên là Vinh ra tệp văn bản vinh.txt (

Thông tin tất cả các bạn quê ở Hà Nội ra tệp văn bản hanoi.txt (

Tổng số bạn có học lực kém (<4), học lực trung bình (≥4 và <8), học lực giỏi (≥8) ra tệp văn bản hocluc.txt. (

BÀI 3: Sau khi thực hiện bài 2, hãy viết chương trình cho biết kích thước của tệp văn bản student.txt, vinh.txt, hanoi.txt, hocluc.txt. Kết quả ghi ra tệp văn bản all.txt. (

BÀI 4: Viết chương trình kiểm tra xem tệp văn bản student.txt có tồn tại hay không? Nếu tồn tại thì hiện ra màn hình các thông tin sau:

Số lượng sinh viên trong tệp (

Số lượng dòng trong tệp (

Ghi vào cuối tệp văn bản dòng chữ "CHECKED" (

Nếu không tồn tại, thì hiện ra màn hình dòng chữ "NOT EXISTED". (

BÀI 5: Tệp văn bản numbers.txt gồm nhiều dòng, mỗi dòng chứa một danh sách các số nguyên hoặc thực. Hai số đứng liền nhau cách nhau ít nhất một dấu cách. Hãy viết chương trình tổng hợp các thông tin sau và ghi vào tệp văn bản info.txt những thông tin sau:

Số lượng số trong tệp

Số lượng các số nguyên

Số lượng các số thực (Lưu ý: Test chương trình với cả trường hợp tệp văn bản number.txt chứa một hay nhiều dòng trắng ở cuối tệp.

BÀI 6: Trình bày sự khác nhau, ưu điểm, nhược điểm giữa tệp văn bản và tệp văn bản nhị phân. Khi nào thì nên dùng tệp văn bản nhị phân.

BÀI 7: Cho file văn bản numbers.txt chứa các số nguyên hoặc thực. Hãy viết một chương trình đọc các số từ file numbers.txt và ghi ra file nhị phân numbers.bin các số nguyên nằm trong file numbers.txt.

BÀI 8: Sau khi thực hiện bài 7, hãy viết một chương trình đọc và tính tổng của tất cả các số nguyên ở file nhị phân numbers.bin. Hiện ra màn hình kết quả thu được.

BÀI 9: Sau khi thực hiện bài 7, hãy viết một chương trình đọc các số nguyên ở file nhị phân numbers.bin. Ghi các số nằm ở vị trí chẵn (số đầu tiên trong file được tính ở vị trí số 0) trong file nhị phân numbers.bin vào cuối file numbers.txt.

BÀI 10: Cho hai file văn bản num1.txt và num2.txt, mỗi file chứa 1 dãy số đã được sắp không giảm. Số lượng số trong mỗi file không quá 109. Hãy viết chương trình đọc và ghi ra file văn bản num12.txt các số trong hai file num1.txt và num2.txt thỏa mãn điều kiện các số trong file num12.txt cũng được sắp xếp không giảm.

BÀI 11: File văn bản document.txt chứa một văn bản tiếng anh. Các câu trong văn bản được phân cách nhau bởi dấu '.' hoặc '!'. Hãy ghi ra file văn bản sentences.txt nội dung của văn bản document.txt, mỗi câu được viết trên một dòng.

Ví dụ:

document.txt	sentences.txt
this is good house! , however, too expensive.	This is good house! However, too expensive.

BÀI THỰC HÀNH 7: MẢNG CẤU TRÚC

7.1. Mục đích

- ✓ Giải thích cách truyền cấu trúc vào hàm như các đối số
- ✓ Giải thíchTìm hiểu cách truyền đốitham số kiểu kiểu cấu trúc vào hàm
- ✓ Sử dụng mảng các cấu trúc
- ✓ Giải thích Tìm hiểu sự cách khởi tạo của các mảng cấu trúc

7.2. Yêu cầu

- Chuẩn bị trước các bài tập thực hành
- Hoàn thành các bài tập trong chương

7.3. Tóm tắt lý thuyết

Một cấu trúc là một nhóm các mẫu dữ liệu có thể có kiểu khác nhau. Mỗi cấu trúc phải được định nghĩa trước khi nó được sử dụng trong khai báo biến. Một định nghĩa cấu trúc có thể bao gồm một thành phần là một cấu trúc khác. Việc khởi tạo cấu trúc tương tự như việc khởi tạo mảng.

7.4. Nội dung thực hành

7.4.1. Bài thực hành mẫu

Câu 1:

struc dienthoai{
int sdt; //Số điện thoại
char hoten[25]; //Họ và tên
float sotien; //Số tiền phải nộp
} thuebao[100];

Hãy xây dựng một hàm để nhập số liệu cho n thuê bao. Sau đó viết một chương trình sử dụng hàm nói trên để nhập số liệu và in bảng số tiền phải nộp của các thuê bao theo dạng ba cột: Họ tên, số điện thoại, số tiền phải nộp.

```
#include<iostream.h>
#include<iomanip.h>
struct dienthoai{
int sdt;char hoten[25]; float sotien;
} thuebao[100];
//Hàm nhập danh sách n thuê bao void nhap(int n,dienthoai *a)
{{
```

```
} }
for(int i=0;i<n;i++)
cout<<"\n Nguoi Thu "<<i;
cout<<"\nSDT:"; cin>>a[i].sdt;
cin.ignore(1);
//Hàm in danh sách các thuê bao theo dạng ba cột 'Số điện thoại','Họ và tên','Số
tiền"void inds(int n,dienthoai *a){
    cout<<setw(20)<<"Dien thoai";
    cout<<setw(20)<<"Ho ten";
    cout<<setw(20)<<"So tien"<<endl;
    for(int i=0;i<n;i++)
{
cout<<setw(20)<<a[i].sdt;
cout<<setw(20)<<a[i].hoten;
cout<<setw(20)<<a[i].sotien<<endl;
} }
main()
{
dienthoai a[100];
int n;
cout<<"So thue bao: "; cin>>n;
nhap(n,a);
inds(n,a);
}
```

Câu 2. [thisinh.cpp]

Cho chương trình xử lý thông tin thí sinh dự thi khối A đại học như bên dưới. Chương trình sử dụng cấu trúc Thisinh để biểu diễn thông tin của một thí sinh. Cấu trúc này có 5 biến thành viên là: mã dự thi (maDuThi), họ tên (hoten), điểm toán (diemToan), điểm lý (diemLy), điểm hóa (diemHoa).

Chương trình đã định nghĩa sẵn 2 hàm:

- hàm nhập thông tin cho 1 thí sinh từ bàn phím (nhapThisinh)

- hàm in thông tin của 1 thí sinh ra màn hình dưới dạng 1 hàng trong bảng (inThisinh), nếu tham số inDongDau của hàm này bằng true, nó sẽ in thêm dòng tiêu đề của bảng

```
#include <iostream>
#include <iomanip>
```

```
using namespace std;
struct Thisinh{
    int maDuThi;
   string hoten;
    int diemToan;
    int diemLy;
    int diemHoa;
};
void nhapThisinh(Thisinh& ts);
// Ham in thong tin cua thi sinh duoi dang bang
   // Neu inDongDau == true thi in ra dong tieu de cac cot,
   // nguoc lai thi khong in
   void inThisinh(Thisinh ts, bool inDongDau=false);
   int main(){
      Thisinh ts1 = {1234, "Nguyen Van Bac", 7, 8, 9};
      Thisinh ts2;
      cout << "Nhap thong tin thi sinh 2: " << endl;
      nhapThisinh(ts2);
      inThisinh(ts1, true);
      inThisinh(ts2);
   cin.get();
   return 0; }
   void nhapThisinh(Thisinh& ts){
      cout << "Nhap ma du thi: ";
      cin >> ts.maDuThi;
      cout << "Nhap ho va ten: ";
      cin.ignore(); // bo di dau ENTER trong istream
      getline(cin, ts.hoten);
      cout << "Nhap diem toan: ";
      cin >> ts.diemToan;
      cout << "Nhap diem ly: ";
      cin >> ts.diemLy;
      cout << "Nhap diem hoa: ";
      cin >> ts.diemHoa;
      cin.ignore(80, '\n');
   }
   void inThisinh(Thisinh ts, bool inDongDau){
      cout.setf(ios::left);
      if(inDongDau)
         cout << setw(10) << "Ma"
            << setw(30) << "Ho ten"
            << "D.toan\tD.ly\tD.hoa" << endl;
      cout << setw(10) << ts.maDuThi
         << setw(30) << ts.hoten
         << ts.diemToan << "\t"
         << ts.diemLy << "\t"
         << ts.diemHoa << endl;
   }
```

a) Hãy bổ sung code vào hàm inThongtin để nó in thêm tổng điểm 3 môn của thí sinh.

b) Hãy viết hàm nhapMangThisinh thực hiện nhập thông tin cho một danh sách thí sinh.

c) Hãy viết hàm inMangThisinh thực hiện in thông tin cho một danh sách thí sinh dưới dạng bảng. Các hàm này cần tận dụng tối đa những hàm đã cho. Lưu ý bổ sung code vào hàm main để gọi tới 2 hàm mới này.

7.4.2. Các bài thực hành cơ bản

Bài 2: Định nghĩa kiểu dữ liệu của học sinh HOCSINH gồm:

- Mã số học sinh (MSHS): chuỗi có tối đa 5 ký tự.
- Họ tên (hoten): chuỗi có tối đa 30 ký tự.
- Ngày tháng năm sinh (ngaysinh): kiểu DATE.
- Địa chỉ (diachi): chuỗi có tối đa 50 ký tự.
- Giới tính (phai): chuỗi có tối đa 3 ký tự.
- Điểm trung bình (diemtb): số thực.

Ta định nghĩa kiểu HOCSINH như sau:

```
struct DATE
{
        char thu[5];
        unsigned char ngay;
        unsigned char  thang;
        int nam;
};
typedef struct HOCSINH
{
        char MSHS[6];
        char hoten[31];
        struct DATE ngaysinh;
        char diachi[51];
        unsigned char phai[4];
        float diemtb;
};
```

Khi định nghĩa kiểu dữ liệu struct lồng nhau, ta cần lưu ý: Kiểu dữ liệu được sử dụng phải khai báo phía trên.

Bài 4: Cho một lớp học gồm n học sinh ($n \leq 50$). Thông tin của một học sinh được mô tả ở ví dụ 2, mục I.2. Hãy viết chương trình nhập và xuất danh sách học sinh sau đó đếm xem có bao nhiêu học sinh được lên lớp (Điều kiện được lên lớp là điểm trung bình $\geq$ 5.0).

Cách làm:

- Trước hết ta phải xây dựng hàm nhập và xuất cho 1 học sinh.
- Xây dựng hàm nhập và xuất ngày tháng năm (Kiểu dữ liệu DATE).
- Sau đó mới xây dựng hàm nhập và xuất cho danh sách học sinh.

7.4.3. Các bài thực hành nâng cao

Bài 5: Viết một chương trình C++ để lưu trữ dữ liệu các thông tinh về sinh viên trong một cấu trúc. Dữ liệu phải bao gồm mã sinh viên, tên sinh viên, khóa học đã đăng ký và năm đăng ký. Viết một hàm để hiển thị các thông tin chi tiết của các sinh viên đã nhập học trong một năm học nào đó. Viết một hàm khác để định vị xác định và hiển thị thông tin chi tiết của một sinh viên dựa vào một mã sinh viên đã cho khi biết mã của sinh viên đó.

Yêu cầu:

8. Định nghĩa một cấu trúc để lưu trữ thông tin chi tiết của sinh viên.
9. Khai báo và khởi tạo biến cấu trúc với thông tin chi tiết của 10 sinh viên.
10. Viết vòng lặp để hiển thị một menu các thao tác mà chương trình có thể thực hiện.
10. Nhận vào lựa chọn danh mục và gọi hàm thích hợp với tham số là mảng cấu trúc.
11. Trong hàm dùng để hiển thị thông tin chi tiết của các sinh viên nhập học trong cho một năm, cho nhập vào năm. Đặt một vòng lặp để kiểm tra năm nhập học của mỗi sinh viên, và hiển thị nếu nó trùng. Ở cuối hàm, cho phép người dùng nhập vào một năm học khác viết chương trình để thực hiện nhập vào năm học cần được hiển thị thông tin, sau đó sử dụng vòng lặp để kiểm tra năm nhập học của từng sinh viên, nếu trùng với năm cần hiển thị thông tin yêu cầu thì hiển thị thông tin của sinh viên đó. Ngoài ra, hàm này còn cho phép người dùng có thể tiếp tục thực hiện

việc hiển thị thông tin của những năm khác cho đến khi họ không muốn sử dụng chức năng này nữa.

12. Trong Hàm dùng để định vị hiển thị thông tin chi tiết của sinh viên, cho phép nhập vào mã của sinh viên. Dùng một vòng lặp để kiểm tra mã của mỗi sinh viên, và hiển thị nếu nó trùng nếu mã của sinh viên nào trùng với mã đã được nhập thì hiển thị thông tin chi tiết của sinh viên đó. Ở cuối hàm, cho phép người dùng nhập vào một mã sinh viên khác. Ngoài ra, hàm này còn cho phép người dùng có thể tiếp tục thực hiện việc hiển thị thông tin của những sinh viên khác cho đến khi họ không muốn sử dụng chức năng này nữa

Bài 6: Viết một chương trình C để lưu trữ 5 độ dài trong một mảng cấu trúc. Các Mỗi độ dài phải lưu ở dạng bao gồm 3 thông tin về yards, feet và inches. Sắp xếp và hiển thị các độ dài.

Bài 7: Viết một chương trình C để lưu trữ thông tin chi tiết của nhân viên trong một mảng cấu trúc. Dữ liệu Thông tin của một nhân viên phải bao gồm mã nhân viên, tên, lương và ngày vào làm. Ngày vào làm phải được lưu trong một cấu trúc khác. Chương trình phải thực hiện các thao tác sau đây dựa trên sự lựa chọn trong một danh mục menu các chức năng của chương trình:

1. Tăng lương theo các luật sau:

Salary Range	**Percentage increase**
<= 2000	15%
> 2000 and <= 5000	10%
>5000	No increase

2. Hiển thị thông tin chi tiết của các nhân viên đã làm việc trong công ty từ 10 năm trở lên.

BÀI THỰC HÀNH 8: KIỂU DỮ LIỆU CON TRỎ CẤU TRÚC

8.1. Mục đích

Cung cấp cơ chế cho phép khai báo các kiểu dữ liệu con trỏ cấu trúc để giải quyết theo yêu cầu của bài toán vào những kiểu dữ liệu cơ bản được cài đặt sẵn trong ngôn ngữ lập trình.

8.2. Yêu cầu

- Chuẩn bị trước các bài tập thực hành
- Hoàn thành các bài tập trong chương

8.3. Tóm tắt lý thuyết

Việc khai báo một biến con trỏ kiểu cấu trúc cũng tương tự như khi khai báo một biến con trỏ khác, nghĩa là đặt thêm dấu * vào phía trước tên biến.

Cú pháp:

struct <Tên cấu trúc> * <Tên biến con trỏ>;

Ví dụ 1: Ta có thể khai báo một con trỏ cấu trúc kiểu NgayThang như sau:

```
struct NgayThang *p;
/* NgayThang *p; // Nếu có định nghĩa kiểu */
```

Sử dụng các con trỏ kiểu cấu trúc

Khi khai báo biến con trỏ cấu trúc, biến con trỏ chưa có địa chỉ cụ thể. Lúc này nó chỉ mới được cấp phát 2 byte để lưu giữ địa chỉ và được ghi nhận là con trỏ chỉ đến 1 cấu trúc, nhưng chưa chỉ đến 1 đối tượng cụ thể. Muốn thao tác trên con trỏ cấu trúc hợp lệ, cũng tương tự như các con trỏ khác, ta phải:

- Cấp phát một vùng nhớ cho nó (sử dụng hàm malloc() hay calloc);
- Hoặc, cho nó quản lý địa chỉ của một biến cấu trúc nào đó.

Ví dụ 2 : Sau khi khởi tạo giá trị của cấu trúc:

```
struct NgayThang Ngay = {29,8,1986};
p = &Ngay;
```

lúc này biến con trỏ p đã chứa địa chỉ của Ngay.

Truy cập các thành phần của cấu trúc đang được quản lý bởi con trỏ

Để truy cập đến từng trường của 1 cấu trúc thông qua con trỏ của nó, ta sử dụng toán tử dấu mũi tên (->: dấu - và dấu >). Ngoài ra, ta vẫn có thể sử dụng đến phép toán *

để truy cập vùng dữ liệu đang được quản lý bởi con trỏ cấu trúc để lấy thông tin cần thiết.

8.4. Nội dung thực hành

8.4.1. Bài thực hành mẫu

Bài 1: Chạy chương trình dưới và cho biết kết quả.

```
#include <iostream>
 #include <cstring>
using namespace std;
void printBook( struct Books *book );
struct Books {
char  title[50];
char  author[50];
char  subject[100];
int   book_id; };
int main( ) {
struct Books Book1;        // Declare Book1 of type Book
struct Books Book2;        // Declare Book2 of type Book
                            // Book 1 specification
strcpy( Book1.title, "Learn C++ Programming");
strcpy( Book1.author, "Chand Miyan");
strcpy( Book1.subject, "C++ Programming");
Book1.book_id = 6495407;     // Book 2 specification
strcpy( Book2.title, "Telecom Billing");
strcpy( Book2.author, "Yakit Singha");
strcpy( Book2.subject, "Telecom");
Book2.book_id = 6495700;      // Print Book1 info, passing address of structure
printBook( &Book1 );     // Print Book1 info, passing address of structure
printBook( &Book2 );
return 0; } // This function accept pointer to structure as parameter.
void printBook( struct Books *book ) {
cout << "Book title : " << book->title <<endl;
cout << "Book author : " << book->author <<endl;
cout << "Book subject : " << book->subject <<endl;
```

```
cout << "Book id : " << book->book_id <<endl;
}
```

Bài 2: Sử dụng con trỏ cấu trúc.

```
#include<conio.h>
#include<stdio.h>
typedef struct
{
   unsigned char Ngay;
   unsigned char Thang;
   unsigned int Nam;
} NgayThang;
int main()
{
   NgayThang Ng={29,8,1986};
   NgayThang *p;
   p=&Ng;
   printf("Truy cap binh thuong %d-%d-%d\n", Ng.Ngay,Ng.Thang,Ng.Nam);
   printf("Truy cap qua con tro %d-%d-%d\n",p->Ngay,p->Thang,p->Nam);
   printf("Truy cap qua vung nho con tro %d-%d-%d\n",
   (*p).Ngay,(*p).Thang,(*p).Nam);
   getch();
   return 0;
}
```

- Cho phép sử dụng cấu trúc, con trỏ cấu trúc là đối số của hàm như các loại biến khác
- Cho phép hàm xây dựng trả về là kiểu cấu trúc

Bài 3 : Gõ lại chương trình xử lý danh sách sinh viên, sử dụng hàm với đối số là cấu trúc bằng C++.

```
#include <stdio.h>
#include <conio.h>
```

```
//Ngay thang
struct Ngaythang {
int ng ;
int th ;
int nam ;
};
//Sinh vien
struct Sinhvien {
char hoten[25] ;
Ngaythang ns;
int gt;
float diem ;
} ;
//cac ham xu ly
int sua(Sinhvien &r) ;
int in(Sinhvien x) ;
int nhap(Sinhvien *p) ;
int nhapds(Sinhvien *a, int n) ;
int suads(Sinhvien *a, int n) ;
int inds(const Sinhvien *a, int n) ;
struct Sinhvien a[10];
int main()
{
    nhap(a);
//  in(a[1]);
//  sua(a[1]);
    nhapds(a,9);
    suads(a,9);
    inds(a,9);
    getch();
    return 0;
}
```

```
///trien khai cac ham
int sua(Sinhvien &r)
{
 int chon;
 do {
    printf("1: Sua ho ten\n2: Sua ngay sinh\n3:Sua gioi tinh\n4:Sua diem\n5:In\n0: Ket thuc\n");
    scanf("%d",&chon);
    switch (chon)
    {
        case 1:
            printf("Nhap ho ten:");
            scanf("%s",r.hoten);
            break;
        case 2:
            printf("Nhap ngay thang nam sinh:");
            scanf("%d%d%d",&r.ns.ng,&r.ns.th,&r.ns.nam);
            break;
        case 3:
            printf("Nhap gioi tinh 0:Nu 1:Nam:");
            scanf("%d",&r.gt) ;
            break;
        case 4:
            printf("Nhap diem:");
            scanf("%f",&r.diem);
            break;
        case 5:
            printf("Sinh vien:");
            in(r);
            break;
        case 0:
            break;
```

```
                default:
                                printf("Nhap gia tri khong dung\n");
                                break;
            }
         } while (chon) ;
         return 0;
        }
        ////////
        int in(Sinhvien x)
        {
        printf("Ho ten :%s\nNgay sinh %d/%d/%d\n",x.hoten,x.ns.ng, x.ns.th, x.ns.nam)
;
        printf("Gioi tinh :%s\nDiem :%f\n",(x.gt==1) ?"Nam" :"Nu",x.diem) ;
        return 0;
        }

        ///////
        int nhap(Sinhvien *p)
        {
                printf("Nhap ho va ten: ");
                scanf("%s",p->hoten);
                printf("Nhap ngay sinh (ngay thang nam): ");
                scanf("%d%d%d", &p->ns.ng ,&p->ns.th,&p->ns.nam);
                printf("Gioi tinh 0: nu, 1: nam: ");
                scanf("%d",& p->gt);
                printf("Diem: ");
                scanf("%f",& p->diem);
                return 0;
        }
        //////
        int nhapds(Sinhvien *a, int n)
        {
```

```
 for (int i=1; i<=n; i++) nhap(&a[i]) ;
 return 0;
}
/////
int suads(Sinhvien *a, int n)
{
    int chon;
    do
    {
        printf("\nNhap phan tu duoc su tu 1 den %d, gia tri khac thoat:",n);
        scanf("%d",&chon);
        if(chon>0 &&chon<=n)
        {
            sua(a[chon]) ;
        }
    }while(chon>0 && chon<=n);
    return 0;
}
//////////
int inds(const Sinhvien *a, int n)
{
   for (int i=1; i<=n; i++)
      in(a[i]) ;
   return 0;
}
```

8.4.2. Các bài thực hành cơ bản

Bài 4: Viết chương trình sử dụng con trỏ cấu trúc để hiển thị giờ, phút, giây ra màn hình, và tính khoảng cách giữa 2 mốc thời gian.

Bài 5: Viết chương trình sử dụng con trỏ cấu trúc thể hiện ngày, tháng, năm ra màn hình, và tính khoảng cách giữa 2 ngày.

Bài 6: Tổ chức dữ liệu để quản lí sinh viên bằng cấu trúc mẫu tin trong một mảng

N phần tử, mỗi phần tử có cấu trúc như sau:

- Mã sinh viên.
- Tên.
- Năm sinh.
- Điểm toán, lý, hoá, điểm trung bình.

Viết chương trình thực hiện những công việc sau:

- Nhập danh sách các sinh viên cho một lớp học.
- Xuất danh sách sinh viên ra màn hình.
- Tìm sinh viên có điểm trung bình cao nhất.

BÀI THỰC HÀNH 9: BẮT VÀ XỬ LÝ NGOẠI LỆ

9.1. Mục đích

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kĩ năng cơ bản bắt và xử lý ngoại lệ, nguyên tắc hoạc động của **try**, **throw** và **catch**. cách truyền bởi throw, các exceptions chuẩn.

9.2. Yêu cầu

- Chuẩn bị trước các bài tập thực hành
- Hoàn thành các bài tập trong chương

9.3. Tóm tắt lý thuyết

Những loại tình huống bất thường này được nằm trong cái được gọi là exceptions và C++ đã vừa tích hợp ba toán tử mới để xử lý những tình huống này: **try**, **throw** và **catch**.

Dạng thức sử dụng như sau:

```
try {
// đoạn mã cần thử
throw exception;
}
catch (type  exception)
{
// đoạn được thực hiện trong trường hợp có lỗi
}
```

Nguyên tắc hoạt động:

- Đoạn mã nằm trong khối try được thực hiện một cách bình thường. Trong trường hợp có lỗi xảy ra, đoạn mã này phải sử dụng từ khoá throw và một tham số để báo lỗi. Kiểu tham số này mô tả chi tiết hoá lỗi và có thể là bất kì kiểu hợp lệ nào.
- Nếu có lỗi xảy ra, nếu lệnh throw đã được thực hiện bên trong khối try, khối catch sẽ được thực hiện và nhận tham số được truyền bởi throw.

9.4. Nội dung thực hành

Các bước trong bài học này được trình bày chi tiết, rõ ràng và cẩn thận. Điều này giúp ta hiểu rõ về công cụ lập trình. Thực hiện theo các bước sau thật cẩn thận.

Các biến con trỏ trong C chứa địa chỉ của một biến có bất kỳ kiểu nào. Nghĩa là, các con trỏ có thể là kiểu dữ liệu số nguyên hoặc ký tự. Một biến con trỏ số

nguyên sẽ chưa địa chỉ của một biến số nguyên. Một con trỏ ký tự sẽ chứa địa chỉ của một biến kiểu ký tự

9.4.1. Bài thực hành mẫu

9.4.2. Các bài thực hành cơ bản

Bài 1: Cho chương trình sau

```
// exceptions
#include
int main () {
char myarray[10];
try
{
for (int n=0; n<=10; n++)
{
if (n>9) throw "Out of range";
myarray[n]= z ;
}
}
catch (char * str)
{
cout << "Exception: " << str << endl;
}
return 0;
}
```

Trong ví dụ này, nếu bên trong vòng lặp mà n lớn hơn 9 thì một lỗi sẽ được thông báo vì myarray[n] trong trường hợp đó có thể trỏ đến địa chỉ ô nhớ không tin cậy. Khi throw được thực hiện, khối try ngay lập tức kết thúc và mọi đối tượng được tạo bên trong khối try bị phá huỷ. Sau đó, quyền điều khiển được chuyển cho khối catch tương ứng (chỉ được thực hiện trong những tình huống như thế này). Cuối cùng chương trình tiếp tục ngay sau khối, trong trường hợp này: return 0;.

Cú pháp được sử dụng bởi throw tương tự với return: Chỉ có một tham số và không cần đặt nó nằm trong cặp ngoặc đơn.

Khối catch phải nằm ngay sau khối try mà không được có đoạn mã nào nằm giữa chúng. Tham số mà catch chấp nhận có thể là bất kì kiểu dữ liệu hợp lệ nào. Hơn nữa, catch có thể được quá tải để có thể chấp nhận nhiều kiểu dữ liệu khác nhau. Trong trường hợp này khối catch được thực hiện là khối phù hợp với kiểu của tham số được gửi đến bởi throw:

```
// exceptions: multiple catch blocks
#include
int main () {
try
{
char * mystring;
mystring = new char [10];
if (mystring == NULL) throw "Allocation failure";
for (int n=0; n<=100; n++)
{
if (n>9) throw n;
mystring[n]= z ;
}
}
catch (int i)
{
cout << "Exception: ";
cout << "index " << i << " is out of range" << endl;
}
catch (char * str)
{
cout << "Exception: " << str << endl;
}
return 0;
}
```

Ở đây có thể có hai trường hợp xảy ra:

2. Khối dữ liệu 10 kí tự không thể được cấp phát (gần như là chẳng bao giờ xảy ra nhưng không có nghĩa là không thể): lỗi này sẽ bị chặn bởi **catch (to char * str)**.

3. Chỉ số cực đại của **mystring** đã bị vượt quá: lỗi này sẽ bị chặn bởi **catch (int i)**, since parameter is an integer number.

Chúng ta có thể định nghĩa một khối **catch** để chặn tất cả các exceptions mà không phụ thuộc vào kiểu được dùng để gọi **throw**. Để làm việc này chúng ta phải viết dấu ba chấm thay vì kiểu và tên số tham số:

```
try {
// code here
}
catch (...) {
cout << "Exception occurred";
}
```

Còn có thể lồng các khối **try-catch** vào các khối **try** khác. Trong trường hợp này, một khối **catch** bên trong có thể chuyển tiếp exception nhận được cho khối bên ngoài, để làm việc này chúng ta sử dụng biểu thức **throw;** không có tham số.

```
Ví dụ:
try {
try {
// code here
}
catch (int n) {
throw;
}
}
catch (...) {
cout << "Exception occurred";
}
```

Bài 2: Exceptions chuẩn:

Một số hàm thuộc thư viện C++ chuẩn gửi các exceptions mà chúng ta có thể chặn nếu chúng ta sử dụng một khối **try**. Những exceptions này được gửi đi với kiểu tham số là một lớp thừa kế từ **std::exception**. Lớp này (std::exception) được định nghĩa trong file header C++ chuẩn và được dùng làm mẫu cho hệ thống phân cấp các exception chuẩn:

Bởi vì đây là một hệ thống phân lớp có thứ bậc, nếu bạn sử dụng một khối **catch** để chặn bất kì một exception nào nằm trong hệ thông này bằng cách sử dụng tham số biến (thêm một dấu & vào phía trước tên của tham số) bạn sẽ chặn được tất cả các exception thừa kế (luật thừa kế trong C++)

Ví dụ dưới đây chặn một exception có kiểu **bad_typeid** (được thừa kế từ **exception**), lỗi này được tạo ra khi muốn biết kiểu của một con trỏ null.

```
// Những exception chuẩn
#include
#include
#include
class A {virtual f() {}; };
int main () {
try {
A * a = NULL;
```

Exception: Attempted typeid of NULL pointer

```
typeid (*a);
}
catch (std::exception& e)
{
cout << "Exception: " << e.what();
}
return 0;
}
```

Bạn có thể sử dụng các lớp của hệ thống phân cấp các exception chuẩn này báo những lỗi của mình hoặc thừa kế những lớp mới từ chúng.

BÀI THỰC HÀNH 10: ÔN TẬP VÀ KIỂM TRA

10.1. Mục đích

10.2. Yêu cầu

- Chuẩn bị trước các bài tập thực hành
- Hoàn thành các bài tập trong chương

10.3. Tóm tắt lý thuyết

10.4. Nội dung thực hành

10.4.1. Bài thực hành mẫu.

Đề bài: Nhập n phần tử số nguyên, in ra màn hình giá trị nhỏ nhất và lớn nhất trong mảng

Cách 1 (đơn giản nhất):

+ đầu tiên yêu cầu người dùng nhập số lượng phần tử là n

+ lặp n lần, mỗi lần nhập 1 giá trị, vừa nhập xong là so sánh với biến min và max, nếu giá trị vừa nhập bé hơn min thì min sẽ bằng số vừa nhập, và nếu số đó lớn hơn biến max thì max sẽ bằng số đó

```
#include <iostream.h>
void main(){
  int x, i, n, min, max;
  cout << "Nhap so luong n = "; cin >> n;
  if ( n > 0 ) {
cout<<" Gia tri phan tu thu dau tien = "; cin >> x;
min = max = x;
for ( i = 2 ; i<=n; i++ ){
cout<<" Gia tri phan tu thu "<<i<<" = "; cin >> x;
      if ( x < min ) min = x;
      if ( x > max ) max = x;
    }
    cout << "Gia tri lon nhat = " << max << endl;
    cout << "Gia tri nho nhat = " << min << endl;
} }
```

Cách giải 2: dùng mảng (kinh điển)+ nhập mảng n phần tử số nguyên từ bàn phím (đã có thuật toán và code ở trên)+ gán tạm thời min và max bằng phần tử đầu tiên

+ duyệt qua toàn mảng: nếu giá trị đang duyệt bé hơn min hoặc lớn hơn max thì cập nhật lại min và max

```
#include <iostream.h>
void NhapMang(int *a, int &n){
  cout << "Nhap N = ";  cin >> n;
  cout << "Nhap mang" << endl;
  for (int i = 0; i < n ; i++){
    cout << "a[" << i << "] = ";
    cin >> *(a+i);
  }
}

void TimMinMax(int a[], int n, int &min, int &max){
  min = max = a[0];
  for (int i = 1; i < n ; i++){
    if ( a[i] < min ) min = a[i];
    if ( a[i] > max ) max = a[i];
} }

void main(){
  int a[100], n, min, max;
  NhapMang (a, n);
  TimMinMax (a, n, min, max);
  cout << "Gia tri lon nhat = " << max << endl;
  cout << "Gia tri nho nhat = " << min << endl;
}
```

ở code trên hàm tìm min và max dùng cách kinh điển, ta có thể thay đổi bằng cách dùng con trỏ như sau:

```
void TimMinMax(int *a, int n, int &min, int &max){
  min = max = *a;
  for (int i = 1, *p=++a; i < n ; i++, p++){
    if ( *p < min ) min = *p;
    if ( *p > max ) max = *p;
  }
}
```

Cách giải 3: dài dòng nhất+ nhập mảng n phần tử số nguyên từ bàn phím (đã có thuật toán và code ở trên)+ sắp xếp tăng dần (hoặc giảm dần) + nếu sắp xếp tăng dần thì min sẽ là phần tử đầu tiên, và max sẽ là phần tử cuối cùng trong mảng đã sắp xếp.

```
#include <iostream.h>
void NhapMang(int *a, int &n){
  cout << "Nhap N = ";  cin >> n;
  cout << "Nhap mang" << endl;
  for (int i = 0; i < n ; i++){
        cout << "a[" << i << "] = ";
         cin >> *(a+i);
  }
}

void SapXep(int a[], int n) { //sắp xếp tăng dần for (int i = 0; i < n - 1; i++)

    for (int j = i + 1; j < n; j++)
      if ( a[i] > a[j] ){
        int tg  = a[i];
        a[i] = a[j];
        a[j] = tg;
}

}

void main(){
  int a[100], n;
  NhapMang (a, n);
  SapXep (a, n);
  cout << "Gia tri nho nhat = " << a[0]
  cout << "Gia tri lon nhat = " << a[n-1] << endl;
}
```

Đề bài: Nhập n phần tử số nguyên. Nhập phần tử cần tìm kiếm X. Nếu trong n phần tử đã nhập có X thì báo "tìm thấy", "số lần tìm thấy" và "các vị trí tìm thấy", ngược lại báo "không tìm thấy"

Ví dụ: mảng A gồm các phần tử (theo thứ tự chỉ số tăng dần từ 0) là **5, 2, 1, 6, 2, 4, 1, 3** giá trị tìm kiếm **X = 2** vậy kết quả sẽ là: **Tìm thấy 2 (2 lần) tại vị trí: 1, 4**

Phân tích:

sau khi nhập mảng ta sẽ đếm số lượng phần tử X có trong mảng, nếu đếm thấy có: tiến hành liệt kê các vị trí, ngược lại thì thông báo không có. Đơn giản vậy thôi□ta sẽ chia các công việc ra từng hàm riêng hàm nhập kinh điển, hàm đếm trả về số lượng phần tử có giá trị bằng giá trị cho trước hàm liệt kê: tương tự hàm trên, mỗi khi gặp a[i] bằng x thì in vị trí ra (vị trí là i)

```
#include <iostream.h>

void NhapMang(int a[], int &n){
  cout << "Nhap so luong phan tu cua mang: ";  cin >> n;
  for (int i = 0; i < n ; i++){
    cout << "a[" << i << "] = ";  cin >> a[i];
  }
}

int DemSoLan(int a[], int n, int x) {
  int kq = 0;
  for (int i=0; i < n; i++) if (a[i] == x) kq++;
  return kq;
}

void LietKe(int a[], int n, int x) {
  for (int i = 0; i < n; i++) if (a[i] == x) cout << i << " ";
}
void main(){
  int a[100], n, x, d;
  NhapMang (a, n);
  cout << "Gia tri tim kiem = "; cin >> x;
  d == DemSoLan (a, n, x);
  if (d > 0){
    cout << "Tim thay " << x << " (" << d << " lan) tai vi tri: ";
    LietKe (a, n, x);
  }
  else
    cout << "Khong tim thay " << x;
}
```

10.4.2. Các bài thực hành cơ bản và nâng cao

Bài 1: Tính tiền thuê máy dịch vụ Internet và in ra màn hình kết quả với dữ liệu nhập vào là giờ bắt đầu thuê (GBD), giờ kết thúc thuê (GKT), số máy thuê (SoMay).

- Điều kiện cho dữ liệu nhập: 6<=GBD<GKT<=21. Giờ là số nguyên.

- Đơn giá: 2500đ cho mỗi giờ máy trước 17:30 và 3000đ cho mỗi giờ máy sau 17:30.

Bài 2: Nhập số nguyên dương n gồm k chữ số ($0 < k \leq 5$):

- đếm xem n có bao nhiêu chữ số chẵn và bao nhiêu chữ số lẻ.

- đếm xem n có bao nhiêu chữ số là số nguyên tố.
- tính tổng các ước số dương của n.
- tìm ước số lẻ lớn nhất của n.
- kiểm tra xem các chữ số của n có toàn lẻ hay toàn chẵn không.
- sắp xếp các chữ số của n theo thứ tự tăng dần.
- tìm vị trí xuất hiện của chữ số có giá trị x trong n.
- kiểm tra xem các chữ số của n có được sắp thứ tự không.
- tính giá trị trung bình các chữ số chẵn trong n.

Bài 3: Sắp xếp mảng:

- theo thứ tự tăng dần của các phần tử là số nguyên tố.
- các phần tử lẻ tăng dần.
- các phần tử chẵn giảm dần.
- các phần tử chẵn nằm bên trái theo thứ tự tăng dần còn các phần tử lẻ bên phải theo thứ tự giảm dần.
- các phần tử âm giảm dần từ trái sang phải, phần tử dương tăng dần từ phải sang trái.

Bài 4: Xoá:

- phần tử tại vị trí lẻ trong mảng.
- phần tử có giá trị lớn nhất trong mảng.
- xoá tất cả các phần tử có giá trị nhỏ hơn X.
- xoá phần tử có giá trị gần X nhất.

Bài 5: Chèn phần tử có giá trị X vào:

- vị trí đầu tiên của mảng.
- phía sau phần tử có giá trị lớn nhất trong mảng.
- trước phần tử có giá trị là số nguyên tố đầu tiên trong mảng.
- phía sau tất cả các phần tử có giá trị chẵn trong mảng.

Bài 6: Tách 1 mảng các số nguyên thành 2 mảng a và b, sao cho mảng a chứa toàn số lẻ và mảng b chứa toàn số chẵn.

Ví dụ: Mảng ban đầu: 1 3 ***8 2*** *7 5 9 0* ***10***

Mảng a: 1 3 7 5 9

Mảng b: 8 2 10

Bài 7: Cho 2 mảng số nguyên a và b kích thước lần lượt là n và m. Viết chương trình nối 2 mảng trên thành mảng c theo nguyên tắc chẵn ở đầu mảng và lẻ ở cuối mảng.

Ví dụ: Mảng a: 3 2 7 5 9

Mảng b: 1 8 10 4 12 6

Mảng c: ***6 12 4 10 8 2*** *3 7 5 9 1*

Bài 8: Tìm và xóa tất cả các phần tử trùng với x trong mảng một chiều các số nguyên, nếu không tồn tại phần tử x trong mảng thì trả về -1.

Bài 9: Tách 1 mảng các số nguyên thành 2 mảng a và b, sao cho kết quả thu được là: • Mảng a chứa toàn số lẻ tăng dần. • Mảng b chứa toàn số chẵn giảm dần.*(Không dùng sắp xếp)* Hướng dẫn: Tìm vị trí chèn thích hợp khi trích phần tử từ mảng ban đầu.

Ví dụ: Mảng ban đầu: 9 3 ***8 2*** *7 5 1 0* ***10***

Mảng a: 1 3 5 7 9

Mảng b: 10 8 2

Bài 10: *Tổ chức dữ liệu quản lí danh mục các bộ phim VIDEO, các thông tin liên quan đến bộ phim này như sau: - Tên phim (tựa phim).- Thể loại (3 loại : hình sự, tình cảm, hài). - Tên đạo diễn.- Tên điễn viên nam chính.- Tên diễn viên nữ chính.*

- Năm sản xuất.- Hãng sản xuất

Viết chương trình thực hiện những công việc sau:

• Nhập vào bộ phim mới cùng với các thông tin liên quan đến bộ phim này.

• Nhập một thể loại. In ra danh sách các bộ phim thuộc thể loại này.

• Nhập một tên nam diễn viên. In ra các bộ phim có diễn viên này đóng. •

Nhập tên đạo diễn. In ra danh sách các bộ phim do đạo diễn này dàn dựng.

Bài 11: Một cửa hàng cần quản lý các mặt hàng, thông tin một mặt hàng bao gồm: *- Mã hàng.- Tên mặt hàng.- Số lượng.- Đơn giá.- Số lượng tồn.- Thời gian bảo hành (tính theo đơn vị tháng).*

Viết chương trình thực hiện những công việc sau:

• Hãy nhập vào một danh sách các mặt hàng.

• Tìm mặt hàng có số lượng tồn nhiều nhất.

• Tìm mặt hàng có số lượng tồn ít nhất.

• Tìm mặt hàng có giá tiền cao nhất.

• In ra những mặt hàng có thời gian bảo hành lớn hơn 12 tháng.

• Sắp xếp các mặt hàng theo thứ tự tăng dần của số lượng tồn.

Bài 12: Viết chương trình quản lý vé tàu, thông tin một vé tàu như sau :

- Ngày giờ khởi hành, ngày giờ đến.- Ga đi, ga đến.- Loại tàu, loại chỗ ngồi (ngồi, nằm, cứng, mềm). - Số toa, số ghế.

Viết chương trình thực hiện những công việc sau:

• nhập vào danh sách các vé tàu.

• In danh sách các vé tàu có ga đến là Huế.

• In danh sách các vé tàu có ga đến là Hà Nội và đi ngày 8/6/2005.

• Đếm xem có bao nhiêu khách đi tàu loại chỗ ngồi là *nằm cứng*.

Bài 13: Tính tiền điện hàng tháng của các hộ gia đình, thông tin các khách hàng như sau:

- Kỳ thu, từ ngày.....đến ngày.- Tên khách hàng, mã khách hàng. - Địa chỉ.- Điện năng tiêu thụ (Kwh).

Viết chương trình thực hiện những công việc sau:

• Nhập vào danh sách các khách hàng.

• Xuất danh sách hoá đơn theo thứ tự tăng dần của điện năng tiêu thụ.

• Tính tiền điện của các khách hàng theo quy định sau.

- 100 kw đầu tiên là 550 đ / kw

- 50 kw tiếp theo là 900 đ / kw

- 50 kw tiếp theo là 1210 đ / kw

- Thuế 10 % trên tổng số tiền phải trả

• Tính tổng số tiền thu được của các khách hàng.

PHỤ LỤC: HƯỚNG DẪN VIẾT CHƯƠNG TRÌNH TRÊN MÔI TRƯỜNG DEV C++

-----o0o-----

MÔT SỐ LỖI BIÊN DỊCH (Compile – time Errors)

STT	Thông báo lỗi gốc	Ý nghĩa
1	(expected	**Thiếu dấu …** Các lỗi này thường xảy ra khi ta sơ sót, dẫn ñến thiếu các dấu mở hoặc ñóng ngoặc.
2	) expected	
3	, expected	
4	{ expected	
5	} expected	
6	286/287 instructions not enabled	Tập lệnh của bộ Vi xử lí 80286 và bộ xử lí toán học chưa ñược kích hoạt. **!Vào Options/Compiler/Advanced Code** generation… ñể ñiều chỉnh lại
7	Ambiguity between 'function1' and 'function2'	2 hàm function1 và function2 giống nhau, không thể phân biệt ñược.
8	Array bounds missing]	Thiếu dấu ñóng ngoặc] khi truy xuất ñến các phần tử của mảng
9	Array must have at least one element	Khi khai báo mảng phải có ít nhất 1 phần tử. Xảy ra khi khai báo mảng mà SPT tối ña là âm hoặc bằng 0.
10	Array size too large	Kích thước của mảng quá lớn, vượt quá dung lượng vùng nhớ quy ước là 64K
11	Bit field cannot be static	Kiểu dữ liệu bit field không thể có kiểu static
12	Bit field too large	Kích thước của bit field quá lớn
13	Bit fields must be signed or unsigned int	Kiểu dữ liệu của bit field phải là số nguyên
14	Bit fields must contain at least one bit	Kích thước của mỗi bit field phải ≥ 1 bit
15	Body already defined for this function	Hàm ñã ñược ñịnh nghĩa rồi. Lỗi xảy ra khi ta viết

		phần thân của một hàm nào ñó ≥ 2 lần
16	Call of nonfunction	Câu lệnh gọi hàm của ta là sai. Tên hàm mà ta gọi có thể là một tên kiểu/hằng/biến,…
17	Cannot call 'main' from within the program	Không thể gọi thực hiện hàm main() trong chương trình, vì ñây là một hàm ñặc biệt, tự ñộng thực hiện 1 lần trong mỗi lần chạy CT.
18	Cannot cast from 'type1' to 'type2'	Không thể ép kiểu dữ liệu từ kiểu 1 sang kiểu 2
19	Cannot convert 'type1' to 'type2'	Không thể chuyển ñổi kiểu dữ liệu từ kiểu 1 sang kiểu 2
20	Cannot initialize 'type1' with 'type2'	Không thể khởi gán dữ liệu thuộc kiểu 2 cho biến thuộc kiểu 1
21	Cannot modify a const object	Không thể thay đổi giá trị của một hằng số. Xảy ra khi ta thực hiện phép gán giá trị mới cho 1 hằng
22	Case outside of switch	Lệnh CASE nằm bên ngoài SWITCH
23	Case statement missing :	Lệnh CASE thiếu dấu 2 chấm (:)
24	Character constant must be one or two characters long	Kích thước của hằng kí tự không ñúng. Xảy ra khi ta ghi một chuỗi dài các kí tự vào giữa cặp dấu nháy ñơn ‘’
25	Compound statement missing }	Thiếu dấu } kết thúc khối lệnh
26	Constant expression required	Vị trí này lẽ ra phải là một biểu thức hằng, có giá trị không ñổi.
27	Could not find a match for argument(s)	Không tìm thấy ñối số thích hợp.
28	Could not find file 'filename'	Không tìm thấy tập tin
29	Declaration is not allowed here	Vị trí khai báo sai. Không ñược khai báo tại đây.
30	Declaration missing ;	Khai báo thiếu dấu chấm phẩy (;)
31	Declaration syntax error	Khai báo không ñúng cú pháp

32	Declaration terminated incorrectly	Khai báo sai (gần giống lỗi trên)
33	Declaration was expected	Thiếu khai báo
34	Default outside of switch	Lệnh mặc ñịnh DEFAULT nằm bên ngoài khối lệnh SWITCH
35	Default value missing	Thiếu giá trị mặc ñịnh
36	Division by zero	Chia cho 0, lỗi này xảy ra khi mẫu số của một phân số có giá trị bằng 0.
37	do statement must have while	Lệnh do phải ñi với while. Xảy ra khi thiếu while trong câu lệnh do…
38	do-while statement missing (	**Thiếu … trong câu lệnh do…while**
39	do-while statement missing)	
40	do-while statement missing ;	
41	Duplicate case	Lệnh CASE bị trùng, xảy ra khi ta viết 2 dòng case khác nhau nhưng cùng một giá trị như nhau.
42	Expression expected	Vị trí này phải là một biểu thức
43	Expression syntax	Sai cú pháp khi xây dựng biểu thức
44	Extra parameter in call to function	Gọi thực hiện hàm nhưng lại truyền dư tham số
45	File name too long	Tên tập tin quá dài.
46	For statement missing (	**Thiếu … trong câu lệnh for**
47	For statement missing)	
48	For statement missing ;	
49	'function' cannot return a value	Hàm có tên ‘function’ không thể trả về một giá trị, **thông thường vì ta khai báo nó là hàm kiểu void**
50	'function' must be declared with no parameters	Hàm có tên ‘function’ phải ñược khai báo không có tham số, xảy ra khi phần khai báo (prototype) và phần thân hàm không giống nhau về số tham số
51	'function' must be declared with one parameter	Tương tự như lỗi trên
52	'function' must be declared with	

	two parameters	
53	Function 'function' should have a prototype	Hàm có tên ‘function’ cần phải ñược khai báo. Lỗi thường gặp khi trình biên dịch C không hiểu 1 tên hàm nào ñó mà ta sử dụng, có thể do thiếu #include tập tin tiêu ñề tương ứng, hoặc gõ sai tên.
54	Function call missing)	Gọi thực hiện hàm thiếu)
55	Function calls not supported	Không thể gọi hàm dạng này/kiểu này
56	Function should return a value	Hàm cần phải trả về 1 giá trị, xảy ra khi ta khai báo hàm có kiểu trả về nhưng lại thiếu câu lệnh **return…**
57	Goto statement missing label	Dùng lệnh goto mà không có nhãn
58	'identifier' is not a member of struct	Tên … không phải là thành phần của cấu trúc, xảy ra khi ta viết tên thành phần sai
59	'identifier' is not a parameter	Tên … không phải là một tham số
60	Identifier expected	Thiếu tên biến.
61	If statement missing (	Câu lệnh if thiếu mở hay ñóng ngoặc
62	If statement missing)	
63	Illegal character 'character' (0x'value')	Kí tự không hợp lệ, thường xảy ra khi ta biểu diễn các hằng số hệ hexa, nhưng lại sử dụng các chữ cái khác A..F hay a..f
64	Illegal octal digit	Không phải là một số hệ 8 hợp lệ
65	Illegal pointer subtraction	Thực hiện phép trừ không hợp lệ trên con trỏ
66	Illegal use of floating point	Dùng dấu chấm ñộng không ñúng, ví dụ sử dụng phép toán modulo % trên số thực chẳng hạn.
67	Illegal use of pointer	Dùng con trỏ không hợp lệ
68	Implicit conversion of 'type1' to 'type2' not allowed	Không cho phép ngầm chuyển từ kiểu 1 sang kiểu 2
69	Improper use of typedef 'identifier'	Kiểu dữ liệu … ñược sử dụng không ñúng.
70	Incompatible type conversion	Không thể chuyển ñổi kiểu dữ liệu

71	Incorrect number format	Không phải là dữ liệu dạng số, thường xảy ra khi ta gõ các kí tự khác 0..9 trong một dữ liệu kiểu số
72	Incorrect use of default	Dùng DEFAULT không ñúng
73	Invalid use of dot	Dùng dấu chấm (.) không ñúng vị trí
74	Lvalue required	Vế trái của phép gán phải là một tên biến. Lỗi xảy ra khi ta gán giá trị cho một hằng.
75	main must have a return type of int	Hàm main phải trả về 1 giá trị kiểu int
76	Misplaced break	Dùng break ngoài vòng lặp hoặc ngoài SWITCH
77	Misplaced continue	Dùng continue ngoài vòng lặp
78	Misplaced decimal point	Dấu chấm thập phân sai vị trí
79	Misplaced else	Dùng else sai vị trí (thiếu if, …)
80	'new' and 'delete' not supported	Không ñược phép dùng new và delete trong cấp phát vùng nhớ ñộng.
81	No : following the ?	Toán tử ñiều kiện thiếu dấu 2 chấm (:)
82	No file name ending	Không có phần kết thúc tên tập tin
83	No file names given	Không có tên tập tin
84	No type information	Không tìm thấy thông tin gì về kiểu dữ liệu
85	Not an allowed type	Kiểu dữ liệu này không cho phép dùng ở ñây
86	Numeric constant too large	Hằng số có giá trị quá lớn
87	Pointer to structure required on left side of -> or ->*	Xảy ra khi dùng con trỏ cấu trúc không ñúng cách để truy xuất các thành phần của cấu trúc
88	sizeof may not be applied to a bit field	Toán tử sizeof() không dùng cho kiểu bit field
89	sizeof may not be applied to a function	Toán tử sizeof() không dùng cho hàm
90	Size of 'identifier' is unknown or zero	Kích thước của … bằng 0 hoặc không xác ñịnh
91	Size of the type is unknown or zero	Kích thước của kiểu dữ liệu bằng 0 hoặc không xác ñịnh
92	Statement missing ;	Thiếu dấu chấm phẩy (;), thông thường do thiếu

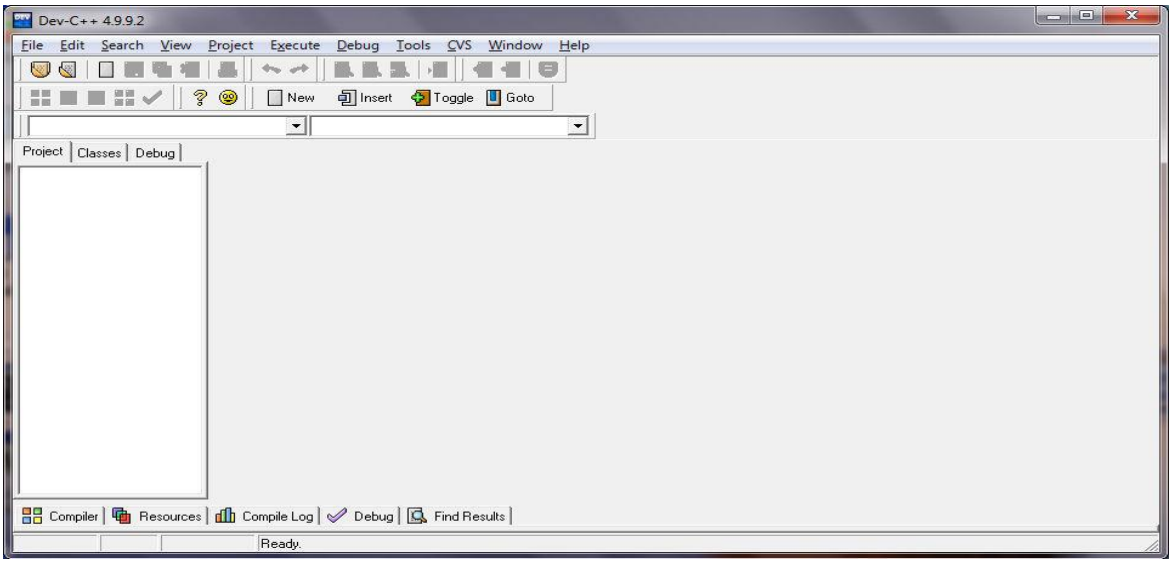
		dấu ; tai dòng trên của dòng báo lỗi
93	Structure required on left side of . or .*	Xảy ra khi truy xuất các thành phần của cấu trúc không đúng cách.
94	Structure size too large	Kích thước của cấu trúc quá lớn, vượt quá giới hạn 64K chẳng hạn.
95	Switch statement missing (	Câu lệnh switch thiếu ngoặc. Phần giá trị của lệnh switch phải ñược ñặt trong cặp dấu ngoặc
96	Switch statement missing)	
97	The value for 'identifier' is not within the range of an int	Giá trị của biến … không nằm trong phạm vi của một biến kiểu nguyên (int)
98	Too few parameters in call to function	Gọi thực hiện hàm nhưng lại truyền không ñủ số lượng tham số.
99	Too many decimal points	Biểu diễn số thực nhưng dùng nhiều hơn 1 dấu chấm thập phân
100	Too many default cases	Trong câu lệnh switch có nhiều hơn 1 lệnh default
101	Too many errors or warning messages	Có quá nhiều lỗi hoặc cảnh báo trong chương trình. Xảy ra khi chương trình có nhiều hơn 25 lỗi
102	Too many types in declaration	Khai báo quá nhiều kiểu dữ liệu mới (ít gặp)
103	Too much global data defined in file	Có quá nhiều biến toàn cục trong chương trình, gây tràn vùng nhớ dành riêng cho các biến này.
104	Type mismatch in default argument value	Giá trị mặc ñịnh của tham số truyền cho CT con bị sai kiểu.
105	Type mismatch in default value for parameter 'parameter'	Giá trị mặc ñịnh của tham số … bị sai kiểu.
106	Type mismatch in parameter 'number' in call to 'function'	Truyền tham số cho chương trình con ‘function’ bị sai kiểu ở tham số ‘number’
107	Type mismatch in parameter 'parameter'	Tham số … bị sai kiểu
108	Type mismatch in parameter 'parameter'	Gần giống lỗi 106

	in call to 'function'	
109	Type name expected	Thiếu tên kiểu tại vị trí báo lỗi
110	Type 'typename' may not be defined here	Kiểu dữ liệu … không thể ñịnh nghĩa ở vị trí này được
111	Unable to create turboc.$ln	Không thể tạo ñược tập tin turboc.1$n Thường xảy ra khi ta chạy TurboC trên ñĩa mềm hay ñĩa CD.
112	Unable to execute command 'command'	Không thể thực hiện lệnh …
113	Unable to open include file 'filename'	Không thể mở ñược tập tin tiêu ñề … thường xảy ra do ta viết tên tập tin tiêu ñề sai, hoặc tập tin này không tồn tại trên ñĩa.
114	Undefined label 'identifier'	Nhãn … chưa ñược khai báo
115	Undefined structure 'structure'	Cấu trúc … chưa ñược khai báo
116	Undefined symbol 'identifier'	Ký hiệu … chưa ñược khai báo, thường xảy ra trong trường hợp ta sử dụng biến mà chưa khai báo.
117	Unexpected }	Dư dấu đóng ngoặc }
118	Unexpected end of file in comment started on 'line number'	Thường xảy ra trong trường hợp thiếu dấu đóng ngoặc } của hàm main().
119	Unexpected end of file in conditional started on 'line number'	
120	Unknown language, must be C or C++	Một cú pháp lạ, không phải là cú pháp của C hay C++
121	User break	Chương trình bị ngắt do người sử dụng
122	Value of type void is not allowed	Không ñược phép gán dữ liệu cho biến kiểu void
123	Variable 'identifier' is initialized more than once	Biến … được khởi tạo nhiều lần.
124	void & is not a valid type	Không chấp nhận tham chiếu ñến biến kiểu void
125	While statement missing (	Câu lệnh while thiếu ngoặc. Phần điều kiện của lệnh while phải ñược ñặt trong dấu ngoặc
126	While statement missing)	

HƯỚNG DẪN SỬ DỤNG PHẦN MỀM DEV C++

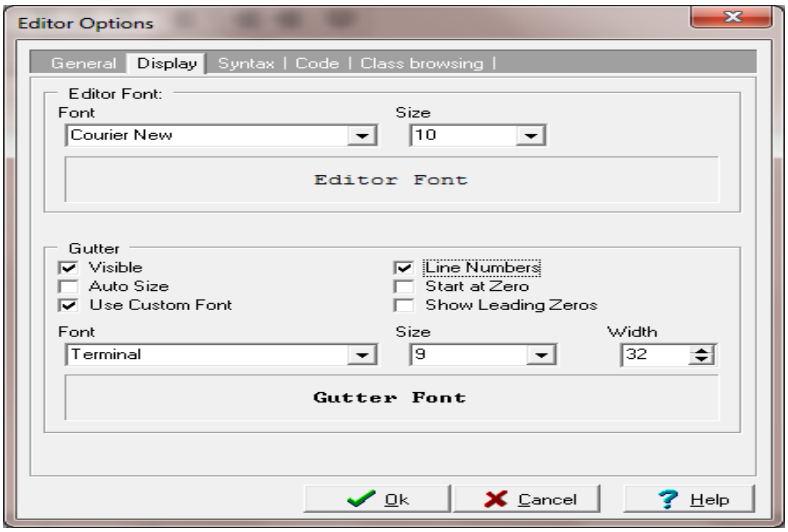
(phiên bản DevC++ Version 4.9.9.2)

1. Giới thiệu DevC++ và cách Compile + Run

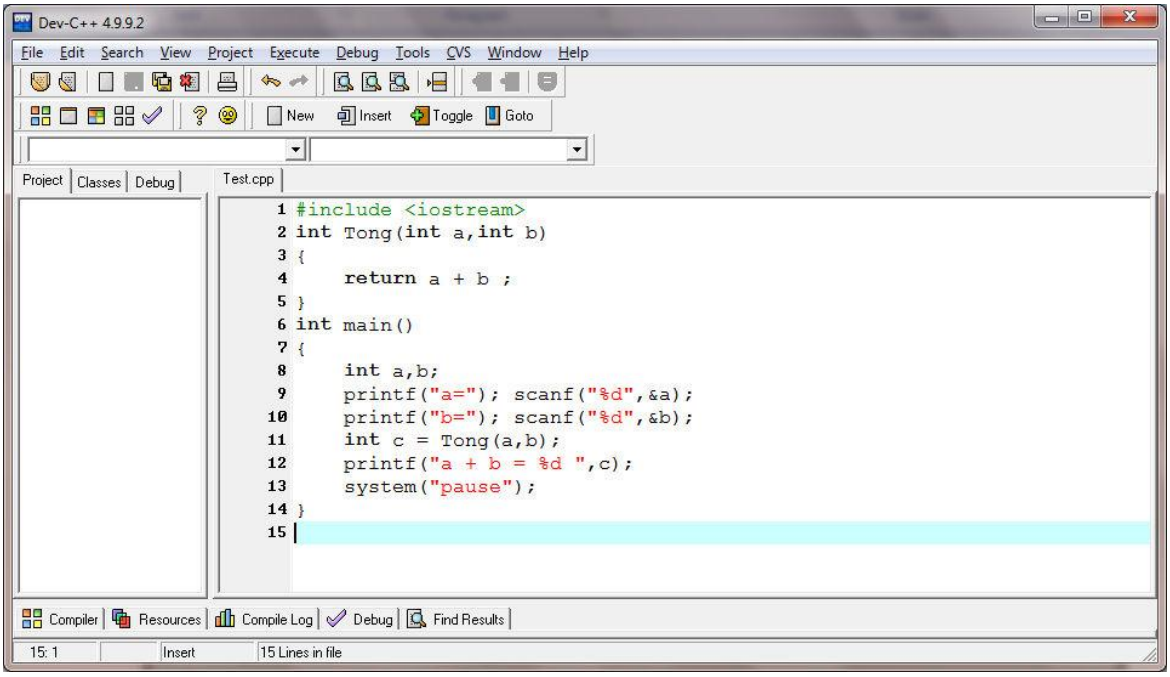


File/New/SourceFile (Ctrl+N tạo ra một cửa sổ lập trình

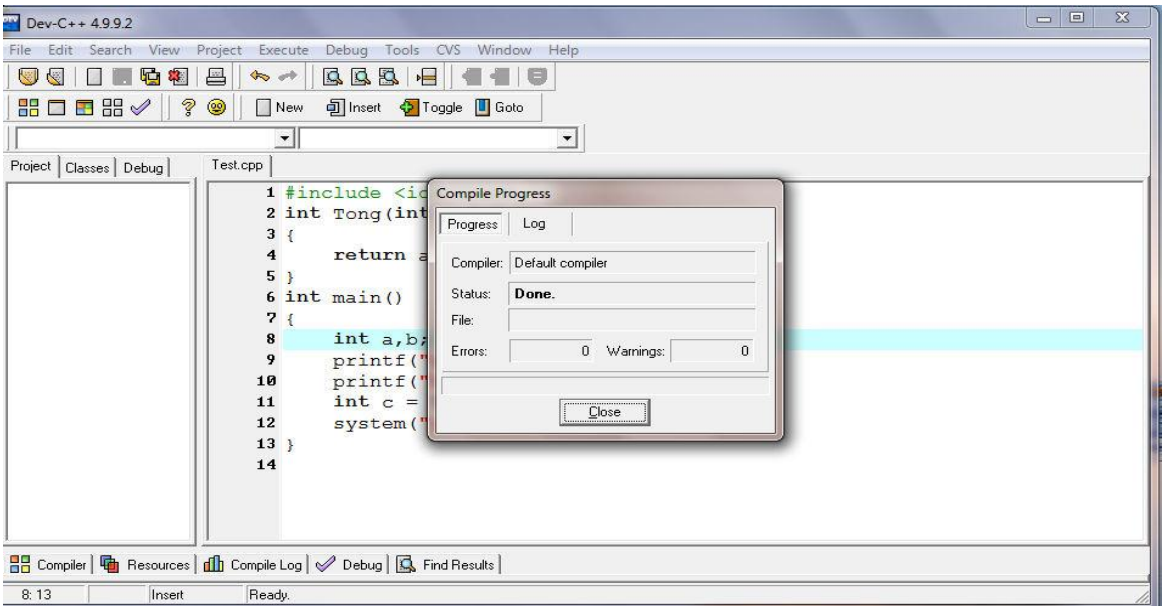
vào Tool / Editor Option chọn Tab Display Click chọn Line Numbers và chọn Size (chọn 12 chẳng hạn) để thuận tiện cho việc nhìn và xem code (chữ lớn hơn và có số dòng)



Đây là hình ảnh code đầy đủ đối với DevC++ các bạn nên sử dụng thư viện như hình, thư viện iostream và lệnh dừng màn hình system(“pause”); nếu dùng thư viện STL thì thêm dòng using namespace std; bên dưới #include<iostream>



Nhấn Ctrl + F9 để dịch (Compile), F9 để dịch và chạy (Compile + Run), F10 để chạy (Run)

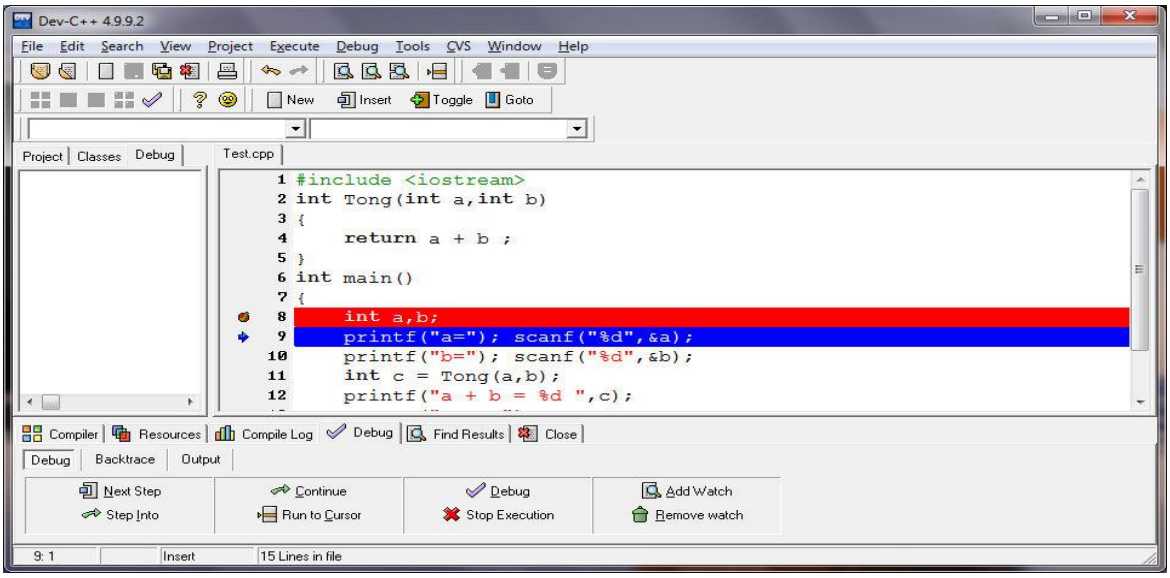


Nhập giá trị của a và b (Kết thúc bằng việc Enter), Màn hình kết quả như sau. (Màn hình DOS – Compile _ Console)

1. Cách Debug với DevC++

Trước tiên bạn phải tạo Breakpoit bằng cách Click vào dòng số 8 hoặc tại dòng 8

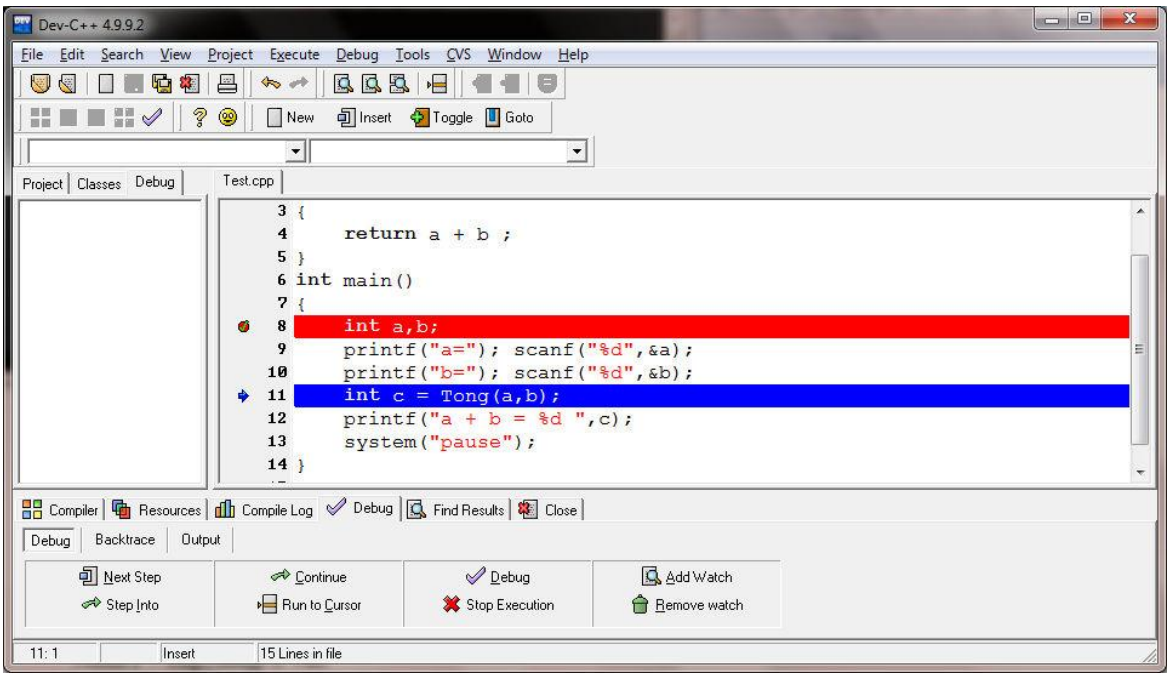
nhấn Ctrl +5(như hình) dưới



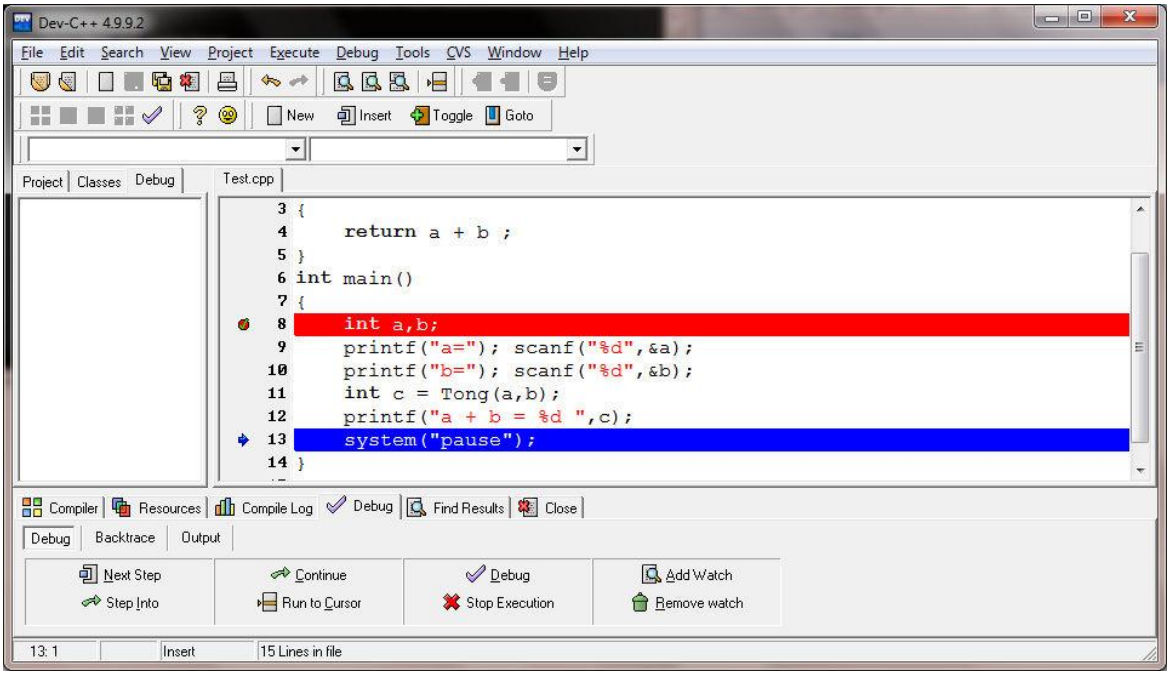
Nhấn F7 và nhập giá trị của a vào -> Enter

Nhấn F7 và nhập giá trị của b vào -> Enter

Đến đây nhấn shift +F7 để nhảy đến hàm con(chương trình con Tong(int a, int b))



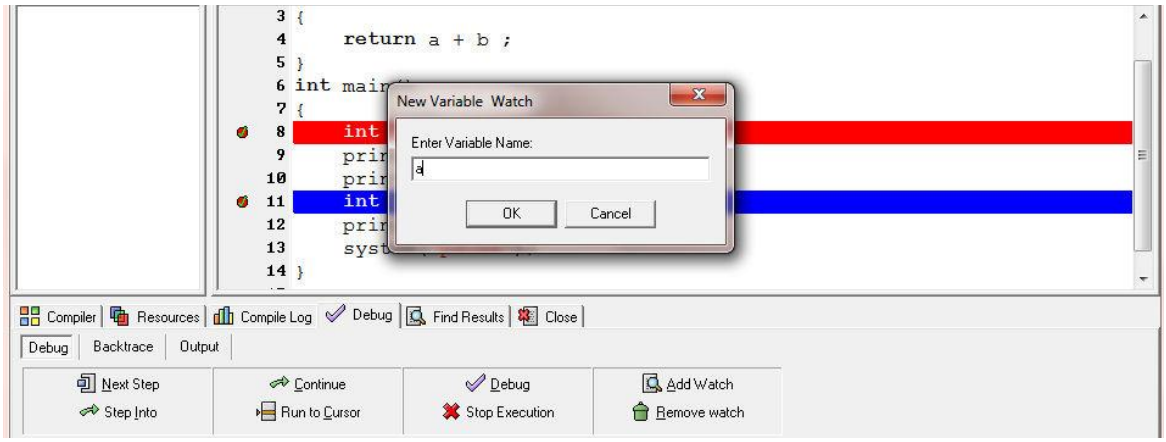
Nhảy đến chương trình con Tong(a,b), tiếp tục nhấn F7 đến khi kết thúc chương trình con nhảy đến thực hiện lệnh sau hàm đã nhảy tới Tong(int a, int b) là dòng 13



Nhấn F8 kết thúc việc debug

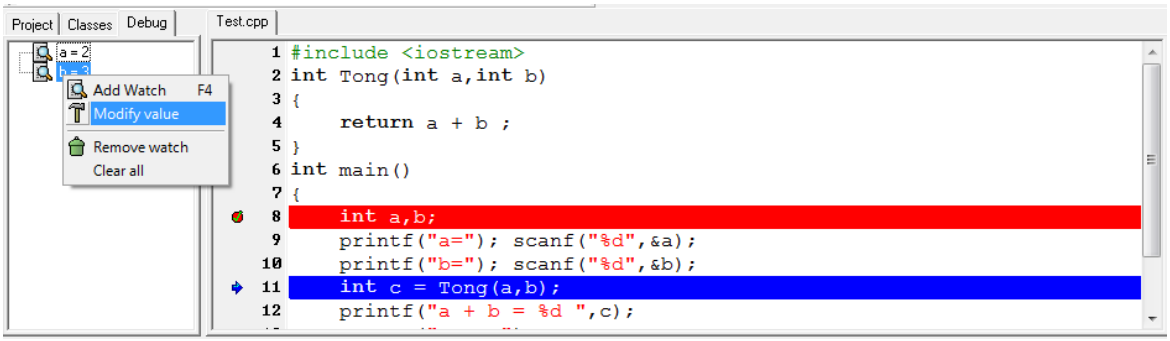
Điểm mạnh của debug là chúng ta có thể xem được giá thị của các biến, hàm, cũng như thay đổi các giá trị đó.

Như ví vụ trên khi đến dòng 11 các bạn có thể nhấn F4, hoặc click chuột vào button Add Wach để xem các biến

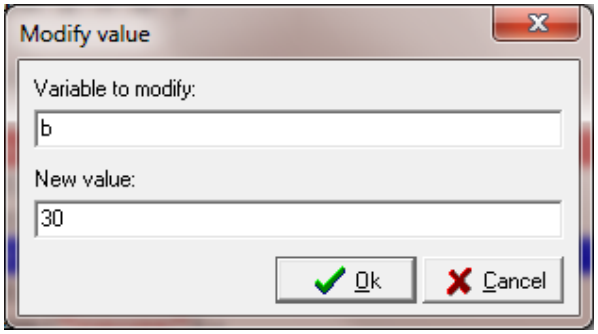


Tương tự như vậy với b và c chúng ta có thể xem được giá trị của chúng.

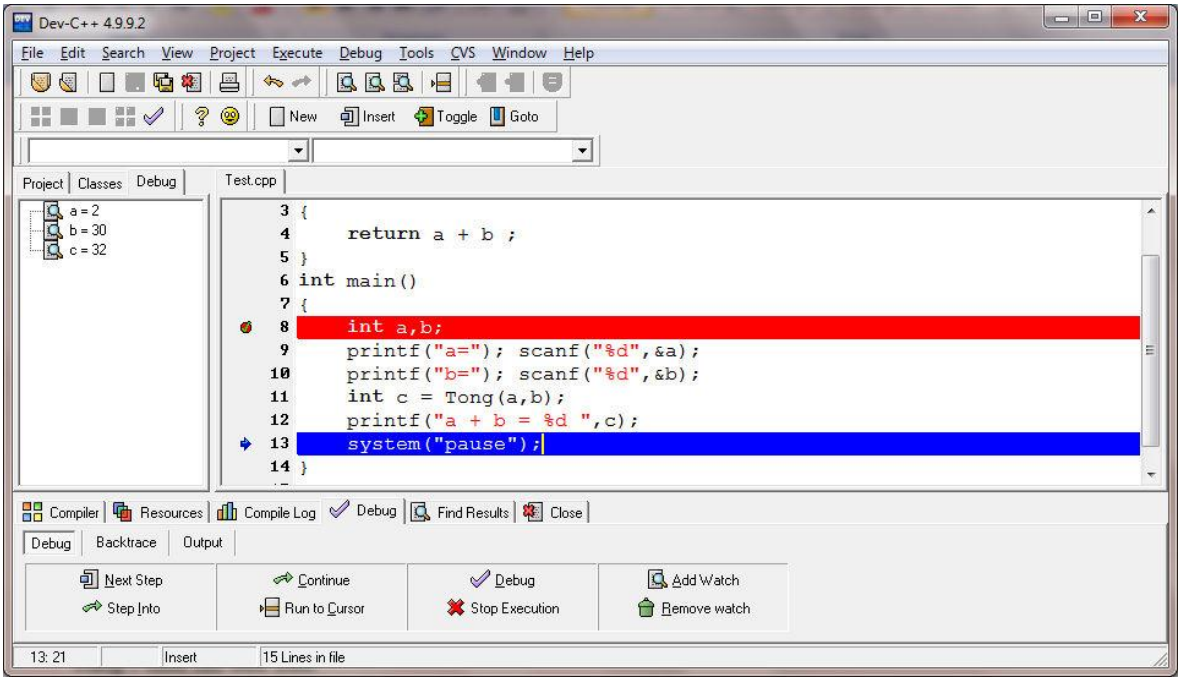
Ngoài ra bạn có thể thay đổi được giá trị của biến, bằng cách Right Click vào biến, chon Modify value (như hình)



Giả sử thay đổi biến b, Right Click vào biến b(lúc này b=3 như hình) ở bên khung trái debug, và thay đổi giá trị của nó, chẳng hạn 3 thành 30 rồi OK



Nhấn F7 để thực hiện câu lệnh tiếp theo, thì lúc này b=30, và c=a+b bây giờ =32 chứ không bằng 5 như lúc đầu nữa.



Ta thấy lúc nhập từ bàn phím là a=2 và b=3 nhưng sau khi debug và thay đổi giá trị của biến ta có kết quả như màn hình DOS là a+b=32.

3. Các phím tắt và ý nghĩa

- Ctrl+N:New -> Tạo 1 File mới
- Ctrl+O:Open -> Mở 1File đã có
- Ctrl+F4:Close -> đóng cửa sổ hiên hành
- Ctrl+Z:Undo ->Quay lại bước trước đó
- Ctrl+Shift+Z:Redo -> Ngược lại với Undo
- Ctrl+F:Find Tìm kiếm Text
- F9:Compile Biêndch
- Ctrl+F9:Compile+Run Biên dịch xong rồi chạy luôn
- Ctrl +F10 : Run Chạy chương trình (Console)
- Ctrl + F5 : Creat Breakpoint Tạo điểm để debug
- F8 : Start Debug
- F7:NextStep nhảy đến dòng lệnh tiếp theo
- Shift + F7 : Step Info nhảy vào hàm (Function) hoặc thủ tục (Procedure)
- F4:AddWatch thêm biến để xem giá trị
- Ctrl + W : Watch Varible Xem giá trị của biến

4. Nhận xét

- Điểm mạnh:

Phầm mềm hoàn toàn miễn phí

Giao diện đẹp dễ sử dụng, được dùng trong kì thi Olympic tin học sinh viên toàn quốc

Sử dụng ngôn ngữ C++ theo chuẩn

Dịch được cả C và C++

- Điểm yếu

  Các hàm, thư viện chưa đầy đủ(không thể bằng được Visual C++)

  Debug còn yếu

  Một số hàm dùng không dùng được giống như Turbo C++(như lệnh xoá màn hình)

5. Lưu ý

- Tải phầm mềm DEV C++: http://www.bloodshed.net/download.html
- Khi Compile các bạn nên để theo mẫu sau:

```
#include <iostream>
using namespace std; // s_ d_ng các th_ vi_n cho STL;
 int Function(các __i s_) // Hàm{
// Kh_i công vi_c }
void Procedure(các __i s_) // Th_ t_c {
// Kh_i công vi_c
}int main() {
// Công vi_c chính
system("pause"); // d_ng màn hình }
```

Đây là code tham khảo theo hướng dẫn:

```
#include <iostream>
using namespace std; int Tong(int a,int b) {
return a + b ; }
int main() {
int a,b;printf("a="); scanf("%d",&a); printf("b="); scanf("%d",&b); int c =
Tong(a,b); printf("a + b = %d ",c); system("pause");
}
```

All C++ Functions

Bitset Constructors (C++ Bitsets)	create new bitsets
Bitset Operators (C++ Bitsets)	compare and assign bitsets
Vector constructors	create vectors and initialize them with some data
Container constructors (C++ Double-ended Queues)	create containers and initialize them with some data
Container constructors (C++ Lists)	create containers and initialize them with some data
Container constructors & destructors (C++ Sets)	default methods to allocate, copy, and deallocate containers
Container constructors & destructors (C++ Multisets)	default methods to allocate, copy, and deallocate multisets
Map constructors & destructors (C++ Maps)	default methods to allocate, copy, and deallocate maps
Multimap constructors & destructors (C++ Multimaps)	default methods to allocate, copy, and deallocate containers
Container operators (C++ Lists)	assign and compare containers
Container operators (C++ Sets)	assign and compare containers
Container operators (C++ Multisets)	assign and compare containers
Multimap operators (C++ Multimaps)	assign and compare containers
Vector operators	compare, assign, and access elements of a vector
Container operators (C++ Double-ended Queues)	compare, assign, and access elements of a container
I/O Constructors (C++ I/O)	constructors
Map operators (C++ Maps)	assign, compare, and access elements of a map
Priority queue constructors (C++ Priority Queues)	construct a new priority queue

Queue constructor (C++ Queues)	construct a new queue
Stack constructors (C++ Stacks)	construct a new stack
String constructors (C++ Strings)	create strings from arrays of characters and other strings
String operators (C++ Strings)	concatenate strings, assign strings, use strings for I/O, compare strings
accumulate (C++ Algorithms)	sum up a range of elements
adjacent_difference (C++ Algorithms)	compute the differences between adjacent elements in a range
adjacent_find (C++ Algorithms)	finds two items that are adjacent to eachother
any (C++ Bitsets)	true if any bits are set
append (C++ Strings)	append characters and strings onto a string
assign (C++ Vectors)	assign elements to a container
assign (C++ Double-ended Queues)	assign elements to a container
assign (C++ Lists)	assign elements to a container
assign (C++ Strings)	give a string values from strings of characters and other C++ strings
at (C++ Vectors)	returns an element at a specific location
at (C++ Double-ended Queues)	returns an element at a specific location
at (C++ Strings)	returns an element at a specific location
auto_ptr (Miscellaneous C++)	create pointers that automatically destroy objects
back (C++ Vectors)	returns a reference to last element of a container
back (C++ Double-ended Queues)	returns a reference to last element of a container
back (C++ Lists)	returns a reference to last element of a container
back (C++ Queues)	returns a reference to last element of a

	container
bad (C++ I/O)	true if an error occurred
begin (C++ Strings)	returns an iterator to the beginning of the container
begin (C++ Vectors)	returns an iterator to the beginning of the container
begin (C++ Double-ended Queues)	returns an iterator to the beginning of the container
begin (C++ Lists)	returns an iterator to the beginning of the container
begin (C++ Sets)	returns an iterator to the beginning of the container
begin (C++ Multisets)	returns an iterator to the beginning of the container
begin (C++ Maps)	returns an iterator to the beginning of the container
begin (C++ Multimaps)	returns an iterator to the beginning of the container
binary_search (C++ Algorithms)	determine if an element exists in a certain range
c_str (C++ Strings)	returns a standard C character array version of the string
capacity (C++ Vectors)	returns the number of elements that the container can hold
capacity (C++ Strings)	returns the number of elements that the container can hold
clear (C++ I/O)	clear and set status flags
clear (C++ Strings)	removes all elements from the container
clear (C++ Vectors)	removes all elements from the container
clear (C++ Double-ended Queues)	removes all elements from the container

clear (C++ Lists)	removes all elements from the container
clear (C++ Sets)	removes all elements from the container
clear (C++ Multisets)	removes all elements from the container
clear (C++ Maps)	removes all elements from the container
clear (C++ Multimaps)	removes all elements from the container
close (C++ I/O)	close a stream
compare (C++ Strings)	compares two strings
copy (C++ Strings)	copies characters from a string into an array
copy (C++ Algorithms)	copy some range of elements to a new location
copy_backward (C++ Algorithms)	copy a range of elements in backwards order
copy_n (C++ Algorithms)	copy N elements
count (C++ Sets)	returns the number of elements matching a certain key
count (C++ Multisets)	returns the number of elements matching a certain key
count (C++ Maps)	returns the number of elements matching a certain key
count (C++ Multimaps)	returns the number of elements matching a certain key
count (C++ Bitsets)	returns the number of set bits
count (C++ Algorithms)	return the number of elements matching a given value
count_if (C++ Algorithms)	return the number of elements for which a predicate is true
data (C++ Strings)	returns a pointer to the first character of a string
empty (C++ Strings)	true if the container has no elements
empty (C++ Vectors)	true if the container has no elements

empty (C++ Double-ended Queues)	true if the container has no elements
empty (C++ Lists)	true if the container has no elements
empty (C++ Sets)	true if the container has no elements
empty (C++ Multisets)	true if the container has no elements
empty (C++ Maps)	true if the container has no elements
empty (C++ Multimaps)	true if the container has no elements
empty (C++ Stacks)	true if the container has no elements
empty (C++ Queues)	true if the container has no elements
empty (C++ Priority Queues)	true if the container has no elements
end (C++ Strings)	returns an iterator just past the last element of a container
end (C++ Vectors)	returns an iterator just past the last element of a container
end (C++ Double-ended Queues)	returns an iterator just past the last element of a container
end (C++ Lists)	returns an iterator just past the last element of a container
end (C++ Sets)	returns an iterator just past the last element of a container
end (C++ Multisets)	returns an iterator just past the last element of a container
end (C++ Maps)	returns an iterator just past the last element of a container
end (C++ Multimaps)	returns an iterator just past the last element of a container
eof (C++ I/O)	true if at the end-of-file
equal (C++ Algorithms)	determine if two sets of elements are the same
equal_range (C++ Sets)	returns iterators to the first and just past the last elements matching a specific key

equal_range (C++ Multisets)	returns iterators to the first and just past the last elements matching a specific key
equal_range (C++ Maps)	returns iterators to the first and just past the last elements matching a specific key
equal_range (C++ Multimaps)	returns iterators to the first and just past the last elements matching a specific key
equal_range (C++ Algorithms)	search for a range of elements that are all equal to a certain element
erase (C++ Strings)	removes elements from a string
erase (C++ Vectors)	removes elements from a container
erase (C++ Double-ended Queues)	removes elements from a container
erase (C++ Lists)	removes elements from a container
erase (C++ Sets)	removes elements from a container
erase (C++ Multisets)	removes elements from a container
erase (C++ Maps)	removes elements from a container
erase (C++ Multimaps)	removes elements from a container
fail (C++ I/O)	true if an error occurred
fill (C++ I/O)	manipulate the default fill character
fill (C++ Algorithms)	assign a range of elements a certain value
fill_n (C++ Algorithms)	assign a value to some number of elements
find (C++ Algorithms)	find a value in a given range
find (C++ Sets)	returns an iterator to specific elements
find (C++ Multisets)	returns an iterator to specific elements
find (C++ Maps)	returns an iterator to specific elements
find (C++ Multimaps)	returns an iterator to specific elements
find (C++ Strings)	find characters in the string
find_end (C++ Algorithms)	find the last sequence of elements in a certain range
find_first_not_of (C++ Strings)	find first absence of characters

find_first_of (C++ Strings)	find first occurrence of characters
find_first_of (C++ Algorithms)	search for any one of a set of elements
find_if (C++ Algorithms)	find the first element for which a certain predicate is true
find_last_not_of (C++ Strings)	find last absence of characters
find_last_of (C++ Strings)	find last occurrence of characters
flags (C++ I/O)	access or manipulate io stream format flags
flip (C++ Bitsets)	reverses the bitset
flush (C++ I/O)	empty the buffer
for_each (C++ Algorithms)	apply a function to a range of elements
front (C++ Vectors)	returns a reference to the first element of a container
front (C++ Double-ended Queues)	returns a reference to the first element of a container
front (C++ Lists)	returns a reference to the first element of a container
front (C++ Queues)	returns a reference to the first element of a container
gcount (C++ I/O)	number of characters read during last input
generate (C++ Algorithms)	saves the result of a function in a range
generate_n (C++ Algorithms)	saves the result of N applications of a function
get (C++ I/O)	read characters
getline (C++ I/O)	read a line of characters
getline (C++ Strings)	read data from an I/O stream into a string
good (C++ I/O)	true if no errors have occurred
ignore (C++ I/O)	read and discard characters
includes (C++ Algorithms)	returns true if one set is a subset of another
inner_product (C++ Algorithms)	compute the inner product of two ranges of

	elements
inplace_merge (C++ Algorithms)	merge two ordered ranges in-place
insert (C++ Strings)	insert characters into a string
insert (C++ Vectors)	inserts elements into the container
insert (C++ Double-ended Queues)	inserts elements into the container
insert (C++ Lists)	inserts elements into the container
insert (C++ Sets)	insert items into a container
insert (C++ Multisets)	inserts items into a container
insert (C++ Multimaps)	inserts items into a container
insert (C++ Maps)	insert items into a container
is_heap (C++ Algorithms)	returns true if a given range is a heap
is_sorted (C++ Algorithms)	returns true if a range is sorted in ascending order
iter_swap (C++ Algorithms)	swaps the elements pointed to by two iterators
key_comp (C++ Sets)	returns the function that compares keys
key_comp (C++ Multisets)	returns the function that compares keys
key_comp (C++ Maps)	returns the function that compares keys
key_comp (C++ Multimaps)	returns the function that compares keys
length (C++ Strings)	returns the length of the string
lexicographical_compare (C++ Algorithms)	returns true if one range is lexicographically less than another
lexicographical_compare_3way (C++ Algorithms)	determines if one range is lexicographically less than or greater than another
lower_bound (C++ Sets)	returns an iterator to the first element greater than or equal to a certain value
lower_bound (C++ Multisets)	returns an iterator to the first element greater than or equal to a certain value
lower_bound (C++ Maps)	returns an iterator to the first element greater than or equal to a certain value

lower_bound (C++ Multimaps)	returns an iterator to the first element greater than or equal to a certain value
lower_bound (C++ Algorithms)	search for the first place that a value can be inserted while preserving order
make_heap (C++ Algorithms)	creates a heap out of a range of elements
max (C++ Algorithms)	returns the larger of two elements
max_element (C++ Algorithms)	returns the largest element in a range
max_size (C++ Strings)	returns the maximum number of elements that the container can hold
max_size (C++ Vectors)	returns the maximum number of elements that the container can hold
max_size (C++ Double-ended Queues)	returns the maximum number of elements that the container can hold
max_size (C++ Lists)	returns the maximum number of elements that the container can hold
max_size (C++ Sets)	returns the maximum number of elements that the container can hold
max_size (C++ Multisets)	returns the maximum number of elements that the container can hold
max_size (C++ Maps)	returns the maximum number of elements that the container can hold
max_size (C++ Multimaps)	returns the maximum number of elements that the container can hold
merge (C++ Lists)	merge two lists
merge (C++ Algorithms)	merge two sorted ranges
min (C++ Algorithms)	returns the smaller of two elements
min_element (C++ Algorithms)	returns the smallest element in a range
mismatch (C++ Algorithms)	finds the first position where two ranges differ
next_permutation (C++ Algorithms)	generates the next greater lexicographic

	permutation of a range of elements
none (C++ Bitsets)	true if no bits are set
nth_element (C++ Algorithms)	put one element in its sorted location and make sure that no elements to its left are greater than any elements to its right
open (C++ I/O)	create an input stream
partial_sort (C++ Algorithms)	sort the first N elements of a range
partial_sort_copy (C++ Algorithms)	copy and partially sort a range of elements
partial_sum (C++ Algorithms)	compute the partial sum of a range of elements
partition (C++ Algorithms)	divide a range of elements into two groups
peek (C++ I/O)	check the next input character
pop (C++ Stacks)	removes the top element of a container
pop (C++ Queues)	removes the top element of a container
pop (C++ Priority Queues)	removes the top element of a container
pop_back (C++ Vectors)	removes the last element of a container
pop_back (C++ Double-ended Queues)	removes the last element of a container
pop_back (C++ Lists)	removes the last element of a container
pop_front (C++ Double-ended Queues)	removes the first element of the container
pop_front (C++ Lists)	removes the first element of the container
pop_heap (C++ Algorithms)	remove the largest element from a heap
precision (C++ I/O)	manipulate the precision of a stream
prev_permutation (C++ Algorithms)	generates the next smaller lexicographic permutation of a range of elements
push (C++ Stacks)	adds an element to the top of the container
push (C++ Queues)	adds an element to the end of the container

TÀI LIỆU THAM KHẢO

[1] **PHẠM VĂN ẤT**: "Kỹ thuật lập trình C: cơ sở và nâng cao". Nhà Xuất Bản Khoa Học Kỹ Thuật – 1996.

[2] **OHN R. HUBBARD**: "455 Bài tập cấu trúc dữ liệu cài đặt bằng C++". Bản dịch của Minh Trung, Gia Việt – Nhà Xuất Bản Thống Kê.

[3] **SANFORD LEESTMA LARRY NYHOFF**: "Pascal Programming and Solving". Macmillan Publishing Company