ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KIỂU CẤU TRÚC

(Bản thảo ngày 14/05/2022)

Nguyễn Tần Trần Minh Khang
Võ Duy Nguyên

THÀNH PHỐ HỒ CHÍ MINH 2019

MỤC LỤC

CHƯƠNG 01. TRỪU TƯỢNG HÓA DỮ LIỆU

01.01 PHƯƠNG PHÁP LUẬN GIẢI QUYẾT BÀI TOÁN BẰNG PHƯƠNG PHÁP TRỪU TƯỢNG HÓA DỮ LIỆU

Phương pháp luận để giải quyết một bài toán hay một vấn đề bằng phương pháp lập trình hướng thủ tục hàm:

- Bước 1: Xác định các kiểu dữ liệu. Trong bước này ta phải xác định xem trong bài toán (vấn đề) phải làm việc với những kiểu dữ liệu nào.
  - + Kiểu dữ liệu nào đã có sẵn.
  - + Kiểu dữ liệu nào phải định nghĩa mới.
- Bước 2: Thiết kế hàm.
  - + Xác định xem trong bài toán mà ta giải quyết cần phải có bao nhiêu hàm, tên của các hàm ra sao, các tham số đầu vào, kiểu dữ liệu trả như thế nào.
  - + Quan trọng hơn nữa là các tham số thì tham số nào là tham trị và tham số nào là tham biến.
- Bước 3: Định nghĩa hàm.
  - + Trong bước này ta sẽ tiến hành định nghĩa các hàm đã thiết kế và khai báo trong bước 2.
  - + Hơn nữa ta phải định nghĩa hàm main với các khai báo biến và thực hiện các lời gọi hàm cần thiết cho chương trình.
- Lưu ý: Bước 1 và Bước 2 là hai bước quan trọng nhất.

01.02 KHÁI NIỆM KIỂU CẤU TRÚC

- Khái niệm: Kiểu dữ liệu cấu trúc là một phương pháp tích hợp các kiểu dữ liệu đơn thành kiểu dữ liệu phức nhằm mô tả biểu diễn thông tin của một khái niệm hay một đối tượng trong thế giới thực.

01.03 CÚ PHÁP

```
00001.  struct <kieudulieu>
00002.  {
```

```
00003.     Thành phần 1;
00004.     Thành phần 2;
00005.     ...
00006.  };
00007.  typedef struct <kieudulieu> <KIEUDULIEU>;
```

01.04 NHÌN MỘT BÀI TOÁN DƯỚI SỰ VẬN ĐỘNG VÀ PHÁT TRIỂN TRONG MÔN HỌC NHẬP MÔN LẬP TRÌNH

Đặt vấn đề: Viết chương trình nhập họ tên, điểm toán, điểm văn của một học sinh. Tính điểm trung bình và xuất kết quả.

01.04.01 Viết chương trình với duy nhất hàm main và sử dụng biến toàn cục

- Khái niệm: Biến toàn cục (global variable) là biến được khai báo bên ngoài tất cả các hàm và được hiểu bên trong tất cả các hàm.
- Thông thường biến toàn cục được khai báo ở đầu chương trình.

Bài cơ sở 001. Viết chương trình nhập họ tên, điểm toán, điểm văn của một học sinh. Tính điểm trung bình và xuất kết quả.

- Chương trình

```
00008.  #include <iostream>
00009.  #include <string>
00010.  using namespace std;
00011.
00012.  string HoTen;
00013.  int Toan;
00014.  int Van;
00015.  float DiemTrungBinh;
00016.
00017.  int main()
00018.  {
00019.     cout << "Nhap ho va ten: ";
00020.     getline(cin, HoTen);
00021.     cout << "Nhap diem toan: ";
00022.     cin >> Toan;
00023.     cout << "Nhap diem van: ";
00024.     cin >> Van;
00025.
00026.     DiemTrungBinh = (float)(Toan + Van) / 2;
00027.
```

```
00028.     cout << "Ho ten: " << HoTen << endl;
00029.     cout << "Diem toan: " << Toan << endl;
00030.     cout << "Diem van: " << Van << endl;
00031.     cout << "Diem trung binh: ";
00032.     cout << DiemTrungBinh << endl;
00033.     return 1;
00034. }
```

- Trong chương trình trên các biến `HoTen, Toan, Van, DiemTrungBinh` được khai báo ở đầu chương trình, bên ngoài tất cả các hàm và đóng vai trò là biến toàn cục trong chương trình.
- Nhắc lại: Biến toàn cục (global variable) là biến được khai báo bên ngoài tất cả các hàm và được hiểu bên trong tất cả các hàm.

Chương trình trên ta có thể gặp các câu hỏi sau:

```
00035. include là gì?
00036. iostream là gì?
00037. string là gì?
00038. using là gì?
00039. namespace là gì?
00040. std là gì?
00041. string là gì?
00042. int là gì?
00043. float là gì?
00044. main là gì?
00045. cout là gì?
00046. << là gì:
00047. getline là gì?
00048. cin là gì?
00049. >> là gì:
00050. endl là gì?
00051. return là gì?
00052. Tại sao phải có hàm main?
00053. Tại sao kiểu dữ liệu trả về của hàm main là int?
00054. Tại sao phải return 1?
00055. Hàm getline thuộc thư viện nào?
```

Trả lời được các câu hỏi trên thể hiện bạn có nền tảng vững về nhập môn lập trình C/C++.

Để học tiếp các kiến thức Cấu trúc Dữ liệu và các kiến thức trong Phương pháp Lập trình Hướng đối tượng các bạn cần hiểu rõ và nắm vững thêm khái niệm Con trỏ và khái niệm Đệ quy là bạn có một hành trang tốt bước vào hai thế giới đó.

Giải thích từng dòng code chương trình – Nội dung này dành cho các bạn không có nền tảng lập trình, dành cho các bạn chỉ trả lời được một phần trong các câu hỏi trên. Các bạn đã vững kiến thức vui lòng bỏ qua.

```
00056.  #include <iostream>
```

- Dòng lệnh trên đọc là khai báo sử dụng thư viện `iostream`.
- Thư viện `iostream` là thư viện được xây dựng sẵn trong các Compiler của ngôn ngữ C/C++.

```
00057.  #include <string>
```

- Dòng lệnh trên đọc là khai báo sử dụng thư viện `string`.
- Thư viện `string` là thư viện được xây dựng sẵn trong các Compiler của ngôn ngữ C/C++.

```
00058.  using namespace std;
```

- Dòng lệnh trên đọc là khai báo sử dụng không gian tên `std`.

```
00059.  string HoTen;
```

- Dòng lệnh trên đọc là `HoTen` là biến có kiểu string.
- Ngoài ra có một cách đọc khác chuẩn hơn: `HoTen` là đối tượng thuộc lớp đối tượng `string`.
- Lớp đối tượng `string` là lớp đối tượng được xây dựng sẵn trong ngôn ngữ C/C++.

```
00060.  int Toan;
```

- Dòng lệnh trên đọc là `Toan` là biến có kiểu int.
- Kiểu dữ liệu `int` là kiểu dữ liệu được xây dựng sẵn trong ngôn ngữ C/C++.

```
00061.  int Van;
```

- Dòng lệnh trên đọc là `Van` là biến có kiểu int.
- Kiểu dữ liệu `int` là kiểu dữ liệu được xây dựng sẵn trong ngôn ngữ C/C++.

```
00062.  float DiemTrungBinh;
```

- Dòng lệnh trên đọc là `DiemTrungBinh` là biến có kiểu float.
- Kiểu dữ liệu `float` là kiểu dữ liệu được xây dựng sẵn trong ngôn ngữ C/C++.

```
00063.      cout << "Nhap ho va ten: ";
```

- Dòng lệnh trên đọc là: Xuất câu thông báo `"Nhap ho va ten: "`.
- `cout` là đối tượng thuộc lớp đối tượng `ostream`.
- Ký tự `<<` hiểu đơn giản là toán tử xuất, toán tử ra.

```
00064.      getline(cin, HoTen);
```

- Dòng lệnh trên đọc là: Hàm `getline` được gọi thực hiện với hai đối số là `cin` và `HoTen`.

```
00065.    cout << "Nhap diem toan: ";
```

- Dòng lệnh trên đọc là: Xuất câu thông báo "Nhap diem toan: ".

```
00066.    cin >> Toan;
```

- Dòng lệnh trên đọc là: Nhập điểm Toan.
- cin là đối tượng thuộc lớp đối tượng istream.
- Ký tự >> hiểu đơn giản là toán tử nhập, toán tử vào.

```
00067.    cout << "Nhap diem van: ";
```

- Dòng lệnh trên đọc là: Xuất câu thông báo "Nhap diem van: ".

```
00068.    cin >> Van;
```

- Dòng lệnh trên đọc là: Nhập điểm Van.
- cin là đối tượng thuộc lớp đối tượng istream.
- Ký tự >> hiểu đơn giản là toán tử nhập, toán tử vào.

```
00069.    DiemTrungBinh = (float)(Toan + Van) / 2;
```

- Dòng lệnh trên đọc là: Tính giá trị cho biến DiemTrungBinh.
- Phân tích về kiểu dữ liệu trong câu lệnh trên:
  - + Biến Toan có kiểu int, biến Van có kiểu int.
  - + Do đó, biểu thức (Toan+Van) về mặt kiểu dữ liệu giá trị kết quả có kiểu dữ liệu là int.
  - + Sau đó, câu lệnh (float)(Toan + Van) sẽ ép kiểu dữ liệu của giá trị kết quả thành kiểu số thực float.
  - + Cuối cùng, câu lệnh (float)(Toan + Van)/2 sẽ chia giá trị kết quả sau khi ép kiểu cho số nguyên 2.
  - + Kết quả cuối cùng xét về kiểu dữ liệu là kiểu số thực.

```
00070.    cout << "Ho ten: " << HoTen << endl;
```

- Dòng lệnh trên đọc là: Xuất thông báo "Ho ten: " và xuất giá trị của biến HoTen.
- Sau khi xuất xong xuống dòng.
- cout là đối tượng thuộc lớp đối tượng ostream.
- Ký tự << hiểu đơn giản là toán tử xuất, toán tử ra.

```
00071.    cout << "Diem toan: " << Toan << endl;
```

- Dòng lệnh trên đọc là: Xuất thông báo "Diem toan: " và xuất giá trị của biến Toan.
- Sau khi xuất xong xuống dòng.
- cout là đối tượng thuộc lớp đối tượng ostream.
- Ký tự << hiểu đơn giản là toán tử xuất, toán tử ra.

```
00072.    cout << "Diem van: " << Van << endl;
```

- Dòng lệnh trên đọc là: Xuất thông báo "Diem van: " và xuất giá trị của biến Van.
- Sau khi xuất xong xuống dòng.
- cout là đối tượng thuộc lớp đối tượng ostream.
- Ký tự << hiểu đơn giản là toán tử xuất, toán tử ra.

```
00073.     cout << "Diem trung binh: ";
00074.     cout << DiemTrungBinh << endl;
```

- Dòng lệnh trên đọc là: Xuất thông báo "Diem trung binh: " và xuất giá trị của biến DiemTrungBinh.
- Sau khi xuất xong xuống dòng.
- cout là đối tượng thuộc lớp đối tượng ostream.
- Ký tự << hiểu đơn giản là toán tử xuất, toán tử ra.

```
00075.     return 1;
```

- Dòng lệnh trên đọc là: Kết thúc lời gọi hàm và trả về giá trị 1.
- Hàm trong ngữ cảnh của câu lệnh này là hàm main.
- Khối lệnh

```
00076. {
00077.     …
00078. }
```

- Trong ngôn ngữ lập trình C/C++ một khối lệnh (*block*) bắt đầu bởi ký tự { và kết thúc bởi ký tự }.
- Thông thường, khi định nghĩa một hàm (hàm main là trường hợp trong đó) ta bắt đầu bởi ký tự { và kết thúc bởi ký tự }.
- Ngoài ra, khi định nghĩa một thân của một vòng lặp có nhiều hơn một câu lệnh ta bắt đầu bởi ký tự { và kết thúc bởi ký tự }.

01.04.02 Viết chương trình với duy nhất hàm main và sử dụng biến cục bộ

- Khái niệm: Biến cục bộ (local variable) là biến được khai báo và được hiểu bên trong một phạm vi (scope) nào đó của chương trình, ra khỏi phạm vi này biến không còn được biết đến nữa vì không gian bộ nhớ cấp phát cho biến được tự động thu hồi.
- Thông thường biến cục bộ được khai báo bên trong thân của một hàm (function) hay một khối lệnh (block).
- Trong ngôn ngữ C khối lệnh (block) được bắt đầu bằng ký tự { và kết thúc bằng ký tự }.
- Lưu ý: Một biến được khai báo bên trong thân hàm main là biến cục bộ của hàm main.

Bài cơ sở 002. Viết chương trình nhập họ tên, điểm toán, điểm văn của một học sinh. Tính điểm trung bình và xuất kết quả.

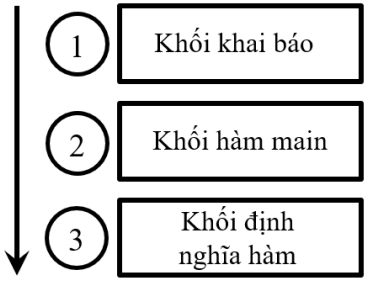
– Chương trình

```
00079.  #include <iostream>
00080.  #include <string>
00081.  using namespace std;
00082.
00083.  int main()
00084.  {
00085.      string HoTen;
00086.      int Toan;
00087.      int Van;
00088.      float DiemTrungBinh;
00089.
00090.      cout << "Nhap ho va ten: ";
00091.      getline(cin, HoTen);
00092.      cout << "Nhap diem toan: ";
00093.      cin >> Toan;
00094.      cout << "Nhap diem van: ";
00095.      cin >> Van;
00096.
00097.      DiemTrungBinh = (float)(Toan + Van) / 2;
00098.
00099.      cout << "Ho ten: " << HoTen << endl;
00100.      cout << "Diem toan: " << Toan << endl;
00101.      cout << "Diem van: " << Van << endl;
00102.      cout << "Diem trung binh: ";
00103.      cout << DiemTrungBinh << endl;
00104.      return 1;
00105.  }
```

– Trong chương trình trên các biến `HoTen`, `Toan`, `Van`, `DiemTrungBinh` được khai báo bên trong thân hàm `main` và đóng vai trò là biến cục bộ của hàm `main`.

01.04.03 Viết chương trình với kỹ thuật lập trình hàm và sử dụng biến toàn cục

– Kiến trúc của một chương trình C cơ bản bao gồm 3 khối lệnh chính như sau: khối khai báo, khối hàm main và khối định nghĩa hàm. Ba khối lệnh này được trình bày theo thứ tự của hình vẽ bên dưới.



- Khối khai báo (declaration–block): chứa các khai báo hàm, khai báo biến toàn cục, khai báo sử dụng thư viện, khai báo hằng, khai báo kiểu dữ liệu…
- Khối hàm main (main–block): chứa duy nhất hàm main và thân hàm của nó. Trong thân hàm main chứa các lời gọi hàm cần thiết cho chương trình.
- Khối định nghĩa hàm (function–definition–block): chứa các định nghĩa hàm đã được khai báo trong khối khai báo.

Bài cơ sở 003. Viết chương trình nhập họ tên, điểm toán, điểm văn của một học sinh. Tính điểm trung bình và xuất kết quả.

- Chương trình

```
00106.  #include <iostream>
00107.  #include <string>
00108.  using namespace std;
00109.
00110.  string HoTen;
00111.  int Toan;
00112.  int Van;
00113.  float DiemTrungBinh;
00114.
00115.  void Nhap();
00116.  void Xuat();
00117.  void Xuly();
00118.
00119.  int main()
00120.  {
00121.     Nhap();
00122.     Xuly();
00123.     Xuat();
00124.     return 1;
00125.  }
00126.
```

```
00127.  void Nhap()
00128.  {
00129.     cout << "Nhap ho ten: ";
00130.     getline(cin, HoTen);
00131.     cout << "Nhap diem toan: ";
00132.     cin >> Toan;
00133.     cout << "Nhap diem van: ";
00134.     cin >> Van;
00135.  }
00136.
00137.  void XuLy()
00138.  {
00139.     DiemTrungBinh = (float)(Toan + Van) / 2;
00140.  }
00141.
00142.  void Xuat()
00143.  {
00144.     cout << "Ho ten: " << HoTen << endl;
00145.     cout << "Diem toan: " << Toan << endl;
00146.     cout << "Diem van: " << Van << endl;
00147.     cout << "Diem trung binh: ";
00148.     cout << DiemTrungBinh << endl;
00149.  }
```

Ba khối lệnh chính trong chương trình trên:

- Khối khai báo: (declaration–block): chứa các khai báo hàm, khai báo biến toàn cục, khai báo sử dụng thư viện, khai báo hằng, khai báo kiểu dữ liệu…

```
00150.  #include <iostream>
00151.  #include <string>
00152.  using namespace std;
00153.
00154.  string HoTen;
00155.  int Toan;
00156.  int Van;
00157.  float DiemTrungBinh;
00158.
00159.  void Nhap();
00160.  void Xuat();
00161.  void Xuly();
```

– Khối hàm main (main–block): chứa duy nhất hàm main và thân hàm của nó. Trong thân hàm main chứa các lời gọi hàm cần thiết cho chương trình.

```
00162.  int main()
00163.  {
00164.     Nhap();
00165.     Xuly();
00166.     Xuat();
00167.     return 1;
00168.  }
```

+ Dòng lệnh thứ nhất trong thân hàm main đọc là: Hàm `Nhap` được gọi thực hiện.
+ Dòng lệnh thứ hai trong thân hàm main đọc là: Hàm `XuLy` được gọi thực hiện.
+ Dòng lệnh thứ ba trong thân hàm main đọc là: Hàm `Xuat` được gọi thực hiện.

– Khối định nghĩa hàm (function–definition–block): chứa các định nghĩa hàm đã được khai báo trong khối khai báo.

```
00169.  void Nhap()
00170.  {
00171.     cout << "Nhap ho ten: ";
00172.     getline(cin, HoTen);
00173.     cout << "Nhap diem toan: ";
00174.     cin >> Toan;
00175.     cout << "Nhap diem van: ";
00176.     cin >> Van;
00177.  }
00178.
00179.  void XuLy()
00180.  {
00181.     DiemTrungBinh = (float)(Toan + Van) / 2;
00182.  }
00183.
00184.  void Xuat()
00185.  {
00186.     cout << "Ho ten: " << HoTen << endl;
00187.     cout << "Diem toan: " << Toan << endl;
00188.     cout << "Diem van: " << Van << endl;
00189.     cout << "Diem trung binh: ";
00190.     cout << DiemTrungBinh << endl;
00191.  }
```

Chi tiết các khối lệnh trong chương trình trên:

- Các dòng dưới đây là khai báo sử dụng các thư viện `iostream`, `string` và không gian tên `std`.

```
00192. #include <iostream>
00193. #include <string>
00194. using namespace std;
```

- Các dòng dưới đây là khai báo biến toàn cục trong chương trình. Các biến này sẽ lưu trữ dữ liệu của chương trình.

```
00195. string HoTen;
00196. int Toan;
00197. int Van;
00198. float DiemTrungBinh;
```

- Các dòng dưới đây là khai báo các hàm trong chương trình.

```
00199. void Nhap();
00200. void Xuat();
00201. void XuLy();
```

- Các dòng dưới đây là định nghĩa hàm `main`.

```
00202. int main()
00203. {
00204.    Nhap();
00205.    XuLy();
00206.    Xuat();
00207.    return 1;
00208. }
```

- Các dòng dưới đây là định nghĩa hàm `Nhap`.

```
00209. void Nhap()
00210. {
00211.    cout << "Nhap ho ten: ";
00212.    getline(cin, HoTen);
00213.    cout << "Nhap diem toan: ";
00214.    cin >> Toan;
00215.    cout << "Nhap diem van: ";
00216.    cin >> Van;
00217. }
```

- Các dòng dưới đây là định nghĩa hàm `XuLy`.

```
00218. void XuLy()
00219. {
00220.    DiemTrungBinh = (float)(Toan + Van) / 2;
00221. }
```

- Các dòng dưới đây là định nghĩa hàm `Xuat`.

```
00222.  void Xuat()
00223.  {
00224.      cout << "Ho ten: " << HoTen << endl;
00225.      cout << "Diem toan: " << Toan << endl;
00226.      cout << "Diem van: " << Van << endl;
00227.      cout << "Diem trung binh: ";
00228.      cout << DiemTrungBinh << endl;
00229.  }
```

01.04.04 Viết chương trình với kỹ thuật lập trình hàm và sử dụng kỹ thuật truyền tham số

- Khái niệm tham số hàm: Các thông số đầu vào của một hàm được gọi là tham số hàm.
- Phân loại tham số: có hai loại tham số.
  - + Tham trị.
  - + Tham biến.
- Sự thay đổi giá trị của tham số sau lời gọi hàm:
  - + Tham trị: không thay đổi.
  - + Tham biến: giá trị của tham số thay đổi sau lời gọi hàm.
- Cấp phát bộ nhớ
  - + Tham trị: cấp phát bộ nhớ.
  - + Tham biến: không cấp phát bộ nhớ.
- Cơ chế cấp phát bộ nhớ
  - + Tham trị: khi hàm được gọi thực hiện chương trình sẽ cấp phát bộ nhớ cho các tham số là tham trị sau đó sao chép giá trị của đối số tương ứng trong lời gọi hàm vào vùng nhớ tương ứng.
    - Ngôn ngữ lập trình C/C++ xem các tham số loại tham trị như là biến cục bộ của hàm.
    - Mọi sự thay đổi về mặt giá trị đối với tham số loại tham trị chỉ có ý nghĩa bên trong phạm vi của hàm và không làm thay đổi giá trị của đối số tương ứng.
    - Không gian bộ nhớ cấp phát cho các biến cục bộ của một hàm khi hàm được gọi thực hiện sẽ được tự động thu hồi bởi chương trình khi hàm thực hiện xong.
  - + Tham biến: khi hàm được gọi thực hiện chương trình không cấp phát bộ nhớ cho các tham số là tham biến mà sử dụng bộ nhớ của đối số tương ứng. Do đó mọi sự thay đổi bên trong hàm đối với tham số là tham biến đều dẫn

tới sự thay đổi về mặt giá trị của đối số tương ứng trong lời gọi hàm.

Bài cơ sở 004. Viết chương trình nhập họ tên, điểm toán, điểm văn của một học sinh. Tính điểm trung bình và xuất kết quả.

– Chương trình

```
00230. #include<iostream>
00231. #include<string>
00232. using namespace std;
00233.
00234. void Nhap(string&, int&, int&);
00235. void Xuat(string, int, int, float);
00236. void XuLy(int, int, float&);
00237.
00238. int main()
00239. {
00240.    string ht;
00241.    int t;
00242.    int v;
00243.    float tb;
00244.
00245.    Nhap(ht, t, v);
00246.    Xuly(t, v, tb);
00247.    Xuat(ht, t, v, tb);
00248.    return 1;
00249. }
00250.
00251. void Nhap(string& HoTen, int& Toan, int& Van)
00252. {
00253.    cout << "Nhap ho ten: ";
00254.    getline(cin, HoTen);
00255.    cout << "Nhap diem toan: ";
00256.    cin >> Toan;
00257.    cout << "Nhap diem van: ";
00258.    cin >> Van;
00259. }
00260.
00261. void Xuat(string HoTen, int Toan, int Van,
00262.           float DiemTrungBinh)
00263. {
00264.    cout << "Ho ten: " << HoTen << endl;
00265.    cout << "Diem toan: " << Toan << endl;
```

```
00266.     cout << "Diem van: " << Van << endl;
00267.     cout << "Diem trung binh: ";
00268.     cout << DiemTrungBinh << endl;
00269. }
00270.
00271. void XuLy(int Toan, int Van,
00272.           float& DiemTrungBinh)
00273. {
00274.     DiemTrungBinh = (float)(Toan + Van) / 2;
00275. }
```

01.04.05 Viết chương trình với kỹ thuật trừu tượng hóa dữ liệu

– Khái niệm: Kiểu dữ liệu cấu trúc là một phương pháp tích hợp các kiểu dữ liệu đơn thành kiểu dữ liệu phức nhằm mô tả biểu diễn thông tin của một khái niệm hay một đối tượng trong thế giới thực.

Bài cơ sở 005. Viết chương trình nhập họ tên, điểm toán, điểm văn của một học sinh. Tính điểm trung bình và xuất kết quả.

– Chương trình

```
00276. #include <iostream>
00277. #include <iomanip>
00278. #include <string>
00279. using namespace std;
00280.
00281. struct HocSinh
00282. {
00283.     string HoTen;
00284.     int Toan;
00285.     int Van;
00286.     float DiemTrungBinh;
00287. };
00288. typedef struct HocSinh HOCSINH;
00289.
00290. void Nhap(HOCSINH&);
00291. void Xuat(HOCSINH);
00292. void XuLy(HOCSINH&);
00293.
00294. int main()
00295. {
00296.     HOCSINH hs;
```

```
00297.     Nhap(hs);
00298.     XuLy(hs);
00299.     Xuat(hs);
00300.     return 0;
00301.  }
00302.
00303.  void Nhap(HOCSINH& x)
00304.  {
00305.     cout << "Nhap ho ten: ";
00306.     getline(cin, x.HoTen);
00307.     cout << "Nhap diem toan: ";
00308.     cin >> x.Toan;
00309.     cout << "Nhap diem van: ";
00310.     cin >> x.Van;
00311.  }
00312.
00313.  void Xuat(HOCSINH x)
00314.  {
00315.     cout << "Ho ten: " << x.HoTen << endl;
00316.     cout << "Diem toan: " << x.Toan << endl;
00317.     cout << "Diem van: " << x.Van << endl;
00318.     cout << "Diem trung binh: ";
00319.     cout << x.DiemTrungBinh << endl;
00320.  }
00321.
00322.  void XuLy(HOCSINH& x)
00323.  {
00324.     x.DiemTrungBinh = (float)(x.Toan+x.Van)/2;
00325.  }
```

01.05 CÁC BÀI TOÁN CƠ SỞ

01.05.01 Kiểu dữ liệu tích hợp đơn

Bài cơ sở 006. Hãy viết chương trình nhập và xuất thông tin của một nhân viên. Biết rằng một nhân viên gồm những thành phần như sau:
+ Mã nhân viên (MaSo): chuỗi ký tự.
+ Tên nhân viên (HoTen): chuỗi ký tự.
+ Lương nhân viên (Luong): kiểu số thực.

– Chương trình.

```
00326.  #include <iostream>
00327.  #include <iomanip>
00328.  #include <string>
00329.  using namespace std;
00330.
00331.  struct NhanVien
00332.  {
00333.     string MaSo;
00334.     string HoTen;
00335.     float Luong;
00336.  };
00337.  typedef struct NhanVien NHANVIEN;
00338.
00339.  void Nhap(NHANVIEN&);
00340.  void Xuat(NHANVIEN);
00341.
00342.  int main()
00343.  {
00344.     NHANVIEN nv;
00345.     Nhap(nv);
00346.     Xuat(nv);
00347.     return 0;
00348.  }
00349.
00350.  void Nhap(NHANVIEN& x)
00351.  {
00352.     cout << "Nhap ma so nhan vien: ";
00353.     getline(cin, x.MaSo);
00354.     cout << "Nhap ho ten nhan vien: ";
00355.     getline(cin, x.HoTen);
00356.     cout << "Nhap luong nhan vien: ";
00357.     cin >> x.Luong;
00358.  }
00359.
00360.  void Xuat(NHANVIEN x)
00361.  {
00362.     cout << "Ma nhan vien: " << x.MaSo <<endl;
00363.     cout << "Ho ten: " << x.HoTen << endl;
00364.     cout << setw(8);
00365.     cout << setprecision(5);
00366.     cout << "Luong: " << x.Luong << endl;
00367.  }
```

01.05.02 Kiểu dữ liệu tích hợp phức

Bài cơ sở 007. Hãy viết chương trình nhập và xuất thông tin của một vé xem phim (VE). Biết rằng một vé xem phim gồm những thành phần như sau:
+ Tên phim (TenPhim): chuỗi ký tự.
+ Giá tiền (GiaTien): kiểu số nguyên.
+ Xuất chiếu (XuatChieu): kiểu thời gian (THOIGIAN).
+ Ngày xem (NgayXem): kiểu dữ liệu ngày (NGAY).

– Chương trình.

```
00368.  #include <iostream>
00369.  #include <string>
00370.  #include <iomanip>
00371.  using namespace std;
00372.
00373.  struct Ngay
00374.  {
00375.      int Ngay;
00376.      int Thang;
00377.      int Nam;
00378.  };
00379.  typedef struct Ngay NGAY;
00380.
00381.  struct ThoiGian
00382.  {
00383.      int Gio;
00384.      int Phut;
00385.      int Giay;
00386.  };
00387.  typedef struct ThoiGian THOIGIAN;
00388.
00389.  struct Ve
00390.  {
00391.      string TenPhim;
00392.      int GiaTien;
00393.      THOIGIAN XuatChieu;
00394.      NGAY NgayXem;
00395.  };
00396.  typedef struct Ve VE;
00397.
00398.  void Nhap(NGAY&);
```

```
00399.  void Xuat(NGAY);
00400.
00401.  void Nhap(THOIGIAN&);
00402.  void Xuat(THOIGIAN);
00403.
00404.  void Nhap(VE&);
00405.  void Xuat(VE);
00406.
00407.  int main()
00408.  {
00409.     VE v;
00410.
00411.     Nhap(v);
00412.     Xuat(v);
00413.     return 1;
00414.  }
00415.
00416.  void Nhap(NGAY& x)
00417.  {
00418.     cout << "Nhap ngay:";
00419.     cin >> x.Ngay;
00420.     cout << "Nhap thang:";
00421.     cin >> x.Thang;
00422.     cout << "Nhap nam:";
00423.     cin >> x.Nam;
00424.  }
00425.
00426.  void Xuat(NGAY x)
00427.  {
00428.     cout << "Ngay: " << x.Ngay << endl;
00429.     cout << "Thang: " << x.Thang << endl;
00430.     cout << "Nam: " << x.Nam << endl;
00431.  }
00432.
00433.  void Nhap(THOIGIAN& x)
00434.  {
00435.     cout << "Nhap gio:";
00436.     cin >> x.Gio;
00437.     cout << "Nhap phut:";
00438.     cin >> x.Phut;
00439.     cout << "Nhap giay:";
00440.     cin >> x.Giay;
00441.  }
```

```
00442.
00443.  void Xuat(THOIGIAN x)
00444.  {
00445.     cout << "Gio: " << x.Gio << endl;
00446.     cout << "Phut: " << x.Phut << endl;
00447.     cout << "Giay: " << x.Giay << endl;
00448.  }
00449.
00450.  void Nhap(VE& x)
00451.  {
00452.     cout << "Nhap ten phim:";
00453.     getline(cin, x.TenPhim);
00454.     cout << "Nhap gia tien:";
00455.     cin >> x.GiaTien;
00456.     cout << "Nhap xuat chieu:" << endl;
00457.     Nhap(x.XuatChieu);
00458.     cout << "Nhap ngay xem:" << endl;
00459.     Nhap(x.NgayXem);
00460.  }
00461.
00462.  void Xuat(VE x)
00463.  {
00464.     cout << "Ve Xem Phim:" << endl;
00465.     cout << "Ten Phim:" << x.TenPhim << endl;
00466.     cout << "Gia tien:" << x.GiaTien << endl;
00467.     cout << "Xuat chieu:" << endl;
00468.     Xuat(x.XuatChieu);
00469.     cout << "Ngay xem:" << endl;
00470.     Xuat(x.NgayXem);
00471.  }
```

01.05.03 Mảng cấu trúc

Bài cơ sở 008. Hãy viết chương trình thực hiện các yêu cầu sau:
a. Nhập mảng một chiều các nhân viên. Biết rằng một nhân viên gồm những thành phần như sau:
+ Mã nhân viên (MaSo): chuỗi ký tự.
+ Tên nhân viên (HoTen): chuỗi ký tự.
+ Lương nhân viên (Luong): kiểu số thực.
b. Xuất mảng.
c. Tính tổng lương các nhân viên có trong mảng.

– Chương trình.

```
00472. #include <iostream>
00473. #include <iomanip>
00474. #include <string>
00475. using namespace std;
00476.
00477. struct NhanVien
00478. {
00479.    string MaSo;
00480.    string HoTen;
00481.    float Luong;
00482. };
00483. typedef struct NhanVien NHANVIEN;
00484.
00485. void Nhap(NHANVIEN&);
00486. void Xuat(NHANVIEN);
00487.
00488. void Nhap(NHANVIEN[], int&);
00489. void Xuat(NHANVIEN[], int);
00490.
00491. float TongLuong(NHANVIEN[], int);
00492.
00493. int main()
00494. {
00495.    int n;
00496.    NHANVIEN b[100];
00497.    Nhap(b, n);
00498.    Xuat(b, n);
00499.
00500.    float kq = TongLuong(b, n);
00501.    cout << "\nTong luong: " << kq;
00502.    return 0;
00503. }
00504.
00505. void Nhap(NHANVIEN& x)
00506. {
00507.    cout << "Nhap ma so nhan vien: ";
00508.    cin.ignore();
00509.    getline(cin, x.MaSo);
00510.    cout << "Nhap ho ten nhan vien: ";
00511.    getline(cin, x.HoTen);
00512.    cout << "Nhap luong nhan vien: ";
```

```
00513.    cin >> x.Luong;
00514. }
00515.
00516. void Xuat(NHANVIEN x)
00517. {
00518.    cout << "\nMa nhan vien: " << x.MaSo;
00519.    cout << "\nHo ten: " << x.HoTen;
00520.    cout << setprecision(5);
00521.    cout << "\nLuong: " << x.Luong << endl;
00522. }
00523.
00524. void Nhap(NHANVIEN a[], int& n)
00525. {
00526.    cout << "\nNhap so luong nhan vien: ";
00527.    cin >> n;
00528.    for (int i = 0; i < n; i++)
00529.    {
00530.       cout << "\nNhan vien thu "<<i+1<<":\n";
00531.       Nhap(a[i]);
00532.    }
00533. }
00534.
00535. void Xuat(NHANVIEN a[], int n)
00536. {
00537.    for (int i = 0; i < n; i++)
00538.    {
00539.       cout << "\nNhan vien thu "<<i+1<<":\n";
00540.       Xuat(a[i]);
00541.    }
00542. }
00543.
00544. float TongLuong(NHANVIEN a[], int n)
00545. {
00546.    float s = 0;
00547.    for (int i = 0; i < n; i++)
00548.       s = s + a[i].Luong;
00549.    return s;
00550. }
```

Bài cơ sở 009. Hãy viết chương trình thực hiện các yêu cầu sau:
a. Nhập mảng một chiều các vé xem phim. Biết rằng một vé xem phim gồm những thành phần như sau:

+ Tên phim (TenPhim): chuỗi ký tự.
+ Giá tiền (GiaTien): kiểu số nguyên.
+ Xuất chiếu (XuatChieu): kiểu thời gian (THOIGIAN).
+ Ngày xem (NgayXem): kiểu dữ liệu ngày (NGAY).

b. Xuất mảng vé xem phim.
c. Tính tổng tiền các vé có trong mảng.
d. Tìm vé xem phim theo tên phim.

– Chương trình.

```
00551.  #include <iostream>
00552.  #include <string>
00553.  #include <iomanip>
00554.  using namespace std;
00555.
00556.  struct Ngay
00557.  {
00558.     int Ngay;
00559.     int Thang;
00560.     int Nam;
00561.  };
00562.  typedef struct Ngay NGAY;
00563.
00564.  struct ThoiGian
00565.  {
00566.     int Gio;
00567.     int Phut;
00568.     int Giay;
00569.  };
00570.  typedef struct ThoiGian THOIGIAN;
00571.
00572.  struct Ve
00573.  {
00574.     string TenPhim;
00575.     int GiaTien;
00576.     THOIGIAN XuatChieu;
00577.     NGAY NgayXem;
00578.  };
00579.  typedef struct Ve VE;
00580.
00581.  void Nhap(NGAY&);
00582.  void Xuat(NGAY);
00583.
00584.  void Nhap(THOIGIAN&);
```

```
00585.  void Xuat(THOIGIAN);
00586.
00587.  void Nhap(VE&);
00588.  void Xuat(VE);
00589.
00590.  void Nhap(VE[], int&);
00591.  void Xuat(VE[], int);
00592.
00593.  int TongTien(VE[], int);
00594.  int TimVe(VE[], int, string);
00595.
00596.  int main()
00597.  {
00598.     int n;
00599.     VE b[100];
00600.     Nhap(b, n);
00601.     Xuat(b, n);
00602.
00603.     int kq = TongTien(b, n);
00604.     cout << "\nTong tien: " << kq;
00605.
00606.     string TenPhim;
00607.     cout << "\nNhap ten phim: ";
00608.     cin.ignore();
00609.     getline(cin, TenPhim);
00610.
00611.     kq = TimVe(b, n, TenPhim);
00612.     if (kq == -1)
00613.        cout << "\nKhong tim thay";
00614.     else
00615.     {
00616.        cout << "\nVe can tim: ";
00617.        Xuat(b[kq]);
00618.     }
00619.     return 1;
00620.  }
00621.
00622.  void Nhap(NGAY& x)
00623.  {
00624.     cout << "Nhap ngay:";
00625.     cin >> x.Ngay;
00626.     cout << "Nhap thang:";
00627.     cin >> x.Thang;
```

```
00628.    cout << "Nhap nam:";
00629.    cin >> x.Nam;
00630. }
00631.
00632. void Xuat(NGAY x)
00633. {
00634.    cout << "Ngay: " << x.Ngay << endl;
00635.    cout << "Thang: " << x.Thang << endl;
00636.    cout << "Nam: " << x.Nam << endl;
00637. }
00638.
00639. void Nhap(THOIGIAN& x)
00640. {
00641.    cout << "Nhap gio:";
00642.    cin >> x.Gio;
00643.    cout << "Nhap phut:";
00644.    cin >> x.Phut;
00645.    cout << "Nhap giay:";
00646.    cin >> x.Giay;
00647. }
00648.
00649. void Xuat(THOIGIAN x)
00650. {
00651.    cout << "Gio: " << x.Gio << endl;
00652.    cout << "Phut: " << x.Phut << endl;
00653.    cout << "Giay: " << x.Giay << endl;
00654. }
00655.
00656. void Nhap(VE& x)
00657. {
00658.    cout << "Nhap ten phim:";
00659.    cin.ignore();
00660.    getline(cin, x.TenPhim);
00661.    cout << "Nhap gia tien:";
00662.    cin >> x.GiaTien;
00663.    cout << "Nhap xuat chieu:" << endl;
00664.    Nhap(x.XuatChieu);
00665.    cout << "Nhap ngay xem:" << endl;
00666.    Nhap(x.NgayXem);
00667. }
00668.
00669. void Xuat(VE x)
00670. {
```

```
00671.     cout << "Ve Xem Phim:" << endl;
00672.     cout << "Ten Phim:" << x.TenPhim << endl;
00673.     cout << "Gia tien:" << x.GiaTien << endl;
00674.     cout << "Xuat chieu:" << endl;
00675.     Xuat(x.XuatChieu);
00676.     cout << "Ngay xem:" << endl;
00677.     Xuat(x.NgayXem);
00678. }
00679.
00680. void Nhap(VE a[], int& n)
00681. {
00682.     cout << "\nNhap so luong ve: ";
00683.     cin >> n;
00684.     for (int i = 0; i < n; i++)
00685.     {
00686.        cout << "\nVe thu " << i + 1 << ": \n";
00687.        Nhap(a[i]);
00688.     }
00689. }
00690.
00691. void Xuat(VE a[], int n)
00692. {
00693.     for (int i = 0; i < n; i++)
00694.     {
00695.        cout << "\nVe thu " << i + 1 << ": ";
00696.        Xuat(a[i]);
00697.     }
00698. }
00699.
00700. int TongTien(VE a[], int n)
00701. {
00702.     int s = 0;
00703.     for (int i = 0; i < n; i++)
00704.        s = s + a[i].GiaTien;
00705.     return s;
00706. }
00707.
00708. int TimVe(VE a[], int n, string x)
00709. {
00710.     for (int i = 0; i < n; i++)
00711.        if (a[i].TenPhim == x)
00712.           return i;
00713.     return -1;
```

```
00714.  }
```

01.06 BÀI TẬP TRỪU TƯỢNG HÓA DỮ LIỆU

01.06.01 Kiểu dữ liệu tích hợp đơn

Bài 001.Hãy viết chương trình nhập và xuất thông tin của một tỉnh (TINH). Biết rằng một tỉnh gồm những thành phần như sau:
+ Mã tỉnh (MaSo): kiểu số nguyên.
+ Tên tỉnh (TenTinh): chuỗi ký tự.
+ Dân số (DanSo): kiểu số nguyên.
+ Diện tích (DienTich): kiểu số thực.

– Chương trình.

```
00715.  #include <iostream>
00716.  #include <iomanip>
00717.  #include <string>
00718.  using namespace std;
00719.
00720.  struct Tinh
00721.  {
00722.     int MaSo;
00723.     string TenTinh;
00724.     int DanSo;
00725.     float DienTich;
00726.  };
00727.  typedef struct Tinh TINH;
00728.
00729.  void Nhap(TINH&);
00730.  void Xuat(TINH);
00731.
00732.  int main()
00733.  {
00734.     TINH t;
00735.     Nhap(t);
00736.     Xuat(t);
00737.     return 0;
00738.  }
00739.
00740.  void Nhap(TINH& x)
00741.  {
00742.     cout << "Nhap ma tinh: ";
00743.     cin >> x.MaSo;
```

```
00744.     cout << "Nhap ten tinh: ";
00745.     getline(cin, x.TenTinh);
00746.     cout << "Nhap dan so: ";
00747.     cin >> x.DanSo;
00748.     cout << "Nhap dien tich: ";
00749.     cin >> x.DienTich;
00750. }
00751.
00752. void Xuat(TINH x)
00753. {
00754.     cout << "Ma tinh: " << x.MaSo << endl;
00755.     cout << "Ten tinh: " << x.TenTinh << endl;
00756.     cout << "Dan so: " << x.DanSo << endl;
00757.     cout << "Dien tich: " << x.DienTich << endl;
00758. }
```

Bài 002.Hãy viết chương trình nhập và xuất thông tin của một mặt hàng (MATHANG). Biết rằng một mặt hàng gồm những thành phần như sau:
+ Tên mặt hàng (TenHang): chuỗi ký tự.
+ Đơn giá (DonGia): kiểu số nguyên.
+ Số lượng tồn (LuongTon): kiểu số nguyên.

– Chương trình.

```
00759. #include <iostream>
00760. #include <iomanip>
00761. #include <string>
00762. using namespace std;
00763.
00764. struct MatHang
00765. {
00766.     string TenHang;
00767.     int DonGia;
00768.     int LuongTon;
00769. };
00770. typedef struct MatHang MATHANG;
00771.
00772. void Nhap(MATHANG&);
00773. void Xuat(MATHANG);
00774.
00775. int main()
00776. {
00777.     MATHANG nv;
```

```
00778.     Nhap(nv);
00779.     Xuat(nv);
00780.     return 0;
00781.  }
00782.
00783.  void Nhap(MATHANG& x)
00784.  {
00785.     cout << "Nhap ten mat hang: ";
00786.     getline(cin, x.TenHang);
00787.     cout << "Nhap don gia: ";
00788.     cin >> x.DonGia;
00789.     cout << "Nhap so luong ton: ";
00790.     cin >> x.LuongTon;
00791.  }
00792.
00793.  void Xuat(MATHANG x)
00794.  {
00795.     cout << "\nTen mat hang: " << x.TenHang;
00796.     cout << "\nDon gia: " << x.DonGia;
00797.     cout << "\nSo luong ton: " << x.LuongTon;
00798.  }
```

Bài 003.Hãy viết chương trình nhập và xuất thông tin của một thí sinh (THISINH). Biết rằng một thí sinh gồm những thành phần như sau:
+ Mã thí sinh (MaSo): chuỗi ký tự.
+ Họ tên thí sinh (HoTen): chuỗi ký tự.
+ Điểm toán (Toan): kiểu số thực.
+ Điểm lý (Ly): kiểu số thực.
+ Điểm hóa (Hoa): kiểu số thực.
+Điểm tổng cộng (Tong): kiểu số thực.

– Chương trình.

```
00799.  #include <iostream>
00800.  #include <iomanip>
00801.  #include <string>
00802.  using namespace std;
00803.
00804.  struct HocSinh
00805.  {
00806.     string MaSo;
00807.     string HoTen;
00808.     float Toan;
00809.     float Ly;
```

```
00810.     float Hoa;
00811.     float Tong;
00812.  };
00813.  typedef struct HocSinh HOCSINH;
00814.
00815.  void Nhap(HOCSINH&);
00816.  void Xuat(HOCSINH);
00817.  void XuLy(HOCSINH&);
00818.
00819.  int main()
00820.  {
00821.     HOCSINH hs;
00822.     Nhap(hs);
00823.     XuLy(hs);
00824.     Xuat(hs);
00825.     return 0;
00826.  }
00827.
00828.  void Nhap(HOCSINH& x)
00829.  {
00830.     cout << "Nhap ma thi sinh: ";
00831.     getline(cin, x.MaSo);
00832.     cout << "Nhap ho ten: ";
00833.     getline(cin, x.HoTen);
00834.     cout << "Nhap diem toan: ";
00835.     cin >> x.Toan;
00836.     cout << "Nhap diem ly: ";
00837.     cin >> x.Ly;
00838.     cout << "Nhap diem hoa: ";
00839.     cin >> x.Hoa;
00840.  }
00841.
00842.  void Xuat(HOCSINH x)
00843.  {
00844.     cout << "\nMa thi sinh: " << x.MaSo << endl;
00845.     cout << "Ho ten: " << x.HoTen << endl;
00846.     cout << setprecision(3);
00847.     cout << "Diem toan: " << x.Toan << endl;
00848.     cout << "Diem ly: " << x.Ly << endl;
00849.     cout << "Diem hoa: " << x.Hoa << endl;
00850.     cout << "Diem tong: " << x.Tong << endl;
00851.  }
00852.
```

```
00853.  void XuLy(HOCSINH& x)
00854.  {
00855.     x.Tong = x.Toan + x.Ly + x.Hoa;
00856.  }
```

Bài 004.Hãy viết chương trình nhập và xuất thông tin của một luận văn (LUANVAN). Biết rằng một luận văn gồm những thành phần như sau:
+ Mã luận văn (MaSo): chuỗi ký tự.
+ Tên luận văn (TenLuanVan): chuỗi ký tự.
+ Họ tên sinh viên thực hiện (HoTenSV): chuỗi ký tự.
+ Họ tên giáo viên hướng dẫn (HoTenGV): chuỗi ký tự.
+ Năm thực hiện (Nam): kiểu số nguyên.

– Chương trình.

```
00857.  #include <iostream>
00858.  #include <iomanip>
00859.  #include <string>
00860.  using namespace std;
00861.
00862.  struct LuanVan
00863.  {
00864.     string MaSo;
00865.     string TenLuanVan;
00866.     string HoTenSV;
00867.     string HoTenGV;
00868.     int Nam;
00869.  };
00870.  typedef struct LuanVan LUANVAN;
00871.
00872.  void Nhap(LUANVAN&);
00873.  void Xuat(LUANVAN);
00874.
00875.  int main()
00876.  {
00877.     LUANVAN lv;
00878.     Nhap(lv);
00879.     Xuat(lv);
00880.     return 0;
00881.  }
00882.
00883.  void Nhap(LUANVAN& x)
00884.  {
```

```
00885.     cout << "Nhap ma luan van: ";
00886.     getline(cin, x.MaSo);
00887.     cout << "Nhap ten luan van: ";
00888.     getline(cin, x.TenLuanVan);
00889.     cout << "Nhap ten sinh vien: ";
00890.     getline(cin, x.HoTenSV);
00891.     cout << "Nhap ten giao vien: ";
00892.     getline(cin, x.HoTenGV);
00893.     cout << "Nhap nam thuc hien: ";
00894.     cin >> x.Nam;
00895. }
00896.
00897. void Xuat(LUANVAN x)
00898. {
00899.     cout << "\nMa luan van: " << x.MaSo;
00900.     cout << endl;
00901.     cout << "Ten luan van: " << x.TenLuanVan;
00902.     cout << endl;
00903.     cout << "Ten sinh vien: " << x.HoTenSV;
00904.     cout << endl;
00905.     cout << "Ten giao vien: " << x.HoTenGV;
00906.     cout << endl;
00907.     cout << "Nam thuc hien: " << x.Nam;
00908.     cout << endl;
00909. }
```

01.06.02 Kiểu dữ liệu tích hợp phức

Bài 005.Hãy viết chương trình nhập và xuất thông tin của một hộp sữa (HOPSUA). Biết rằng một hộp sữa gồm những thành phần như sau:
+ Nhãn hiệu (NhanHieu): chuỗi ký tự.
+ Trọng lượng (TrongLuong): kiểu số thực.
+ Hạn sử dụng (HanDung): kiểu dữ liệu ngày (NGAY).

– Chương trình.

```
00910. #include <iostream>
00911. #include <iomanip>
00912. #include <string>
00913. using namespace std;
00914.
00915. struct Ngay
00916. {
```

```
00917.     int Ngay;
00918.     int Thang;
00919.     int Nam;
00920.  };
00921.  typedef struct Ngay NGAY;
00922.
00923.  struct HopSua
00924.  {
00925.     string NhanHieu;
00926.     float TrongLuong;
00927.     NGAY HanDung;
00928.  };
00929.  typedef struct HopSua HOPSUA;
00930.
00931.  void Nhap(NGAY&);
00932.  void Xuat(NGAY);
00933.
00934.  void Nhap(HOPSUA&);
00935.  void Xuat(HOPSUA);
00936.
00937.  int main()
00938.  {
00939.     HOPSUA hs;
00940.     Nhap(hs);
00941.     Xuat(hs);
00942.     return 0;
00943.  }
00944.
00945.  void Nhap(NGAY& x)
00946.  {
00947.     cout << "\nNhap ngay:";
00948.     cin >> x.Ngay;
00949.     cout << "Nhap thang:";
00950.     cin >> x.Thang;
00951.     cout << "Nhap nam:";
00952.     cin >> x.Nam;
00953.  }
00954.
00955.  void Xuat(NGAY x)
00956.  {
00957.     cout << "\nNgay: " << x.Ngay << endl;
00958.     cout << "Thang: " << x.Thang << endl;
00959.     cout << "Nam: " << x.Nam << endl;
```

```
00960.  }
00961.
00962.  void Nhap(HOPSUA& x)
00963.  {
00964.     cout << "Nhap nhan hieu: ";
00965.     getline(cin, x.NhanHieu);
00966.     cout << "Nhap trong luong: ";
00967.     cin >> x.TrongLuong;
00968.     cout << "Nhap han su dung: ";
00969.     Nhap(x.HanDung);
00970.  }
00971.
00972.  void Xuat(HOPSUA x)
00973.  {
00974.     cout << "\nNhan hieu: " << x.NhanHieu << endl;
00975.     cout << "Trong luong: " << x.TrongLuong <<
endl;
00976.     cout << "Han su dung: ";
00977.     Xuat(x.HanDung);
00978.  }
```

Bài 006.Hãy viết chương trình nhập và xuất thông tin của một một cầu thủ (CAUTHU). Biết rằng một cầu thủ gồm những thành phần như sau:
+ Mã cầu thủ (MaSo): chuỗi ký tự.
+ Tên cầu thủ (HoTen): chuỗi ký tự.
+ Ngày sinh (NgaySinh): kiểu dữ liệu ngày.

– Chương trình.

```
00979.  #include <iostream>
00980.  #include <iomanip>
00981.  #include <string>
00982.  using namespace std;
00983.
00984.  struct Ngay
00985.  {
00986.     int Ngay;
00987.     int Thang;
00988.     int Nam;
00989.  };
00990.  typedef struct Ngay NGAY;
00991.
00992.  struct CauThu
```

```
00993.  {
00994.     string MaSo;
00995.     string HoTen;
00996.     NGAY NgaySinh;
00997.  };
00998.  typedef struct CauThu CAUTHU;
00999.
01000.  void Nhap(NGAY&);
01001.  void Xuat(NGAY);
01002.
01003.  void Nhap(CAUTHU&);
01004.  void Xuat(CAUTHU);
01005.
01006.  int main()
01007.  {
01008.     CAUTHU ct;
01009.     Nhap(ct);
01010.     Xuat(ct);
01011.     return 0;
01012.  }
01013.
01014.  void Nhap(NGAY& x)
01015.  {
01016.     cout << "\nNhap ngay:";
01017.     cin >> x.Ngay;
01018.     cout << "Nhap thang:";
01019.     cin >> x.Thang;
01020.     cout << "Nhap nam:";
01021.     cin >> x.Nam;
01022.  }
01023.
01024.  void Xuat(NGAY x)
01025.  {
01026.     cout << "\nNgay: " << x.Ngay << endl;
01027.     cout << "Thang: " << x.Thang << endl;
01028.     cout << "Nam: " << x.Nam << endl;
01029.  }
01030.
01031.  void Nhap(CAUTHU& x)
01032.  {
01033.     cout << "Nhap ma cau thu: ";
01034.     getline(cin, x.MaSo);
01035.     cout << "Nhap ten cau thu: ";
```

```
01036.     getline(cin, x.HoTen);
01037.     cout << "Nhap ngay sinh: ";
01038.     Nhap(x.NgaySinh);
01039.  }
01040.
01041.  void Xuat(CAUTHU x)
01042.  {
01043.     cout << "\nMa cau thu: " << x.MaSo << endl;
01044.     cout << "Ten cau thu: " << x.HoTen << endl;
01045.     cout << "Ngay sinh: ";
01046.     Xuat(x.NgaySinh);
01047.  }
```

Bài 007.Hãy viết chương trình nhập và xuất thông tin của một chuyến bay (CHUYENBAY). Biết rằng một chuyến bay gồm những thành phần như sau:
+ Mã chuyến bay (MaSo): chuỗi ký tự.
+ Ngày bay (NgayBay): kiểu dữ liệu ngày.
+ Giờ bay (GioBay): kiểu thời gian.
+ Nơi đi (NoiDi): chuỗi ký tự.
+ Nơi đến (NoiDen): chuỗi ký tự.

– Chương trình.

```
01048.  #include <iostream>
01049.  #include <iomanip>
01050.  #include <string>
01051.  using namespace std;
01052.
01053.  struct ThoiGian
01054.  {
01055.     int Gio;
01056.     int Phut;
01057.     int Giay;
01058.  };
01059.  typedef struct ThoiGian THOIGIAN;
01060.
01061.  struct Ngay
01062.  {
01063.     int Ngay;
01064.     int Thang;
01065.     int Nam;
01066.  };
01067.  typedef struct Ngay NGAY;
```

```
01068.
01069.  struct ChuyenBay
01070.  {
01071.     string MaSo;
01072.     NGAY NgayBay;
01073.     THOIGIAN GioBay;
01074.     string NoiDi;
01075.     string NoiDen;
01076.  };
01077.  typedef struct ChuyenBay CHUYENBAY;
01078.
01079.  void Nhap(NGAY&);
01080.  void Xuat(NGAY);
01081.
01082.  void Nhap(THOIGIAN&);
01083.  void Xuat(THOIGIAN);
01084.
01085.  void Nhap(CHUYENBAY&);
01086.  void Xuat(CHUYENBAY);
01087.
01088.  int main()
01089.  {
01090.     CHUYENBAY cb;
01091.     Nhap(cb);
01092.     Xuat(cb);
01093.     return 0;
01094.  }
01095.
01096.  void Nhap(NGAY& x)
01097.  {
01098.     cout << "\nNhap ngay:";
01099.     cin >> x.Ngay;
01100.     cout << "Nhap thang:";
01101.     cin >> x.Thang;
01102.     cout << "Nhap nam:";
01103.     cin >> x.Nam;
01104.  }
01105.
01106.  void Xuat(NGAY x)
01107.  {
01108.     cout << "\nNgay: " << x.Ngay << endl;
01109.     cout << "Thang: " << x.Thang << endl;
01110.     cout << "Nam: " << x.Nam << endl;
```

```
01111. }
01112.
01113. void Nhap(THOIGIAN& x)
01114. {
01115.    cout << "\nNhap gio:";
01116.    cin >> x.Gio;
01117.    cout << "Nhap phut:";
01118.    cin >> x.Phut;
01119.    cout << "Nhap giay:";
01120.    cin >> x.Giay;
01121. }
01122.
01123. void Xuat(THOIGIAN x)
01124. {
01125.    cout << "Gio: " << x.Gio << endl;
01126.    cout << "Phut: " << x.Phut << endl;
01127.    cout << "Giay: " << x.Giay << endl;
01128. }
01129.
01130. void Nhap(CHUYENBAY& x)
01131. {
01132.    cout << "Nhap ma chuyen bay: ";
01133.    getline(cin, x.MaSo);
01134.    cout << "Nhap ngay bay: ";
01135.    Nhap(x.NgayBay);
01136.    cout << "Nhap gio bay: ";
01137.    Nhap(x.GioBay);
01138.    cout << "Nhap noi di: ";
01139.    cin.ignore();
01140.    getline(cin, x.NoiDi);
01141.    cout << "Nhap noi den: ";
01142.    getline(cin, x.NoiDen);
01143. }
01144.
01145. void Xuat(CHUYENBAY x)
01146. {
01147.    cout << "\nMa chuyen bay: " << x.MaSo << endl;
01148.    cout << "Ngay bay: ";
01149.    Xuat(x.NgayBay);
01150.    cout << "\nGio bay: ";
01151.    Xuat(x.GioBay);
01152.    cout << "Noi di: " << x.NoiDi << endl;
01153.    cout << "Noi den: " << x.NoiDen << endl;
```

```
01154.  }
```

Bài 008.Hãy viết chương trình nhập và xuất thông tin của một sổ tiết kiệm (SOTIETKIEM). Biết rằng một sổ tiết kiệm gồm những thành phần như sau:
+ Mã sổ (MaSo): chuỗi ký tự.
+ Loại tiết kiệm (Loai): chuỗi ký tự.
+ Họ tên khách hàng (HoTen): chuỗi ký tự.
+ Chứng minh nhân dân (CMND): kiểu số nguyên.
+ Ngày mở số (NgayMo): kiểu dữ liệu ngày.
+ Số tiền gởi (TienGoi): kiểu số thực.

– Chương trình.

```
01155.  #include <iostream>
01156.  #include <iomanip>
01157.  #include <string>
01158.  using namespace std;
01159.
01160.  struct Ngay
01161.  {
01162.     int Ngay;
01163.     int Thang;
01164.     int Nam;
01165.  };
01166.  typedef struct Ngay NGAY;
01167.
01168.  struct SoTietKiem
01169.  {
01170.     string MaSo;
01171.     string Loai;
01172.     string HoTen;
01173.     int CMND;
01174.     NGAY NgayMo;
01175.     float TienGoi;
01176.  };
01177.  typedef struct SoTietKiem SOTIETKIEM;
01178.
01179.  void Nhap(NGAY&);
01180.  void Xuat(NGAY);
01181.
01182.  void Nhap(SOTIETKIEM&);
01183.  void Xuat(SOTIETKIEM);
01184.
```

```
01185. int main()
01186. {
01187.    SOTIETKIEM stk;
01188.    Nhap(stk);
01189.    Xuat(stk);
01190.    return 0;
01191. }
01192.
01193. void Nhap(NGAY& x)
01194. {
01195.    cout << "\nNhap ngay:";
01196.    cin >> x.Ngay;
01197.    cout << "Nhap thang:";
01198.    cin >> x.Thang;
01199.    cout << "Nhap nam:";
01200.    cin >> x.Nam;
01201. }
01202.
01203. void Xuat(NGAY x)
01204. {
01205.    cout << "\nNgay: " << x.Ngay << endl;
01206.    cout << "Thang: " << x.Thang << endl;
01207.    cout << "Nam: " << x.Nam << endl;
01208. }
01209.
01210. void Nhap(SOTIETKIEM& x)
01211. {
01212.    cout << "Nhap ma so tiet kiem: ";
01213.    getline(cin, x.MaSo);
01214.    cout << "Nhap loai tiet kiem: ";
01215.    getline(cin, x.Loai);
01216.    cout << "Nhap ho ten khach hang: ";
01217.    getline(cin, x.HoTen);
01218.    cout << "Nhap so chung minh nhan dan: ";
01219.    cin >> x.CMND;
01220.    cout << "Nhap ngay mo so: ";
01221.    Nhap(x.NgayMo);
01222.    cout << "Nhap so tien goi: ";
01223.    cin >> x.TienGoi;
01224. }
01225.
01226. void Xuat(SOTIETKIEM x)
01227. {
```

```
01228.     cout << "\nMa so tiet kiem: " << x.MaSo;
01229.     cout << "\nLoai tiet kiem: " << x.Loai;
01230.     cout << "\nHo ten KH: " << x.HoTen;
01231.     cout << "\CMND: " << x.CMND;
01232.     cout << "\nNgay mo so: ";
01233.     Xuat(x.NgayMo);
01234.     cout << "So tien goi: ";
01235.     cout << setprecision(5) << x.TienGoi;
01236. }
```

Bài 009.Hãy viết chương trình nhập và xuất thông tin của một đại lý (DAILY). Biết rằng một đại lý gồm những thành phần như sau:
+ Mã đại lý (MaSo): chuỗi ký tự.
+ Tên đại lý (TenDaiLy): chuỗi ký tự.
+ Điện thoại (DienThoai): kiểu số nguyên.
+ Ngày tiếp nhận (NgayNhan): kiểu dữ liệu ngày.
+ Địa chỉ (DiaChi): chuỗi ký tự.
+ E-Mail (EMail): chuỗi ký tự.

– Chương trình.

```
01237. #include <iostream>
01238. #include <iomanip>
01239. #include <string>
01240. using namespace std;
01241.
01242. struct Ngay
01243. {
01244.     int Ngay;
01245.     int Thang;
01246.     int Nam;
01247. };
01248. typedef struct Ngay NGAY;
01249.
01250. struct DaiLy
01251. {
01252.     string MaSo;
01253.     string TenDaiLy;
01254.     int DienThoai;
01255.     NGAY NgayNhan;
01256.     string DiaChi;
01257.     string EMail;
01258. };
01259. typedef struct DaiLy DAILY;
```

```
01260.
01261.  void Nhap(NGAY&);
01262.  void Xuat(NGAY);
01263.
01264.  void Nhap(DAILY&);
01265.  void Xuat(DAILY);
01266.
01267.  int main()
01268.  {
01269.     DAILY dl;
01270.     Nhap(dl);
01271.     Xuat(dl);
01272.     return 0;
01273.  }
01274.
01275.  void Nhap(NGAY& x)
01276.  {
01277.     cout << "\nNhap ngay:";
01278.     cin >> x.Ngay;
01279.     cout << "Nhap thang:";
01280.     cin >> x.Thang;
01281.     cout << "Nhap nam:";
01282.     cin >> x.Nam;
01283.  }
01284.
01285.  void Xuat(NGAY x)
01286.  {
01287.     cout << "\nNgay: " << x.Ngay << endl;
01288.     cout << "Thang: " << x.Thang << endl;
01289.     cout << "Nam: " << x.Nam << endl;
01290.  }
01291.
01292.  void Nhap(DAILY& x)
01293.  {
01294.     cout << "Nhap ma so dai ly: ";
01295.     getline(cin, x.MaSo);
01296.     cout << "Nhap ten dai ly: ";
01297.     getline(cin, x.TenDaiLy);
01298.     cout << "Nhap so dien thoai: ";
01299.     cin >> x.DienThoai;
01300.     cout << "Nhap ngay tiep nhan: ";
01301.     Nhap(x.NgayNhan);
01302.     cout << "Nhap dia chi: ";
```

```
01303.     cin.ignore();
01304.     getline(cin, x.DiaChi);
01305.     cout << "Nhap E-Mail: ";
01306.     getline(cin, x.EMail);
01307.  }
01308.
01309.  void Xuat(DAILY x)
01310.  {
01311.     cout << "\nMa so dai ly: " << x.MaSo;
01312.     cout << "\nTen dai ly: " << x.TenDaiLy;
01313.     cout << "\nDien thoai: " << x.DienThoai;
01314.     cout << "\nNgay tiep nhan: ";
01315.     Xuat(x.NgayNhan);
01316.     cout << "Dia chi: " << x.DiaChi;
01317.     cout << "\nE-mail: " << x.EMail;
01318.  }
```

01.06.03 Mảng cấu trúc

Bài 010.Viết chương trình nhập và xuất mảng một chiều các tỉnh (TINH). Biết rằng một tỉnh gồm những thành phần như sau:
+ Mã tỉnh (*MaSo*): kiểu số nguyên.
+ Tên tỉnh (*TenTinh*): chuỗi ký tự.
+ Dân số (*DanSo*): kiểu số nguyên.
+ Diện tích (*DienTich*): kiểu số thực.

a. Liệt kê các tỉnh có dân số lớn hơn 1.000.000.
b. Tìm một tỉnh có diện tích lớn nhất.
c. Sắp xếp các tỉnh giảm dần theo diện tích.

– Chương trình.

```
01319.  #include <iostream>
01320.  #include <iomanip>
01321.  #include <string>
01322.  using namespace std;
01323.
01324.  struct Tinh
01325.  {
01326.     int MaSo;
01327.     string TenTinh;
01328.     int DanSo;
01329.     float DienTich;
01330.  };
01331.  typedef struct Tinh TINH;
```

```
01332.
01333.  void Nhap(TINH&);
01334.  void Xuat(TINH);
01335.
01336.  void Nhap(TINH[], int&);
01337.  void Xuat(TINH[], int);
01338.
01339.  int main()
01340.  {
01341.     TINH b[100];
01342.     int n;
01343.     Nhap(b, n);
01344.     Xuat(b, n);
01345.     return 0;
01346.  }
01347.
01348.  void Nhap(TINH& x)
01349.  {
01350.     cout << "Nhap ma tinh: ";
01351.     cin >> x.MaSo;
01352.     cout << "Nhap ten tinh: ";
01353.     cin.ignore();
01354.     getline(cin, x.TenTinh);
01355.     cout << "Nhap dan so: ";
01356.     cin >> x.DanSo;
01357.     cout << "Nhap dien tich: ";
01358.     cin >> x.DienTich;
01359.  }
01360.
01361.  void Xuat(TINH x)
01362.  {
01363.     cout << "Ma tinh: " << x.MaSo << endl;
01364.     cout << "Ten tinh: " << x.TenTinh << endl;
01365.     cout << "Dan so: " << x.DanSo << endl;
01366.     cout << "Dien tich: " << x.DienTich << endl;
01367.  }
01368.
01369.  void Nhap(TINH a[], int& n)
01370.  {
01371.     cout << "\nNhap so luong tinh: ";
01372.     cin >> n;
01373.     for (int i = 0; i < n; i++)
01374.     {
```

```
01375.        cout << "\nTinh thu " << i + 1 << ":\n";
01376.        Nhap(a[i]);
01377.     }
01378.  }
01379.
01380.  void Xuat(TINH a[], int n)
01381.  {
01382.     for (int i = 0; i < n; i++)
01383.     {
01384.        cout << "\nTinh thu " << i + 1 << ":\n";
01385.        Xuat(a[i]);
01386.     }
01387.  }
```

Bài 011.Viết chương trình thực hiện những yêu cầu sau:

a. Nhập mảng một chiều các hộp sữa (HOPSUA). Biết rằng một hộp sữa gồm những thành phần như sau:

+ Nhãn hiệu (*NhanHieu*): chuỗi ký tự.
+ Trọng lượng (*TrongLuong*): kiểu số thực.
+ Hạn sử dụng (*HanDung*): kiểu dữ liệu ngày (NGAY).

b. Xuất mảng.

c. Nhập vào một ngày x. Hãy đếm số lượng hộp sữa trong mảng quá hạn sử dụng so với ngày x.

c. Tìm một hộp sữa có trọng lượng lớn nhất trong mảng.

– Chương trình.

```
01388.  #include <iostream>
01389.  #include <iomanip>
01390.  #include <string>
01391.  using namespace std;
01392.
01393.  struct Ngay
01394.  {
01395.     int Ngay;
01396.     int Thang;
01397.     int Nam;
01398.  };
01399.  typedef struct Ngay NGAY;
01400.
01401.  struct HopSua
01402.  {
01403.     string NhanHieu;
```

```
01404.     float TrongLuong;
01405.     NGAY HanDung;
01406.  };
01407.  typedef struct HopSua HOPSUA;
01408.
01409.  void Nhap(NGAY&);
01410.  void Xuat(NGAY);
01411.
01412.  void Nhap(HOPSUA&);
01413.  void Xuat(HOPSUA);
01414.
01415.  void Nhap(HOPSUA[], int&);
01416.  void Xuat(HOPSUA[], int);
01417.
01418.  int SoSanh(NGAY, NGAY);
01419.  int DemQuaHan(HOPSUA[], int, NGAY);
01420.  HOPSUA LonNhat(HOPSUA[], int);
01421.
01422.  int main()
01423.  {
01424.     HOPSUA b[100];
01425.     int n;
01426.     Nhap(b, n);
01427.     Xuat(b, n);
01428.
01429.     NGAY x;
01430.     cout << "\nNhap ngay x:";
01431.     Nhap(x);
01432.
01433.     int kq = DemQuaHan(b, n, x);
01434.     cout << "\nSo luong hop sua qua han su dung: "
<< kq;
01435.
01436.     HOPSUA ln;
01437.     ln = LonNhat(b, n);
01438.     Xuat(ln);
01439.     return 0;
01440.  }
01441.
01442.  void Nhap(NGAY& x)
01443.  {
01444.     cout << "\nNhap ngay:";
01445.     cin >> x.Ngay;
```

```
01446.    cout << "Nhap thang:";
01447.    cin >> x.Thang;
01448.    cout << "Nhap nam:";
01449.    cin >> x.Nam;
01450. }
01451.
01452. void Xuat(NGAY x)
01453. {
01454.    cout << "\nNgay: " << x.Ngay << endl;
01455.    cout << "Thang: " << x.Thang << endl;
01456.    cout << "Nam: " << x.Nam << endl;
01457. }
01458.
01459. void Nhap(HOPSUA& x)
01460. {
01461.    cout << "Nhap nhan hieu: ";
01462.    cin.ignore();
01463.    getline(cin, x.NhanHieu);
01464.    cout << "Nhap trong luong: ";
01465.    cin >> x.TrongLuong;
01466.    cout << "Nhap han su dung: ";
01467.    Nhap(x.HanDung);
01468. }
01469.
01470. void Xuat(HOPSUA x)
01471. {
01472.    cout << "\nNhan hieu: " << x.NhanHieu << endl;
01473.    cout << "Trong luong: " << x.TrongLuong <<
endl;
01474.    cout << "Han su dung: ";
01475.    Xuat(x.HanDung);
01476. }
01477.
01478. void Nhap(HOPSUA a[], int& n)
01479. {
01480.    cout << "\nNhap so luong hop sua: ";
01481.    cin >> n;
01482.    for (int i = 0; i < n; i++)
01483.    {
01484.       cout << "\nHop sua thu " << i + 1 << ":\n";
01485.       Nhap(a[i]);
01486.    }
01487. }
```

```
01488.
01489.  void Xuat(HOPSUA a[], int n)
01490.  {
01491.     for (int i = 0; i < n; i++)
01492.     {
01493.        cout << "\nHop sua thu " << i + 1 << ":\n";
01494.        Xuat(a[i]);
01495.     }
01496.  }
01497.
01498.  int SoSanh(NGAY x, NGAY y)
01499.  {
01500.     if (x.Nam > y.Nam)
01501.        return 1;
01502.     if (x.Nam < y.Nam)
01503.        return -1;
01504.     if (x.Thang > y.Thang)
01505.        return 1;
01506.     if (x.Thang < y.Thang)
01507.        return -1;
01508.     if (x.Ngay > y.Ngay)
01509.        return 1;
01510.     if (x.Ngay < y.Ngay)
01511.        return -1;
01512.     return 0;
01513.  }
01514.
01515.  int DemQuaHan(HOPSUA a[], int n, NGAY x)
01516.  {
01517.     int dem = 0;
01518.     for (int i = 0; i < n; i++)
01519.     {
01520.        if (SoSanh(a[i].HanDung, x) == -1)
01521.           dem = dem + 1;
01522.     }
01523.     return dem;
01524.  }
01525.
01526.  HOPSUA LonNhat(HOPSUA a[], int n)
01527.  {
01528.     HOPSUA lc = a[0];
01529.     for (int i = 0; i < n; i++)
01530.     {
```

```
01531.        if (a[i].TrongLuong > lc.TrongLuong)
01532.           lc = a[i];
01533.     }
01534.     return lc;
01535.  }
```

Bài 012.Viết chương trình thực hiện những yêu cầu sau:

a. Nhập mảng một chiều các vé xem phim (VE). Biết rằng một vé xem phim gồm những thành phần như sau:

- Tên phim (*TenPhim*): chuỗi ký tự.
- Giá tiền (*GiaTien*): kiểu số nguyên.
- Xuất chiếu (*XuatChieu*): kiểu thời gian (THOIGIAN).
- Ngày xem (*NgayXem*): kiểu dữ liệu ngày (NGAY).

b. Xuất mảng.

c. Tính tổng giá tiền của tất cả các vé trong mảng.

d. Sắp xếp các phần tử trong mảng tăng dần theo ngày xem và xuất chiếu.

– Chương trình.

```
01536.  #include <iostream>
01537.  #include <string>
01538.  #include <iomanip>
01539.  using namespace std;
01540.
01541.  struct Ngay
01542.  {
01543.     int Ngay;
01544.     int Thang;
01545.     int Nam;
01546.  };
01547.  typedef struct Ngay NGAY;
01548.
01549.  struct ThoiGian
01550.  {
01551.     int Gio;
01552.     int Phut;
01553.     int Giay;
01554.  };
01555.  typedef struct ThoiGian THOIGIAN;
01556.
01557.  struct Ve
01558.  {
01559.     string TenPhim;
```

```
01560.     int GiaTien;
01561.     THOIGIAN XuatChieu;
01562.     NGAY NgayXem;
01563. };
01564. typedef struct Ve VE;
01565.
01566. void Nhap(NGAY&);
01567. void Xuat(NGAY);
01568.
01569. void Nhap(THOIGIAN&);
01570. void Xuat(THOIGIAN);
01571.
01572. void Nhap(VE&);
01573. void Xuat(VE);
01574.
01575. void Nhap(VE[], int&);
01576. void Xuat(VE[], int);
01577.
01578. int TongTien(VE[], int);
01579. int SoSanh(NGAY, NGAY);
01580. int SoSanh(THOIGIAN, THOIGIAN);
01581. int SoSanh(VE, VE);
01582. void HoanVi(VE&, VE&);
01583. void SapXep(VE[], int);
01584.
01585. int main()
01586. {
01587.     int n;
01588.     VE b[100];
01589.     Nhap(b, n);
01590.     Xuat(b, n);
01591.
01592.     int kq = TongTien(b, n);
01593.     cout << "\nTong tien: " << kq;
01594.
01595.     SapXep(b, n);
01596.     cout << "\nDanh sach ve sau khi sap xep: ";
01597.     Xuat(b, n);
01598.     return 1;
01599. }
01600.
01601. void Nhap(NGAY& x)
01602. {
```

```
01603.     cout << "Nhap ngay:";
01604.     cin >> x.Ngay;
01605.     cout << "Nhap thang:";
01606.     cin >> x.Thang;
01607.     cout << "Nhap nam:";
01608.     cin >> x.Nam;
01609. }
01610.
01611. void Xuat(NGAY x)
01612. {
01613.     cout << "Ngay: " << x.Ngay << endl;
01614.     cout << "Thang: " << x.Thang << endl;
01615.     cout << "Nam: " << x.Nam << endl;
01616. }
01617.
01618. void Nhap(THOIGIAN& x)
01619. {
01620.     cout << "Nhap gio:";
01621.     cin >> x.Gio;
01622.     cout << "Nhap phut:";
01623.     cin >> x.Phut;
01624.     cout << "Nhap giay:";
01625.     cin >> x.Giay;
01626. }
01627.
01628. void Xuat(THOIGIAN x)
01629. {
01630.     cout << "Gio: " << x.Gio << endl;
01631.     cout << "Phut: " << x.Phut << endl;
01632.     cout << "Giay: " << x.Giay << endl;
01633. }
01634.
01635. void Nhap(VE& x)
01636. {
01637.     cout << "Nhap ten phim:";
01638.     cin.ignore();
01639.     getline(cin, x.TenPhim);
01640.     cout << "Nhap gia tien:";
01641.     cin >> x.GiaTien;
01642.     cout << "Nhap xuat chieu:" << endl;
01643.     Nhap(x.XuatChieu);
01644.     cout << "Nhap ngay xem:" << endl;
01645.     Nhap(x.NgayXem);
```

```
01646. }
01647.
01648. void Xuat(VE x)
01649. {
01650.    cout << "Ve Xem Phim:" << endl;
01651.    cout << "Ten Phim:" << x.TenPhim << endl;
01652.    cout << "Gia tien:" << x.GiaTien << endl;
01653.    cout << "Xuat chieu:" << endl;
01654.    Xuat(x.XuatChieu);
01655.    cout << "Ngay xem:" << endl;
01656.    Xuat(x.NgayXem);
01657. }
01658.
01659. void Nhap(VE a[], int& n)
01660. {
01661.    cout << "\nNhap so luong ve: ";
01662.    cin >> n;
01663.    for (int i = 0; i < n; i++)
01664.    {
01665.       cout << "\nVe thu " << i + 1 << ": \n";
01666.       Nhap(a[i]);
01667.    }
01668. }
01669.
01670. void Xuat(VE a[], int n)
01671. {
01672.    for (int i = 0; i < n; i++)
01673.    {
01674.       cout << "\nVe thu " << i + 1 << ": ";
01675.       Xuat(a[i]);
01676.    }
01677. }
01678.
01679. int TongTien(VE a[], int n)
01680. {
01681.    int s = 0;
01682.    for (int i = 0; i < n; i++)
01683.       s = s + a[i].GiaTien;
01684.    return s;
01685. }
01686.
01687. int SoSanh(NGAY x, NGAY y)
01688. {
```

```
01689.     if (x.Nam > y.Nam)
01690.        return 1;
01691.     if (x.Nam < y.Nam)
01692.        return -1;
01693.     if (x.Thang > y.Thang)
01694.        return 1;
01695.     if (x.Thang < y.Thang)
01696.        return -1;
01697.     if (x.Ngay > y.Ngay)
01698.        return 1;
01699.     if (x.Ngay < y.Ngay)
01700.        return -1;
01701.     return 0;
01702.  }
01703.
01704.  int SoSanh(THOIGIAN x, THOIGIAN y)
01705.  {
01706.     if (x.Gio > y.Gio)
01707.        return 1;
01708.     if (x.Gio < y.Gio)
01709.        return -1;
01710.     if (x.Phut > y.Phut)
01711.        return 1;
01712.     if (x.Phut < y.Phut)
01713.        return -1;
01714.     if (x.Giay > y.Giay)
01715.        return 1;
01716.     if (x.Giay < y.Giay)
01717.        return -1;
01718.     return 0;
01719.  }
01720.
01721.  int SoSanh(VE x, VE y)
01722.  {
01723.     if (SoSanh(x.NgayXem, y.NgayXem) == 1)
01724.        return 1;
01725.     if (SoSanh(x.NgayXem, x.NgayXem) == 0)
01726.        if (SoSanh(x.XuatChieu, y.XuatChieu) == 1)
01727.           return 1;
01728.     return 0;
01729.  }
01730.
01731.  void HoanVi(VE& a, VE& b)
```

```
{
   VE temp;
   temp = a;
   a = b;
   b = temp;
}

void SapXep(VE a[], int n)
{
   for (int i = 0; i < n - 1; i++)
      for (int j = i + 1; j < n; j++)

         HoanVi(a[i], a[j]);
}
```

Bài 013.Viết chương trình thực hiện những yêu cầu sau:

a. Nhập mảng một chiều các mặt hàng (MATHANG). Biết rằng một mặt hàng gồm những thành phần như sau:

+ Tên mặt hàng (*TenHang*): chuỗi ký tự.
+ Đơn giá (*DonGia*): kiểu số nguyên.
+ Số lượng tồn (*LuongTon*): kiểu số nguyên.

b. Xuất mảng.

c. Tìm mặt hàng có tổng giá trị tồn là lớn nhất.

d. Đếm số lượng mặt hàng có số lượng tồn lớn hơn 1.000.

– Chương trình.

```
#include <iostream>
#include <iomanip>
#include <string>
using namespace std;

struct MatHang
{
   string TenHang;
   int DonGia;
   int LuongTon;
};
typedef struct MatHang MATHANG;

void Nhap(MATHANG&);
void Xuat(MATHANG);

void Nhap(MATHANG[], int&);
```

```
01763.  void Xuat(MATHANG[], int);
01764.
01765.  int LonNhat(MATHANG[], int);
01766.  int SoSanh(MATHANG, MATHANG);
01767.  int DemHangTon(MATHANG[], int);
01768.
01769.  int main()
01770.  {
01771.     MATHANG b[100];
01772.     int n;
01773.     Nhap(b, n);
01774.     Xuat(b, n);
01775.
01776.     int ln = LonNhat(b, n);
01777.     if (ln == -1)
01778.        cout << "\nKhong tim thay";
01779.     else
01780.     {
01781.        cout << "\nMat hang co tong ton lon nhat:";
01782.        Xuat(b[ln]);
01783.     }
01784.
01785.     int kq = DemHangTon(b, n);
01786.     cout << "\nSo mat hang co luong ton > 1000: ";
01787.     cout << kq;
01788.     return 0;
01789.  }
01790.
01791.  void Nhap(MATHANG& x)
01792.  {
01793.     cout << "Nhap ten mat hang: ";
01794.     cin.ignore();
01795.     getline(cin, x.TenHang);
01796.     cout << "Nhap don gia: ";
01797.     cin >> x.DonGia;
01798.     cout << "Nhap so luong ton: ";
01799.     cin >> x.LuongTon;
01800.  }
01801.
01802.  void Xuat(MATHANG x)
01803.  {
01804.     cout << "\nTen mat hang: " << x.TenHang;
01805.     cout << "\nDon gia: " << x.DonGia;
```

```
01806.    cout << "\nSo luong ton: " << x.LuongTon;
01807. }
01808.
01809. void Nhap(MATHANG a[], int& n)
01810. {
01811.    cout << "\nNhap so luong mat hang: ";
01812.    cin >> n;
01813.    for (int i = 0; i < n; i++)
01814.    {
01815.       cout << "\nMat hang thu " << i + 1 <<
":\n";
01816.       Nhap(a[i]);
01817.    }
01818. }
01819.
01820. void Xuat(MATHANG a[], int n)
01821. {
01822.    for (int i = 0; i < n; i++)
01823.    {
01824.       cout << "\nMat hang thu " << i + 1 << ":";
01825.       Xuat(a[i]);
01826.    }
01827. }
01828.
01829. int SoSanh(MATHANG x, MATHANG y)
01830. {
01831.    int Tongx = x.DonGia * x.LuongTon;
01832.    int Tongy = y.DonGia * y.LuongTon;
01833.    if (Tongx > Tongy)
01834.       return 1;
01835.    return 0;
01836. }
01837.
01838. int LonNhat(MATHANG a[], int n)
01839. {
01840.    if (n == 0)
01841.       return -1;
01842.    int lc = 0;
01843.    for (int i = 0; i < n; i++)
01844.       if (SoSanh(a[i], a[lc]) == 1)
01845.          lc = i;
01846.    return lc;
01847. }
```

```
01848.
01849.  int DemHangTon(MATHANG a[], int n)
01850.  {
01851.     int dem = 0;
01852.     for (int i = 0; i < n; i++)
01853.        if (a[i].LuongTon > 1000)
01854.           dem = dem + 1;
01855.     return dem;
01856.  }
```

Bài 014.Viết chương trình thực hiện những yêu cầu sau:

a. Nhập mảng một chiều các chuyến bay. Biết rằng một chuyến bay gồm những thành phần như sau:

+ Mã chuyến bay (*MaSo*): chuỗi ký tự.
+ Ngày bay (*NgayBay*): kiểu dữ liệu ngày.
+ Giờ bay (*GioBay*): kiểu thời gian.
+ Nơi đi (*NoiDi*): chuỗi ký tự.
+ Nơi đến (*NoiDen*): chuỗi ký tự.

b. Xuất mảng.

c. Tìm một ngày có nhiều chuyến bay nhất.

d. Tìm một chuyến bay trong mảng theo mã chuyến bay.

– Chương trình.

```
01857.  #include <iostream>
01858.  #include <iomanip>
01859.  #include <string>
01860.  using namespace std;
01861.
01862.  struct ThoiGian
01863.  {
01864.     int Gio;
01865.     int Phut;
01866.     int Giay;
01867.  };
01868.  typedef struct ThoiGian THOIGIAN;
01869.
01870.  struct Ngay
01871.  {
01872.     int Ngay;
01873.     int Thang;
01874.     int Nam;
01875.  };
01876.  typedef struct Ngay NGAY;
```

```
01877.
01878.  struct ChuyenBay
01879.  {
01880.     string MaSo;
01881.     NGAY NgayBay;
01882.     THOIGIAN GioBay;
01883.     string NoiDi;
01884.     string NoiDen;
01885.  };
01886.  typedef struct ChuyenBay CHUYENBAY;
01887.
01888.  void Nhap(NGAY&);
01889.  void Xuat(NGAY);
01890.
01891.  void Nhap(THOIGIAN&);
01892.  void Xuat(THOIGIAN);
01893.
01894.  void Nhap(CHUYENBAY&);
01895.  void Xuat(CHUYENBAY);
01896.
01897.  void Nhap(CHUYENBAY[], int&);
01898.  void Xuat(CHUYENBAY[], int);
01899.
01900.  int SoSanh(NGAY, NGAY);
01901.  int TanSuat(CHUYENBAY[], int, NGAY);
01902.  NGAY TimNgay(CHUYENBAY[], int);
01903.  int TimChuyenBay(CHUYENBAY[], int, string);
01904.
01905.  int main()
01906.  {
01907.     CHUYENBAY b[100];
01908.     int n;
01909.
01910.     Nhap(b, n);
01911.     Xuat(b, n);
01912.
01913.     NGAY Ng = TimNgay(b, n);
01914.     cout << "\nNgay co nhieu chuyen bay nhat: ";
01915.     Xuat(Ng);
01916.
01917.     string MaChuyenBay;
01918.     cout << "\nNhap ma chuyen bay can tim: ";
01919.     getline(cin, MaChuyenBay);
```

```
01920.
01921.    int kq = TimChuyenBay(b, n, MaChuyenBay);
01922.    if (kq == -1)
01923.       cout << "\nKhong tim thay";
01924.    else
01925.    {
01926.       cout << "\nChuyen bay can tim: ";
01927.       Xuat(b[kq]);
01928.    }
01929.    return 0;
01930. }
01931.
01932. void Nhap(NGAY& x)
01933. {
01934.    cout << "\nNhap ngay:";
01935.    cin >> x.Ngay;
01936.    cout << "Nhap thang:";
01937.    cin >> x.Thang;
01938.    cout << "Nhap nam:";
01939.    cin >> x.Nam;
01940. }
01941.
01942. void Xuat(NGAY x)
01943. {
01944.    cout << "\nNgay: " << x.Ngay << endl;
01945.    cout << "Thang: " << x.Thang << endl;
01946.    cout << "Nam: " << x.Nam << endl;
01947. }
01948.
01949. void Nhap(THOIGIAN& x)
01950. {
01951.    cout << "\nNhap gio:";
01952.    cin >> x.Gio;
01953.    cout << "Nhap phut:";
01954.    cin >> x.Phut;
01955.    cout << "Nhap giay:";
01956.    cin >> x.Giay;
01957. }
01958.
01959. void Xuat(THOIGIAN x)
01960. {
01961.    cout << "\nGio: " << x.Gio << endl;
01962.    cout << "Phut: " << x.Phut << endl;
```

```
01963.    cout << "Giay: " << x.Giay << endl;
01964. }
01965.
01966. void Nhap(CHUYENBAY& x)
01967. {
01968.    cout << "Nhap ma chuyen bay: ";
01969.    cin.ignore();
01970.    getline(cin, x.MaSo);
01971.    cout << "Nhap ngay bay: ";
01972.    Nhap(x.NgayBay);
01973.    cout << "Nhap gio bay: ";
01974.    Nhap(x.GioBay);
01975.    cout << "Nhap noi di: ";
01976.    cin.ignore();
01977.    getline(cin, x.NoiDi);
01978.    cout << "Nhap noi den: ";
01979.    getline(cin, x.NoiDen);
01980. }
01981.
01982. void Xuat(CHUYENBAY x)
01983. {
01984.    cout << "\nMa chuyen bay: " << x.MaSo << endl;
01985.    cout << "Ngay bay: ";
01986.    Xuat(x.NgayBay);
01987.    cout << "Gio bay: ";
01988.    Xuat(x.GioBay);
01989.    cout << "Noi di: " << x.NoiDi << endl;
01990.    cout << "Noi den: " << x.NoiDen << endl;
01991. }
01992.
01993. void Nhap(CHUYENBAY a[], int& n)
01994. {
01995.    cout << "\nNhap so luong chuyen bay: ";
01996.    cin >> n;
01997.    for (int i = 0; i < n; i++)
01998.    {
01999.       cout << "\nChuyen bay thu " << i + 1 <<
":\n";
02000.       Nhap(a[i]);
02001.    }
02002. }
02003.
02004. void Xuat(CHUYENBAY a[], int n)
```

```
02005. {
02006.    for (int i = 0; i < n; i++)
02007.    {
02008.       cout << "\nChuyen bay thu " << i + 1 <<
":";
02009.       Xuat(a[i]);
02010.    }
02011. }
02012.
02013. int SoSanh(NGAY x, NGAY y)
02014. {
02015.    if (x.Nam > y.Nam)
02016.       return 1;
02017.    if (x.Nam < y.Nam)
02018.       return -1;
02019.    if (x.Thang > y.Thang)
02020.       return 1;
02021.    if (x.Thang < y.Thang)
02022.       return -1;
02023.    if (x.Ngay > y.Ngay)
02024.       return 1;
02025.    if (x.Ngay < y.Ngay)
02026.       return -1;
02027.    return 0;
02028. }
02029.
02030. int TanSuat(CHUYENBAY a[], int n, NGAY x)
02031. {
02032.    int dem = 0;
02033.    for (int i = 0; i < n; i++)
02034.       if (SoSanh(a[i].NgayBay, x) == 0)
02035.          dem = dem + 1;
02036.    return dem;
02037. }
02038.
02039. NGAY TimNgay(CHUYENBAY a[], int n)
02040. {
02041.    NGAY lc = a[0].NgayBay;
02042.    for (int i = 0; i < n; i++)
02043.    {
02044.       int x = TanSuat(a, n, a[i].NgayBay);
02045.       int y = TanSuat(a, n, lc);
02046.       if (x > y)
```

```
02047.          lc = a[i].NgayBay;
02048.     }
02049.     return lc;
02050. }
02051.
02052. int TimChuyenBay(CHUYENBAY a[], int n, string x)
02053. {
02054.     for (int i = 0; i < n; i++)
02055.        if (a[i].MaSo == x)
02056.           return i;
02057.     return -1;
02058. }
```

Bài 015. Viết chương trình thực hiện những yêu cầu sau:

a. Nhập mảng một chiều các cầu thủ. Biết rằng một cầu thủ gồm những thành phần như sau:
- Mã cầu thủ (*MaSo*): chuỗi ký tự.
- Tên cầu thủ (*HoTen*): chuỗi ký tự.
- Ngày sinh (*NgaySinh*): kiểu dữ liệu ngày.

b. Xuất mảng.

c. Liệt kê danh sách các cầu thủ nhỏ tuổi nhất trong mảng.

d. Sắp xếp các cầu thủ giảm dần theo ngày sinh.

– Chương trình.

```
02059. #include <iostream>
02060. #include <iomanip>
02061. #include <string>
02062. using namespace std;
02063.
02064. struct Ngay
02065. {
02066.     int Ngay;
02067.     int Thang;
02068.     int Nam;
02069. };
02070. typedef struct Ngay NGAY;
02071.
02072. struct CauThu
02073. {
02074.     string MaSo;
02075.     string HoTen;
02076.     NGAY NgaySinh;
02077. };
```

```
02078.  typedef struct CauThu CAUTHU;
02079.
02080.  void Nhap(NGAY&);
02081.  void Xuat(NGAY);
02082.
02083.  void Nhap(CAUTHU&);
02084.  void Xuat(CAUTHU);
02085.
02086.  void Nhap(CAUTHU[], int&);
02087.  void Xuat(CAUTHU[], int);
02088.
02089.  int NamLonNhat(CAUTHU[], int);
02090.  void LietKe(CAUTHU[], int);
02091.
02092.  int SoSanh(NGAY, NGAY);
02093.  void HoanVi(CAUTHU&, CAUTHU&);
02094.  void SapXep(CAUTHU[], int);
02095.
02096.  int main()
02097.  {
02098.     CAUTHU b[100];
02099.     int n;
02100.     Nhap(b, n);
02101.     Xuat(b, n);
02102.
02103.     cout << "\nCac cau thu nho tuoi nhat: ";
02104.     LietKe(b, n);
02105.
02106.     SapXep(b, n);
02107.     cout << "\nDanh sach sau khi sap xep: ";
02108.     Xuat(b, n);
02109.     return 0;
02110.  }
02111.
02112.  void Nhap(NGAY& x)
02113.  {
02114.     cout << "\nNhap ngay:";
02115.     cin >> x.Ngay;
02116.     cout << "Nhap thang:";
02117.     cin >> x.Thang;
02118.     cout << "Nhap nam:";
02119.     cin >> x.Nam;
02120.  }
```

```
02121.
02122.  void Xuat(NGAY x)
02123.  {
02124.     cout << "\nNgay: " << x.Ngay << endl;
02125.     cout << "Thang: " << x.Thang << endl;
02126.     cout << "Nam: " << x.Nam << endl;
02127.  }
02128.
02129.  void Nhap(CAUTHU& x)
02130.  {
02131.     cout << "Nhap ma cau thu: ";
02132.     cin.ignore();
02133.     getline(cin, x.MaSo);
02134.     cout << "Nhap ten cau thu: ";
02135.     getline(cin, x.HoTen);
02136.     cout << "Nhap ngay sinh: ";
02137.     Nhap(x.NgaySinh);
02138.  }
02139.
02140.  void Xuat(CAUTHU x)
02141.  {
02142.     cout << "\nMa cau thu: " << x.MaSo << endl;
02143.     cout << "Ten cau thu: " << x.HoTen << endl;
02144.     cout << "Ngay sinh: ";
02145.     Xuat(x.NgaySinh);
02146.  }
02147.
02148.  void Nhap(CAUTHU a[], int& n)
02149.  {
02150.     cout << "\nNhap so luong cau thu: ";
02151.     cin >> n;
02152.     for (int i = 0; i < n; i++)
02153.     {
02154.        cout << "\nCau thu thu " << i + 1 << ":\n";
02155.        Nhap(a[i]);
02156.     }
02157.  }
02158.
02159.  void Xuat(CAUTHU a[], int n)
02160.  {
02161.     for (int i = 0; i < n; i++)
02162.     {
02163.        cout << "\nCau thu thu " << i + 1 << ":";
```

```
02164.       Xuat(a[i]);
02165.    }
02166. }
02167.
02168. int NamLonNhat(CAUTHU a[], int n)
02169. {
02170.    int lc = a[0].NgaySinh.Nam;
02171.    for (int i = 0; i < n; i++)
02172.       if (a[i].NgaySinh.Nam > lc)
02173.          lc = a[i].NgaySinh.Nam;
02174.    return lc;
02175. }
02176.
02177. void LietKe(CAUTHU a[], int n)
02178. {
02179.    int ln = NamLonNhat(a, n);
02180.    for (int i = 0; i < n; i++)
02181.       if (a[i].NgaySinh.Nam == ln)
02182.          Xuat(a[i]);
02183. }
02184.
02185. void HoanVi(CAUTHU& a, CAUTHU& b)
02186. {
02187.    CAUTHU temp;
02188.    temp = a;
02189.    a = b;
02190.    b = temp;
02191. }
02192.
02193. int SoSanh(NGAY x, NGAY y)
02194. {
02195.    if (x.Nam > y.Nam)
02196.       return 1;
02197.    if (x.Nam < y.Nam)
02198.       return -1;
02199.    if (x.Thang > y.Thang)
02200.       return 1;
02201.    if (x.Thang < y.Thang)
02202.       return -1;
02203.    if (x.Ngay > y.Ngay)
02204.       return 1;
02205.    if (x.Ngay < y.Ngay)
02206.       return -1;
```

```
02207.    return 0;
02208. }
02209.
02210. void SapXep(CAUTHU a[], int n)
02211. {
02212.    for (int i = 0; i < n - 1; i++)
02213.       for (int j = i + 1; j < n; j++)
02214.       {
02215.          NGAY x = a[i].NgaySinh;
02216.          NGAY y = a[j].NgaySinh;
02217.          if (SoSanh(x, y) == -1)
02218.             HoanVi(a[i], a[j]);
02219.       }
02220. }
```

Bài 016.Viết chương trình thực hiện những yêu cầu sau:

a. Nhập mảng một chiều các nhân viên (NHANVIEN). Biết rằng một nhân viên gồm những thành phần như sau:

- Mã nhân viên (*MaSo*): chuỗi ký tự.
- Tên nhân viên (*HoTen*): chuỗi ký tự.
- Lương nhân viên (*Luong*): kiểu số thực.

b. Xuất mảng.

c. Tìm một nhân viên có lương cao nhất trong mảng.

d. Tính tổng lương của các nhân viên.

e. Sắp xếp mảng tăng dần theo lương nhân viên.

– Chương trình.

```
02221. #include <iostream>
02222. #include <iomanip>
02223. #include <string>
02224. using namespace std;
02225.
02226. struct NhanVien
02227. {
02228.    string MaSo;
02229.    string HoTen;
02230.    float Luong;
02231. };
02232. typedef struct NhanVien NHANVIEN;
02233.
02234. void Nhap(NHANVIEN&);
02235. void Xuat(NHANVIEN);
02236.
```

```
02237.  void Nhap(NHANVIEN[], int&);
02238.  void Xuat(NHANVIEN[], int);
02239.
02240.  float LuongCaoNhat(NHANVIEN[], int);
02241.  int TimNhanVien(NHANVIEN[], int);
02242.  float TongLuong(NHANVIEN[], int);
02243.
02244.  void HoanVi(NHANVIEN&, NHANVIEN&);
02245.  void SapXep(NHANVIEN[], int);
02246.
02247.  int main()
02248.  {
02249.     NHANVIEN b[100];
02250.     int n;
02251.     Nhap(b, n);
02252.     Xuat(b, n);
02253.
02254.     int kq = TimNhanVien(b, n);
02255.     if (kq == -1)
02256.        cout << "\nKhong tim thay";
02257.     else
02258.     {
02259.        cout << "\nNhan vien can tim: ";
02260.        Xuat(b[kq]);
02261.     }
02262.
02263.     float tong = TongLuong(b, n);
02264.     cout << "\nTong luong: " << tong;
02265.
02266.     SapXep(b, n);
02267.     cout << "\nMang sau khi da sap xep:";
02268.     Xuat(b, n);
02269.     return 0;
02270.  }
02271.
02272.  void Nhap(NHANVIEN& x)
02273.  {
02274.     cout << "Nhap ma so nhan vien: ";
02275.     cin.ignore();
02276.     getline(cin, x.MaSo);
02277.     cout << "Nhap ho ten nhan vien: ";
02278.     getline(cin, x.HoTen);
02279.     cout << "Nhap luong nhan vien: ";
```

```
02280.    cin >> x.Luong;
02281. }
02282.
02283. void Xuat(NHANVIEN x)
02284. {
02285.    cout << "\nMa nhan vien: " << x.MaSo;
02286.    cout << "\nHo ten: " << x.HoTen;
02287.    cout << setprecision(5);
02288.    cout << "\nLuong: " << x.Luong << endl;
02289. }
02290.
02291. void Nhap(NHANVIEN a[], int& n)
02292. {
02293.    cout << "\nNhap so luong nhan vien: ";
02294.    cin >> n;
02295.    for (int i = 0; i < n; i++)
02296.    {
02297.       cout << "\nNhan vien thu " << i + 1 <<
":\n";
02298.       Nhap(a[i]);
02299.    }
02300. }
02301.
02302. void Xuat(NHANVIEN a[], int n)
02303. {
02304.    for (int i = 0; i < n; i++)
02305.    {
02306.       cout << "\nNhan vien thu " << i + 1 << ":";
02307.       Xuat(a[i]);
02308.    }
02309. }
02310.
02311. float LuongCaoNhat(NHANVIEN a[], int n)
02312. {
02313.    float lc = a[0].Luong;
02314.    for (int i = 0; i < n; i++)
02315.       if (lc < a[i].Luong)
02316.          lc = a[i].Luong;
02317.    return lc;
02318. }
02319.
02320. int TimNhanVien(NHANVIEN a[], int n)
02321. {
```

```
02322.     if (n == 0)
02323.        return -1;
02324.     float ln = LuongCaoNhat(a, n);
02325.     for (int i = 0; i < n; i++)
02326.        if (ln == a[i].Luong)
02327.           return i;
02328.     return -1;
02329. }
02330.
02331. float TongLuong(NHANVIEN a[], int n)
02332. {
02333.     float s = 0;
02334.     for (int i = 0; i < n; i++)
02335.        s = s + a[i].Luong;
02336.     return s;
02337. }
02338.
02339. void HoanVi(NHANVIEN& a, NHANVIEN& b)
02340. {
02341.     NHANVIEN temp;
02342.     temp = a;
02343.     a = b;
02344.     b = temp;
02345. }
02346.
02347. void SapXep(NHANVIEN a[], int n)
02348. {
02349.     for (int i = 0; i < n - 1; i++)
02350.        for (int j = i + 1; j < n; j++)
02351.           if (a[i].Luong > a[j].Luong)
02352.              HoanVi(a[i], a[j]);
02353. }
```

Bài 017.Viết chương trình thực hiện những yêu cầu sau:

a. Nhập mảng một chiều các thí sinh (THISINH). Biết rằng một thí sinh gồm những thành phần như sau:

- Mã thí sinh (*MaSo*): chuỗi ký tự.
- Họ tên thí sinh (*HoTen*): chuỗi ký tự.
- Điểm toán (*Toan*): kiểu số thực.
- Điểm lý (*Ly*): kiểu số thực.
- Điểm hóa (*Hoa*): kiểu số thực.
- Điểm tổng cộng (*Tong*): kiểu số thực.

b. Xuất mảng.

c. Liệt kê các thí sinh thi đậu trong mảng. Một thí sinh được gọi là đậu khi có tổng điểm 3 môn lớn hơn 15 và không có môn nào bị điểm không.
d. Sắp xếp theo thứ tự giảm dần theo điểm tổng cộng.

– Chương trình.

```
02354. #include <iostream>
02355. #include <iomanip>
02356. #include <string>
02357. using namespace std;
02358.
02359. struct ThiSinh
02360. {
02361.    string MaSo;
02362.    string HoTen;
02363.    float Toan;
02364.    float Ly;
02365.    float Hoa;
02366.    float Tong;
02367. };
02368. typedef struct ThiSinh THISINH;
02369.
02370. void Nhap(THISINH&);
02371. void Xuat(THISINH);
02372. void XuLy(THISINH&);
02373.
02374. void Nhap(THISINH[], int&);
02375. void Xuat(THISINH[], int);
02376.
02377. int ktDau(THISINH);
02378. void LietKe(THISINH[], int);
02379. void SapXep(THISINH[], int);
02380.
02381. int main()
02382. {
02383.    THISINH b[100];
02384.    int n;
02385.    Nhap(b, n);
02386.    Xuat(b, n);
02387.
02388.    cout << "\nDanh sach cac thi sinh thi dau: ";
02389.    LietKe(b, n);
02390.
```

```
02391.     SapXep(b, n);
02392.     cout << "\nDanh sach sau khi sap xep: ";
02393.     Xuat(b, n);
02394.     return 0;
02395. }
02396.
02397. void Nhap(THISINH& x)
02398. {
02399.     cout << "Nhap ma thi sinh: ";
02400.     cin.ignore();
02401.     getline(cin, x.MaSo);
02402.     cout << "Nhap ho ten: ";
02403.     getline(cin, x.HoTen);
02404.     cout << "Nhap diem toan: ";
02405.     cin >> x.Toan;
02406.     cout << "Nhap diem ly: ";
02407.     cin >> x.Ly;
02408.     cout << "Nhap diem hoa: ";
02409.     cin >> x.Hoa;
02410. }
02411.
02412. void Xuat(THISINH x)
02413. {
02414.     cout << "\nMa thi sinh: " << x.MaSo << endl;
02415.     cout << "Ho ten: " << x.HoTen << endl;
02416.     cout << setprecision(3);
02417.     cout << "Diem toan: " << x.Toan << endl;
02418.     cout << "Diem ly: " << x.Ly << endl;
02419.     cout << "Diem hoa: " << x.Hoa << endl;
02420.     cout << "Diem tong: " << x.Tong << endl;
02421. }
02422.
02423. void XuLy(THISINH& x)
02424. {
02425.     x.Tong = x.Toan + x.Ly + x.Hoa;
02426. }
02427.
02428. void Nhap(THISINH a[], int& n)
02429. {
02430.     cout << "\nNhap so luong thi sinh: ";
02431.     cin >> n;
02432.     for (int i = 0; i < n; i++)
02433.     {
```

```
02434.       cout << "\nThi sinh thu " << i + 1 <<
":\n";
02435.       Nhap(a[i]);
02436.       XuLy(a[i]);
02437.    }
02438. }
02439.
02440. void Xuat(THISINH a[], int n)
02441. {
02442.    for (int i = 0; i < n; i++)
02443.    {
02444.       cout << "\nThi sinh thu " << i + 1 << ":";
02445.       Xuat(a[i]);
02446.    }
02447. }
02448.
02449. int ktDau(THISINH x)
02450. {
02451.    if (x.Tong <= 15)
02452.       return 0;
02453.    if (x.Toan == 0)
02454.       return 0;
02455.    if (x.Ly == 0)
02456.       return 0;
02457.    if (x.Hoa == 0)
02458.       return 0;
02459.    return 1;
02460. }
02461.
02462. void LietKe(THISINH a[], int n)
02463. {
02464.    for (int i = 0; i < n; i++)
02465.       if (ktDau(a[i]))
02466.          Xuat(a[i]);
02467. }
02468.
02469. void HoanVi(THISINH& a, THISINH& b)
02470. {
02471.    THISINH temp;
02472.    temp = a;
02473.    a = b;
02474.    b = temp;
02475. }
```

```
02476.
02477.  void SapXep(THISINH a[], int n)
02478.  {
02479.     for (int i = 0; i < n - 1; i++)
02480.        for (int j = i + 1; j < n; j++)
02481.           if (a[i].Tong < a[j].Tong)
02482.              HoanVi(a[i], a[j]);
02483.  }
```

Bài 018. Viết chương trình thực hiện những yêu cầu sau:

a. Nhập mảng một chiều các học sinh (HOCSINH). Biết rằng một học sinh gồm những thành phần như sau:

- Tên học sinh (*HoTen*): chuỗi ký tự.
- Điểm toán (*Toan*): kiểu số nguyên.
- Điểm văn (*Van*): kiểu số nguyên.
- Điểm trung bình (*TrungBinh*): kiểu số thực.

b. Xuất mảng.

c. Đếm số lượng học sinh giỏi trong mảng (toán và văn cùng lớn hơn 8).

– Chương trình.

```
02484.  #include <iostream>
02485.  #include <iomanip>
02486.  #include <string>
02487.  using namespace std;
02488.
02489.  struct HocSinh
02490.  {
02491.     string HoTen;
02492.     int Toan;
02493.     int Van;
02494.     float DiemTrungBinh;
02495.  };
02496.  typedef struct HocSinh HOCSINH;
02497.
02498.  void Nhap(HOCSINH&);
02499.  void Xuat(HOCSINH);
02500.  void XuLy(HOCSINH&);
02501.
02502.  void Nhap(HOCSINH[], int&);
02503.  void Xuat(HOCSINH[], int);
02504.
02505.  int ktGioi(HOCSINH);
```

```
02506.  int DemHocSinh(HOCSINH[], int);
02507.
02508.  int main()
02509.  {
02510.     HOCSINH b[100];
02511.     int n;
02512.     Nhap(b, n);
02513.     Xuat(b, n);
02514.
02515.     int kq = DemHocSinh(b, n);
02516.     cout << "\nSo luong hoc sinh gioi: " << kq;
02517.     return 0;
02518.  }
02519.
02520.  void Nhap(HOCSINH& x)
02521.  {
02522.     cout << "Nhap ho ten: ";
02523.     cin.ignore();
02524.     getline(cin, x.HoTen);
02525.     cout << "Nhap diem toan: ";
02526.     cin >> x.Toan;
02527.     cout << "Nhap diem van: ";
02528.     cin >> x.Van;
02529.  }
02530.
02531.  void Xuat(HOCSINH x)
02532.  {
02533.     cout << "Ho ten: " << x.HoTen << endl;
02534.     cout << "Diem toan: " << x.Toan << endl;
02535.     cout << "Diem van: " << x.Van << endl;
02536.     cout << "Diem trung binh: ";
02537.     cout << x.DiemTrungBinh << endl;
02538.  }
02539.
02540.  void XuLy(HOCSINH& x)
02541.  {
02542.     x.DiemTrungBinh = (float)(x.Toan + x.Van) / 2;
02543.  }
02544.
02545.  void Nhap(HOCSINH a[], int& n)
02546.  {
02547.     cout << "\nNhap so luong thi sinh: ";
02548.     cin >> n;
```

```
02549.     for (int i = 0; i < n; i++)
02550.     {
02551.        cout << "\nThi sinh thu " << i+1 << ":\n";
02552.        Nhap(a[i]);
02553.        XuLy(a[i]);
02554.     }
02555.  }
02556.
02557.  void Xuat(HOCSINH a[], int n)
02558.  {
02559.     for (int i = 0; i < n; i++)
02560.     {
02561.        cout << "\nThi sinh thu " << i + 1 << ":";
02562.        Xuat(a[i]);
02563.     }
02564.  }
02565.
02566.  int ktGioi(HOCSINH x)
02567.  {
02568.     if (x.Toan <= 8)
02569.        return 0;
02570.     if (x.Van <= 8)
02571.        return 0;
02572.     return 1;
02573.  }
02574.
02575.  int DemHocSinh(HOCSINH a[], int n)
02576.  {
02577.     int dem = 0;
02578.     for (int i = 0; i < n; i++)
02579.        if (ktGioi(a[i]))
02580.           dem = dem + 1;
02581.     return dem;
02582.  }
```

Bài 019.Viết chương trình thực hiện những yêu cầu sau:

a. Nhập mảng một chiều các đại lý (DAILY). Biết rằng một đại lý gồm những thành phần như sau:

+ Mã đại lý (*MaSo*): chuỗi ký tự.
+ Tên đại lý (*Ten*): chuỗi ký tự.
+ Điện thoại (*DienThoai*): kiểu số nguyên.
+ Ngày tiếp nhận (*NgayNhan*): kiểu dữ liệu ngày.
+ Địa chỉ (*DiaChi*): chuỗi ký tự.

+ E-Mail (*EMail*): chuỗi ký tự.
b. Xuất mảng.
c. Tìm một đại lý theo tên đại lý.

– Chương trình.

```
02583. #include <iostream>
02584. #include <iomanip>
02585. #include <string>
02586. using namespace std;
02587.
02588. struct Ngay
02589. {
02590.    int Ngay;
02591.    int Thang;
02592.    int Nam;
02593. };
02594. typedef struct Ngay NGAY;
02595.
02596. struct DaiLy
02597. {
02598.    string MaSo;
02599.    string TenDaiLy;
02600.    int DienThoai;
02601.    NGAY NgayNhan;
02602.    string DiaChi;
02603.    string EMail;
02604. };
02605. typedef struct DaiLy DAILY;
02606.
02607. void Nhap(NGAY&);
02608. void Xuat(NGAY);
02609.
02610. void Nhap(DAILY&);
02611. void Xuat(DAILY);
02612.
02613. void Nhap(DAILY[], int&);
02614. void Xuat(DAILY[], int);
02615.
02616. int TimDaiLy(DAILY[], int, string);
02617.
02618. int main()
02619. {
02620.    DAILY b[100];
```

```
02621.     int n;
02622.     Nhap(b, n);
02623.     Xuat(b, n);
02624.
02625.     string TenDaiLy;
02626.     cout << "\nNhap ten dai ly can tim: ";
02627.     getline(cin, TenDaiLy);
02628.
02629.     int kq = TimDaiLy(b, n, TenDaiLy);
02630.     if (kq == -1)
02631.        cout << "\nKhong tim thay";
02632.     else
02633.     {
02634.        cout << "\nDai ly can tim: ";
02635.        Xuat(b[kq]);
02636.     }
02637.     return 0;
02638. }
02639.
02640. void Nhap(NGAY& x)
02641. {
02642.     cout << "\nNhap ngay:";
02643.     cin >> x.Ngay;
02644.     cout << "Nhap thang:";
02645.     cin >> x.Thang;
02646.     cout << "Nhap nam:";
02647.     cin >> x.Nam;
02648. }
02649.
02650. void Xuat(NGAY x)
02651. {
02652.     cout << "\nNgay: " << x.Ngay << endl;
02653.     cout << "Thang: " << x.Thang << endl;
02654.     cout << "Nam: " << x.Nam << endl;
02655. }
02656.
02657. void Nhap(DAILY& x)
02658. {
02659.     cout << "Nhap ma so dai ly: ";
02660.     cin.ignore();
02661.     getline(cin, x.MaSo);
02662.     cout << "Nhap ten dai ly: ";
02663.     getline(cin, x.TenDaiLy);
```

```
02664.     cout << "Nhap so dien thoai: ";
02665.     cin >> x.DienThoai;
02666.     cout << "Nhap ngay tiep nhan: ";
02667.     Nhap(x.NgayNhan);
02668.     cout << "Nhap dia chi: ";
02669.     cin.ignore();
02670.     getline(cin, x.DiaChi);
02671.     cout << "Nhap E-Mail: ";
02672.     getline(cin, x.EMail);
02673. }
02674.
02675. void Xuat(DAILY x)
02676. {
02677.     cout << "\nMa so dai ly: " << x.MaSo;
02678.     cout << "\nTen dai ly: " << x.TenDaiLy;
02679.     cout << "\nSo dien thoai: " << x.DienThoai;
02680.     cout << "\nNgay tiep nhan: ";
02681.     Xuat(x.NgayNhan);
02682.     cout << "Dia chi: " << x.DiaChi;
02683.     cout << "\nE-mail: " << x.EMail;
02684. }
02685.
02686. void Nhap(DAILY a[], int& n)
02687. {
02688.     cout << "\nNhap so luong dai ly: ";
02689.     cin >> n;
02690.     for (int i = 0; i < n; i++)
02691.     {
02692.        cout << "\nDai ly thu " << i + 1 << ":\n";
02693.        Nhap(a[i]);
02694.     }
02695. }
02696.
02697. void Xuat(DAILY a[], int n)
02698. {
02699.     for (int i = 0; i < n; i++)
02700.     {
02701.        cout << "\nDai ly thu " << i + 1 << ":";
02702.        Xuat(a[i]);
02703.     }
02704. }
02705.
02706. int TimDaiLy(DAILY a[], int n, string x)
```

```
02707. {
02708.    for (int i = 0; i < n; i++)
02709.       if (a[i].TenDaiLy == x)
02710.          return i;
02711.    return -1;
02712. }
```

01.07 CÁC KIỂU DỮ LIỆU THƯỜNG GẶP TRONG BÀI TẬP LẬP TRÌNH

01.07.01 Điểm trong mặt phẳng Oxy – Point

> Bài cơ sở 010. Hãy khai báo kiểu dữ liệu biểu diễn khái niệm điểm trong mặt phẳng Oxy và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

– Khai báo kiểu dữ liệu biểu diễn điểm

```
02713. struct Diem
02714. {
02715.    float x;
02716.    float y;
02717. };
02718. typedef struct Diem DIEM;
```

– Định nghĩa hàm nhập điểm.

```
02719. void Nhap(DIEM &P)
02720. {
02721.    cout << "Nhap x: ";
02722.    cin >> P.x;
02723.    cout << "Nhap y: ";
02724.    cin >> P.y;
02725. }
```

– Định nghĩa hàm xuất điểm.

```
02726. void Xuat(DIEM P)
02727. {
02728.    cout << setw(6);
02729.    cout << setprecision(3);
02730.    cout << "\n x = " << P.x;
02731.    cout << "\n y = " << P.y;
02732. }
```

Bài 001.Hãy viết chương trình nhập và xuất thông tin của một điểm trong mặt phẳng Oxy (DIEM).

– Chương trình.

```
02733. #include <iostream>
02734. using namespace std;
02735.
02736. struct diem
02737. {
02738.    float x;
02739.    float y;
02740. };
02741. typedef struct diem DIEM;
02742.
02743. void Nhap(DIEM&);
02744. void Xuat(DIEM);
02745.
02746. int main()
02747. {
02748.    DIEM M;
02749.    Nhap(M);
02750.    cout << "\nToa do diem vua nhap:";
02751.    Xuat(M);
02752.    return 1;
02753. }
02754.
02755. void Nhap(DIEM& P)
02756. {
02757.    cout << "Nhap x: ";
02758.    cin >> P.x;
02759.    cout << "Nhap y: ";
02760.    cin >> P.y;
02761. }
02762.
02763. void Xuat(DIEM P)
02764. {
02765.    cout << "\nx: " << P.x;
02766.    cout << "\ny: " << P.y;
02767. }
```

01.07.02 Điểm trong không gian Oxyz

Bài cơ sở 011. Hãy khai báo kiểu dữ liệu biểu diễn khái niệm điểm trong không gian Oxyz và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

– Khai báo kiểu dữ liệu biểu diễn điểm không gian.

```
02768.  struct DiemKhongGian
02769.  {
02770.     float x;
02771.     float y;
02772.     float z;
02773.  };
02774.  typedef struct DiemKhongGian DIEMKHONGGIAN;
```

– Định nghĩa hàm nhập điểm không gian.

```
02775.  void Nhap(DIEMKHONGGIAN &P)
02776.  {
02777.     cout << "Nhap x: ";
02778.     cin >> P.x;
02779.     cout << "Nhap y: ";
02780.     cin >> P.y;
02781.     cout << "Nhap z: ";
02782.     cin >> P.z;
02783.  }
```

– Định nghĩa hàm xuất điểm không gian.

```
02784.  void Xuat(DIEMKHONGGIAN P)
02785.  {
02786.     cout << setw(6);
02787.     cout << setprecision(3);
02788.     cout << "\n x = " << P.x;
02789.     cout << "\n y = " << P.y;
02790.     cout << "\n z = " << P.z;
02791.  }
```

Bài 002.Hãy viết chương trình nhập và xuất thông tin của một điểm trong không gian Oxyz (DIEMKHONGGIAN).

– Chương trình.

```
02792.  #include <iostream>
02793.  using namespace std;
02794.
02795.  struct DiemKhongGian
```

```
02796. {
02797.     float x;
02798.     float y;
02799.     float z;
02800. };
02801. typedef struct diemkhonggian DIEMKHONGGIAN;
02802.
02803. void Nhap(DIEMKHONGGIAN&);
02804. void Xuat(DIEMKHONGGIAN);
02805.
02806. int main()
02807. {
02808.     DIEMKHONGGIAN M;
02809.     Nhap(M);
02810.     cout << "\nToa do diem khong gian vua nhap:";
02811.     Xuat(M);
02812.     return 1;
02813. }
02814.
02815. void Nhap(DIEMKHONGGIAN& P)
02816. {
02817.     cout << "Nhap x: ";
02818.     cin >> P.x;
02819.     cout << "Nhap y: ";
02820.     cin >> P.y;
02821.     cout << "Nhap z: ";
02822.     cin >> P.z;
02823. }
02824.
02825. void Xuat(DIEMKHONGGIAN P)
02826. {
02827.     cout << "\nx: " << P.x;
02828.     cout << "\ny: " << P.y;
02829.     cout << "\nz: " << P.z;
02830. }
```

01.07.03 Phân số – Fraction

Bài cơ sở 012. Hãy khai báo kiểu dữ liệu biểu diễn khái niệm phân số trong toán học và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

– Khai báo kiểu dữ liệu biểu diễn phân số.

```
02831.  struct PhanSo
02832.  {
02833.     int Tu;
02834.     int Mau;
02835.  };
02836.  typedef struct PhanSo PHANSO;
```

– Định nghĩa hàm nhập phân số.

```
02837.  void Nhap(PHANSO &x)
02838.  {
02839.     cout << "Nhap tu: ";
02840.     cin >> x.Tu;
02841.     cout << "Nhap mau: ";
02842.     cin >> x.Mau;
02843.  }
```

– Định nghĩa hàm xuất phân số (cách cơ bản).

```
02844.  void Xuat(PHANSO x)
02845.  {
02846.     cout << setw(6);
02847.     cout << "\n Tu: " << x.Tu;
02848.     cout << "\n Mau: " << x.Mau;
02849.  }
```

– Định nghĩa hàm xuất phân số.

```
02850.  void Xuat(PHANSO x)
02851.  {
02852.     cout << setw(6);
02853.     cout << x.Tu << "/" << x.Mau << endl;
02854.  }
```

Bài 003.Hãy viết chương trình nhập và xuất thông tin của một phân số (PHANSO).

– Chương trình.

```
02855.  #include <iostream>
02856.  using namespace std;
02857.
02858.  struct PhanSo
02859.  {
02860.     int Tu;
02861.     int Mau;
02862.  };
02863.  typedef struct PhanSo PHANSO;
```

```
02864.
02865.  void Nhap(PHANSO&);
02866.  void Xuat(PHANSO);
02867.
02868.  int main()
02869.  {
02870.     PHANSO M;
02871.     Nhap(M);
02872.     cout << "\nPhan so vua nhap:";
02873.     Xuat(M);
02874.     return 1;
02875.  }
02876.
02877.  void Nhap(PHANSO& x)
02878.  {
02879.     cout << "Nhap tu: ";
02880.     cin >> x.Tu;
02881.     cout << "Nhap mau: ";
02882.     cin >> x.Mau;
02883.  }
02884.
02885.  void Xuat(PHANSO x)
02886.  {
02887.     cout << "\nTu: " << x.Tu;
02888.     cout << "\nMau: " << x.Mau;
02889.  }
```

01.07.04 Số phức – Complex numbers

Bài cơ sở 013. Hãy khai báo kiểu dữ liệu biểu diễn khái niệm số phức trong toán học và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

– Khai báo kiểu dữ liệu biểu diễn số phức.

```
02890.  struct SoPhuc
02891.  {
02892.     float Thuc;
02893.     float Ao;
02894.  };
02895.  typedef struct SoPhuc SOPHUC;
```

– Định nghĩa hàm nhập số phức.

```
02896.  void Nhap(SOPHUC &x)
```

```
{
    cout << "Nhap phan thuc: ";
    cin >> x.Thuc;
    cout << "Nhap phan ao: ";
    cin >> x.Ao;
}
```

– Định nghĩa hàm xuất số phức.

```
void Xuat(SOPHUC x)
{
    cout << setw(6);
    cout << setprecision(3);
    cout << "\n Thuc = " << x.Thuc;
    cout << "\n Ao = " << x.Ao;
}
```

Bài 004.Hãy viết chương trình nhập và xuất thông tin của một số phức (SOPHUC).

– Chương trình.

```
#include <iostream>
using namespace std;

struct SoPhuc
{
    float Thuc;
    float Ao;
};
typedef struct SoPhuc SOPHUC;

void Nhap(SOPHUC&);
void Xuat(SOPHUC);

int main()
{
    SOPHUC M;
    Nhap(M);
    cout << "\nSo phuc vua nhap:";
    Xuat(M);
    return 1;
}

void Nhap(SOPHUC& x)
{
```

```
02934.     cout << "Nhap thuc: ";
02935.     cin >> x.Thuc;
02936.     cout << "Nhap ao: ";
02937.     cin >> x.Ao;
02938.  }
02939.
02940.  void Xuat(SOPHUC x)
02941.  {
02942.     cout << "\nThuc: " << x.Thuc;
02943.     cout << "\nAo: " << x.Ao;
02944.  }
```

01.07.05 Hỗn số – Mixed numbers

> Bài cơ sở 014. Hãy khai báo kiểu dữ liệu biểu diễn khái niệm hỗn số trong toán học và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

– Khai báo kiểu dữ liệu biểu diễn hỗn số.

```
02945.  struct HonSo
02946.  {
02947.     int Nguyen;
02948.     int Tu;
02949.     int Mau;
02950.  };
02951.  typedef struct HonSo HONSO;
```

– Định nghĩa hàm nhập hỗn số.

```
02952.  void Nhap(HONSO &x)
02953.  {
02954.     cout << "Nhap nguyen: ";
02955.     cin >> x.Nguyen;
02956.     cout << "Nhap tu: ";
02957.     cin >> x.Tu;
02958.     cout << "Nhap mau: ";
02959.     cin >> x.Mau;
02960.  }
```

– Định nghĩa hàm xuất hỗn số.

```
02961.  void Xuat(HONSO x)
02962.  {
02963.     cout << setw(6);
02964.     cout << "\n Nguyen: " << x.Nguyen;
02965.     cout << "\n Tu: " << x.Tu;
```

```
02966.     cout << "\n Mau: " << x.Mau;
02967.  }
```

Bài 005.Hãy viết chương trình nhập và xuất thông tin của một hỗn số (HONSO).

– Chương trình.

```
02968.  #include <iostream>
02969.  using namespace std;
02970.
02971.  struct HonSo
02972.  {
02973.     int Nguyen;
02974.     int Tu;
02975.     int Mau;
02976.  };
02977.  typedef struct HonSo HONSO;
02978.
02979.  void Nhap(HONSO&);
02980.  void Xuat(HONSO);
02981.
02982.  int main()
02983.  {
02984.     HONSO M;
02985.     Nhap(M);
02986.     cout << "\nHon so vua nhap:";
02987.     Xuat(M);
02988.     return 1;
02989.  }
02990.
02991.  void Nhap(HONSO& x)
02992.  {
02993.     cout << "Nhap nguyen: ";
02994.     cin >> x.Nguyen;
02995.     cout << "Nhap tu: ";
02996.     cin >> x.Tu;
02997.     cout << "Nhap mau: ";
02998.     cin >> x.Mau;
02999.  }
03000.
03001.  void Xuat(HONSO x)
03002.  {
03003.     cout << "\nNguyen: " << x.Nguyen;
```

```
03004.     cout << "\nTu: " << x.Tu;
03005.     cout << "\nMau: " << x.Mau;
03006.  }
```

01.07.06 Thời gian – Time

> Bài cơ sở 015. Hãy khai báo kiểu dữ liệu biểu diễn ngày trong thế giới thực và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

- Kiến thức phổ thông
  - + Một ngày có 24 giờ được đánh số từ 0 cho đến 23.
  - + Một giờ có 60 phút được đánh số từ 0 cho đến 59.
  - + Một phút có 60 giây được đánh số từ 0 cho đến 59.
- Khai báo kiểu dữ liệu biểu diễn thời gian.

```
03007.  struct ThoiGian
03008.  {
03009.     int Gio;
03010.     int Phut;
03011.     int Giay;
03012.  };
03013.  typedef struct ThoiGian THOIGIAN;
```

- Định nghĩa hàm nhập thời gian.

```
03014.  void Nhap(THOIGIAN &x)
03015.  {
03016.     cout << "Nhap gio : ";
03017.     cin >> x.Gio;
03018.     cout << "Nhap phut : ";
03019.     cin >> x.Phut;
03020.     cout << "Nhap giay : ";
03021.     cin >> x.Giay;
03022.  }
```

- Định nghĩa hàm xuất thời gian.

```
03023.  void Xuat(THOIGIAN x)
03024.  {
03025.     cout << setw(6);
03026.     cout << "\n Gio = " << x.Gio;
03027.     cout << "\n Phut = " << x.Phut;
03028.     cout << "\n Giay = " << x.Giay;
03029.  }
```

Bài 006.Hãy viết chương trình nhập và xuất thông tin của một thời gian (THOIGIAN).

– Chương trình.

```
03030. #include <iostream>
03031. using namespace std;
03032.
03033. struct ThoiGian
03034. {
03035.     int Gio;
03036.     int Phut;
03037.     int Giay;
03038. };
03039. typedef struct ThoiGian THOIGIAN;
03040.
03041. void Nhap(THOIGIAN&);
03042. void Xuat(THOIGIAN);
03043.
03044. int main()
03045. {
03046.     THOIGIAN t;
03047.     Nhap(t);
03048.     cout << "\nThoi gian vua nhap:";
03049.     Xuat(t);
03050.     return 1;
03051. }
03052.
03053. void Nhap(THOIGIAN& x)
03054. {
03055.     cout << "Nhap gio: ";
03056.     cin >> x.Gio;
03057.     cout << "Nhap phut: ";
03058.     cin >> x.Phut;
03059.     cout << "Nhap giay: ";
03060.     cin >> x.Giay;
03061. }
03062.
03063. void Xuat(THOIGIAN x)
03064. {
03065.     cout << "\nGio: " << x.Gio;
03066.     cout << "\nPhut: " << x.Phut;
03067.     cout << "\nGiay: " << x.Giay;
03068. }
```

01.07.07 Ngày – Date

Bài cơ sở 016. Hãy khai báo kiểu dữ liệu biểu diễn ngày trong thế giới thực và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

– Khai báo kiểu dữ liệu biểu diễn ngày.

```
03069.  struct Ngay
03070.  {
03071.     int Ngay;
03072.     int Thang;
03073.     int Nam;
03074.  };
03075.  typedef struct Ngay NGAY;
```

– Định nghĩa hàm nhập ngày.

```
03076.  void Nhap(NGAY &x)
03077.  {
03078.     cout << "Nhap ngay : ";
03079.     cin >> x.Ngay;
03080.     cout << "Nhap thang : ";
03081.     cin >> x.Thang;
03082.     cout << "Nhap nam : ";
03083.     cin >> x.Nam;
03084.  }
```

– Định nghĩa hàm xuất ngày.

```
03085.  void Xuat(NGAY x)
03086.  {
03087.     cout << setw(6);
03088.     cout << "\n Ngay = " << x.Ngay;
03089.     cout << "\n Thang = " << x.Thang;
03090.     cout << "\n Nam = " << x.Nam;
03091.  }
```

Bài 007.Hãy viết chương trình nhập và xuất thông tin của một ngày (NGÀY).

– Chương trình.

```
03092.  #include <iostream>
03093.  using namespace std;
03094.
03095.  struct Ngay
03096.  {
```

```
03097.     int Ngay;
03098.     int Thang;
03099.     int Nam;
03100.  };
03101.  typedef struct Ngay NGAY;
03102.
03103.  void Nhap(NGAY&);
03104.  void Xuat(NGAY);
03105.
03106.  int main()
03107.  {
03108.     NGAY d;
03109.     Nhap(d);
03110.     cout << "\nNgay vua nhap:";
03111.     Xuat(d);
03112.     return 1;
03113.  }
03114.
03115.  void Nhap(NGAY &x)
03116.  {
03117.     cout << "Nhap ngay : ";
03118.     cin >> x.Ngay;
03119.     cout << "Nhap thang : ";
03120.     cin >> x.Thang;
03121.     cout << "Nhap nam : ";
03122.     cin >> x.Nam;
03123.  }
03124.
03125.  void Xuat(NGAY x)
03126.  {
03127.     cout << setw(6);
03128.     cout << "\n Ngay = " << x.Ngay;
03129.     cout << "\n Thang = " << x.Thang;
03130.     cout << "\n Nam = " << x.Nam;
03131.  }
```

01.07.08 Đơn thức – Monomial

Bài cơ sở 017. Hãy khai báo kiểu dữ liệu biểu diễn khái niệm đơn thức một biến $f(x) = ax^n$ trong toán học và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

- Ví dụ 01: Xét đơn thức $17.3x^{10}$.
  + Trong đơn thức này có
    - Hệ số là 17.3
    - Bậc lũy thừa là 10.
- Ví dụ 02: 0 được gọi là đơn thức không.
- Ví dụ 03: x là đơn thức có hệ số là 1, bậc lũy thừa là 1.

- Dạng tổng quát của đơn thức một biến theo x là $f(x) = ax^n$.
- Trong đó a được gọi là hệ số, n được gọi là bậc lũy thừa.

- Khai báo kiểu dữ liệu biểu diễn đơn thức.

```
03132.  struct DonThuc
03133.  {
03134.      float a;
03135.      int n;
03136.  };
03137.  typedef struct DonThuc DONTHUC;
```

- Định nghĩa hàm nhập đơn thức.

```
03138.  void Nhap(DONTHUC &f)
03139.  {
03140.      cout << "Nhap he so: ";
03141.      cin >> f.a;
03142.      cout << "Nhap so mu: ";
03143.      cin >> f.n;
03144.  }
```

- Định nghĩa hàm xuất đơn thức.

```
03145.  void Xuat(DONTHUC f)
03146.  {
03147.      cout << setw(6);
03148.      cout << setprecision(3);
03149.      cout << "\n a = " << f.a;
03150.      cout << "\n n = " << f.n;
03151.  }
```

Bài 008.Hãy viết chương trình nhập và xuất thông tin của một đơn thức (DONTHUC).

- Chương trình.

```
03152.  #include <iostream>
03153.  using namespace std;
03154.
03155.  struct DonThuc
```

```
03156.  {
03157.     float a;
03158.     int n;
03159.  };
03160.  typedef struct DonThuc DONTHUC;
03161.
03162.  void Nhap(DONTHUC&);
03163.  void Xuat(DONTHUC);
03164.
03165.  int main()
03166.  {
03167.     DONTHUC g;
03168.     Nhap(g);
03169.     cout << "\nDon thuc vua nhap:";
03170.     Xuat(g);
03171.     return 1;
03172.  }
03173.
03174.  void Nhap(DONTHUC& f)
03175.  {
03176.     cout << "Nhap he so: ";
03177.     cin >> f.a;
03178.     cout << "Nhap so mu: ";
03179.     cin >> f.n;
03180.  }
03181.
03182.  void Xuat(DONTHUC f)
03183.  {
03184.     cout << "\nHe so: " << f.a;
03185.     cout << "\nSo mu: " << f.n;
03186.  }
```

01.07.09 Đường tròn trong mặt phẳng Oxy – Circle

Bài cơ sở 018. Hãy khai báo kiểu dữ liệu biểu diễn khái niệm đường tròn trong mặt phẳng Oxy và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

– Khai báo kiểu dữ liệu để biểu diễn đường tròn.

```
03187.  struct Diem
03188.  {
03189.     float x;
```

```
03190.    float y;
03191. };
03192. typedef struct Diem DIEM;
03193.
03194. struct DuongTron
03195. {
03196.    DIEM I;
03197.    float R;
03198. };
03199. typedef struct DuongTron DUONGTRON;
```

– Định nghĩa hàm nhập điểm.

```
03200. void Nhap(DIEM &P)
03201. {
03202.    cout << "Nhap x: ";
03203.    cin >> P.x;
03204.    cout << "Nhap y: ";
03205.    cin >> P.y;
03206. }
```

– Định nghĩa hàm xuất điểm.

```
03207. void Xuat(DIEM P)
03208. {
03209.    cout << setw(6);
03210.    cout << setprecision(3);
03211.    cout << "\n x = " << P.x;
03212.    cout << "\n y = " << P.y;
03213. }
```

– Định nghĩa hàm nhập đường tròn.

```
03214. void Nhap(DUONGTRON &c)
03215. {
03216.    cout << "Nhap tam: "<<endl;
03217.    Nhap(c.I);
03218.    cout << "Nhap ban kinh: ";
03219.    cin >> c.R;
03220. }
```

– Định nghĩa hàm xuất đường tròn.

```
03221. void Xuat(DUONGTRON c)
03222. {
03223.    cout << "Tam: "<<endl;
03224.    Xuat(c.I);
03225.    cout << setw(6);
03226.    cout << setprecision(3);
```

```
03227.     cout << "R = "<<c.R;
03228.  }
```

Bài 009.Hãy viết chương trình nhập và xuất thông tin của một đường tròn (DUONGTRON).

– Chương trình.

```
03229.  #include <iostream>
03230.  using namespace std;
03231.
03232.  struct Diem
03233.  {
03234.     float x;
03235.     float y;
03236.  };
03237.  typedef struct Diem DIEM;
03238.
03239.  struct DuongTron
03240.  {
03241.     DIEM I;
03242.     float R;
03243.  };
03244.  typedef struct DuongTron DUONGTRON;
03245.
03246.  void Nhap(DIEM&);
03247.  void Xuat(DIEM);
03248.
03249.  void Nhap(DUONGTRON&);
03250.  void Xuat(DUONGTRON);
03251.
03252.  int main()
03253.  {
03254.     DUONGTRON cc;
03255.     Nhap(cc);
03256.
03257.     cout << "\nDuong tron vua nhap:";
03258.     Xuat(M);
03259.     return 1;
03260.  }
03261.
03262.  void Nhap(DIEM& P)
03263.  {
03264.     cout << "Nhap x: ";
```

```
03265.    cin >> P.x;
03266.    cout << "Nhap y: ";
03267.    cin >> P.y;
03268. }
03269.
03270. void Xuat(DIEM P)
03271. {
03272.    cout << "\nx: " << P.x;
03273.    cout << "\ny: " << P.y;
03274. }
03275.
03276. void Nhap(DUONGTRON& c)
03277. {
03278.    cout << "Nhap tam:\n";
03279.    Nhap(c.I);
03280.    cout << "Nhap ban kinh: ";
03281.    cin >> c.R;
03282. }
03283.
03284. void Xuat(DUONGTRON c)
03285. {
03286.    cout << "\nTam: ";
03287.    Xuat(c.I);
03288.    cout << "\nBan kinh: " << c.R;
03289. }
```

01.07.10 Hình cầu trong không gian Oxyz – Sphere

Bài cơ sở 019. Hãy khai báo kiểu dữ liệu biểu diễn khái niệm hình cầu trong không gian Oxyz và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

– Khai báo kiểu dữ liệu để biểu diễn hình cầu.

```
03290. struct DiemKhongGian
03291. {
03292.    float x;
03293.    float y;
03294.    float y;
03295. };
03296. typedef struct DiemKhongGian DIEMKHONGGIAN;
03297.
03298. struct HinhCau
```

```
{
   DIEMKHONGGIAN I;
   float R;
};
typedef struct HinhCau HINHCAU;
```

– Định nghĩa hàm nhập điểm không gian.

```
void Nhap(DIEMKHONGGIAN &P)
{
   cout << "Nhap x: ";
   cin >> P.x;
   cout << "Nhap y: ";
   cin >> P.y;
   cout << "Nhap z: ";
   cin >> P.z;
}
```

– Định nghĩa hàm xuất điểm không gian.

```
void Xuat(DIEMKHONGGIAN P)
{
   cout << setw(6);
   cout << setprecision(3);
   cout << "\n x = " << P.x;
   cout << "\n y = " << P.y;
   cout << "\n z = " << P.z;
}
```

– Định nghĩa hàm nhập hình cầu.

```
void Nhap(HINHCAU &c)
{
   cout << "Nhap tam: "<<endl;
   Nhap(c.I);
   cout << "Nhap ban kinh: ";
   cin >> c.R;
}
```

– Định nghĩa hàm xuất hình cầu.

```
void Xuat(HINHCAU c)
{
   cout << setw(6);
   cout << setprecision(3);
   cout << "Tam: " << endl;
   Xuat(c.I);
   cout << "R = " << c.R;
}
```

Bài 010.Hãy viết chương trình nhập và xuất thông tin của một hình cầu (HINHCAU).

– Chương trình.

```
03336.  #include <iostream>
03337.  using namespace std;
03338.
03339.  struct DiemKhongGian
03340.  {
03341.     float x;
03342.     float y;
03343.     float z;
03344.  };
03345.  typedef struct DiemKhongGian DIEMKHONGGIAN;
03346.
03347.  struct HinhCau
03348.  {
03349.     DIEMKHONGGIAN I;
03350.     float R;
03351.  };
03352.  typedef struct HinhCau HINHCAU;
03353.
03354.  void Nhap(DIEMKHONGGIAN&);
03355.  void Xuat(DIEMKHONGGIAN);
03356.
03357.  void Nhap(HINHCAU&);
03358.  void Xuat(HINHCAU);
03359.
03360.  int main()
03361.  {
03362.     HINHCAU s;
03363.     Nhap(s);
03364.
03365.     cout << "\nHinh cau vua nhap:";
03366.     Xuat(s);
03367.     return 1;
03368.  }
03369.
03370.  void Nhap(DIEMKHONGGIAN& P)
03371.  {
03372.     cout << "Nhap x: ";
03373.     cin >> P.x;
03374.     cout << "Nhap y: ";
```

```
03375.    cin >> P.y;
03376.    cout << "Nhap z: ";
03377.    cin >> P.z;
03378. }
03379.
03380. void Xuat(DIEMKHONGGIAN P)
03381. {
03382.    cout << "\nx: " << P.x;
03383.    cout << "\ny: " << P.y;
03384.    cout << "\nz: " << P.z;
03385. }
03386.
03387. void Nhap(HINHCAU& c)
03388. {
03389.    cout << "Nhap tam:\n";
03390.    Nhap(c.I);
03391.    cout << "Nhap ban kinh: ";
03392.    cin >> c.R;
03393. }
03394.
03395. void Xuat(HINHCAU c)
03396. {
03397.    cout << setw(6);
03398.    cout << setprecision(3);
03399.    cout << "Tam: " << endl;
03400.    Xuat(c.I);
03401.    cout << "\nBan kinh: " << c.R;
03402. }
```

01.07.11 Tam giác – Triangle

Bài cơ sở 020. Hãy khai báo kiểu dữ liệu biểu diễn khái niệm tam giác trong mặt phẳng Oxy và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

– Khai báo kiểu dữ liệu để biểu diễn tam giác.

```
03403. struct Diem
03404. {
03405.    float x;
03406.    float y;
03407. };
03408. typedef struct Diem DIEM;
```

```
03409.
03410.  struct TamGiac
03411.  {
03412.     DIEM A;
03413.     DIEM B;
03414.     DIEM C;
03415.  };
03416.  typedef struct TamGiac TAMGIAC;
```

– Định nghĩa hàm nhập điểm.

```
03417.  void Nhap(DIEM &P)
03418.  {
03419.     cout << "Nhap x: ";
03420.     cin >> P.x;
03421.     cout << "Nhap y: ";
03422.     cin >> P.y;
03423.  }
```

– Định nghĩa hàm xuất điểm.

```
03424.  void Xuat(DIEM P)
03425.  {
03426.     cout << setw(6);
03427.     cout << setprecision(3);
03428.     cout << "\n x = " << P.x;
03429.     cout << "\n y = " << P.y;
03430.  }
```

– Định nghĩa hàm nhập tam giác.

```
03431.  void Nhap(TAMGIAC &t)
03432.  {
03433.     cout << "A: "<<endl;
03434.     Nhap(t.A);
03435.     cout << "B: "<<endl;
03436.     Nhap(t.B);
03437.     cout << "C: "<<endl;
03438.     Nhap(t.C);
03439.  }
```

– Định nghĩa hàm xuất tam giác.

```
03440.  void Xuat(TAMGIAC t)
03441.  {
03442.     cout << "A: "<<endl;
03443.     Xuat(t.A);
03444.     cout << "B: "<<endl;
03445.     Xuat(t.B);
```

```
03446.     cout << "C: "<<endl;
03447.     Xuat(t.C);
03448.  }
```

Bài 011.Hãy viết chương trình nhập và xuất thông tin của một tam giác (TAMGIAC).

– Chương trình.

```
03449.  #include <iostream>
03450.  using namespace std;
03451.
03452.  struct Diem
03453.  {
03454.     float x;
03455.     float y;
03456.  };
03457.  typedef struct Diem DIEM;
03458.
03459.  struct TamGiac
03460.  {
03461.     DIEM A;
03462.     DIEM B;
03463.     DIEM C;
03464.  };
03465.  typedef struct TamGiac TAMGIAC;
03466.
03467.  void Nhap(DIEM&);
03468.  void Xuat(DIEM);
03469.
03470.  void Nhap(TAMGIAC&);
03471.  void Xuat(TAMGIAC);
03472.
03473.  int main()
03474.  {
03475.     TAMGIAC M;
03476.     Nhap(M);
03477.     cout << "\nTam giac vua nhap:";
03478.     Xuat(M);
03479.     return 1;
03480.  }
03481.
03482.  void Nhap(DIEM& P)
03483.  {
```

```
03484.     cout << "Nhap x: ";
03485.     cin >> P.x;
03486.     cout << "Nhap y: ";
03487.     cin >> P.y;
03488. }
03489.
03490. void Xuat(DIEM P)
03491. {
03492.     cout << setw(6);
03493.     cout << setprecision(3);
03494.     cout << "\nx: " << P.x;
03495.     cout << "\ny: " << P.y;
03496. }
03497.
03498. void Nhap(TAMGIAC& t)
03499. {
03500.     cout << "Nhap diem A:\n";
03501.     Nhap(t.A);
03502.     cout << "Nhap diem B:\n";
03503.     Nhap(t.B);
03504.     cout << "Nhap diem C:\n";
03505.     Nhap(t.C);
03506. }
03507.
03508. void Xuat(TAMGIAC t)
03509. {
03510.     cout << "\nToa do diem A: ";
03511.     Xuat(t.A);
03512.     cout << "\nToa do diem B: ";
03513.     Xuat(t.B);
03514.     cout << "\nToa do diem C: ";
03515.     Xuat(t.C);
03516. }
```

01.07.12 Đường thẳng trong mặt phẳng Oxy – Line

Bài cơ sở 021. Hãy khai báo kiểu dữ liệu biểu diễn khái niệm đường thẳng $ax + by + c = 0$ trong mặt phẳng Oxy và định nghĩa hàm nhập, hàm xuất cho kiểu dữ liệu này.

– Khai báo kiểu dữ liệu biểu diễn đường thẳng.

```
03517. struct DuongThang
```

```
{
    float a;
    float b;
    float c;
};
typedef struct DuongThang DUONGTHANG;
```

– Định nghĩa hàm nhập đường thẳng.

```
void Nhap(DUONGTHANG &d)
{
    cout << "Nhap a: ";
    cin >> d.a;
    cout << "Nhap b: ";
    cin >> d.b;
    cout << "Nhap c: ";
    cin >> d.c;
}
```

– Định nghĩa hàm xuất đường thẳng.

```
void Xuat(DUONGTHANG d)
{
    cout << setw(6);
    cout << setprecision(3);
    cout << "\n a = " << d.a;
    cout << "\n b = " << d.b;
    cout << "\n c = " << d.c;
}
```

Bài 012.Hãy viết chương trình nhập và xuất thông tin của một đường thẳng (DUONGTHANG).

– Chương trình.

```
#include <iostream>
using namespace std;

struct DuongThang
{
    float a;
    float b;
    float c;
};
typedef struct DuongThang DUONGTHANG;

void Nhap(DUONGTHANG &);
void Xuat(DUONGTHANG);
```

```
03554.
03555.  int main()
03556.  {
03557.     DUONGTHANG ln;
03558.     Nhap(ln);
03559.
03560.     cout << "\nDuong thangvua nhap:";
03561.     Xuat(ln);
03562.     return 1;
03563.  }
03564.
03565.  void Nhap(DUONGTHANG &l)
03566.  {
03567.     cout << "Nhap a: ";
03568.     cin >> l.a;
03569.     cout << "Nhap b: ";
03570.     cin >> l.b;
03571.     cout << "Nhap c: ";
03572.     cin >> l.c;
03573.  }
03574.
03575.  void Xuat(DUONGTHANG l)
03576.  {
03577.     cout << setw(6);
03578.     cout << setprecision(3);
03579.     cout << "\na: " << l.a;
03580.     cout << "\nb: " << l.b;
03581.     cout << "\nc: " << l.c;
03582.  }
```

01.07.13 Đa thức – Polynomial

Bài cơ sở 022. Hãy khai báo kiểu dữ liệu để biểu diễn khái niệm đa thức một biến trong toán học: $f(x) = a_n x^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_2 x^2 + a_1 x + a_0$ và định nghĩa hàm nhập và hàm xuất cho kiểu dữ liệu này.

– Khai báo kiểu dữ liệu biểu diễn đa thức.

```
03583.  struct DaThuc
03584.  {
03585.     float a[100];
03586.     int n;
```

```
03587.  };
03588.  typedef struct DaThuc DATHUC;
```

– Định nghĩa hàm nhập đa thức.

```
03589.  void Nhap(DATHUC &f)
03590.  {
03591.     cout << "Nhap bac da thuc: ";
03592.     cin >> f.n;
03593.     for(int i=f.n;i>=0;i--)
03594.     {
03595.        cout << "Nhap he so a[" << i << "]: ";
03596.        cin >> f.a[i];
03597.     }
03598.  }
```

– Định nghĩa hàm xuất đa thức.

```
03599.  void Xuat(DATHUC f)
03600.  {
03601.     cout << setw(6);
03602.     cout << setprecision(3);
03603.     for(int i=f.n;i>=1;i--)
03604.     {
03605.        cout << f.a[i];
03606.        cout << "x^" << i;
03607.     }
03608.     cout << f.a[0];
03609.  }
```

Bài 013.Hãy viết chương trình nhập và xuất thông tin của một đa thức (DATHUC).

– Chương trình.

```
03610.  #include <iostream>
03611.  #include <iomanip>
03612.  using namespace std;
03613.
03614.  struct DaThuc
03615.  {
03616.     float a[100];
03617.     int n;
03618.  };
03619.  typedef struct DaThuc DATHUC;
03620.
03621.  void Nhap(DATHUC&);
03622.  void Xuat(DATHUC);
```

```
03623.
03624.  int main()
03625.  {
03626.     DATHUC ff;
03627.     Nhap(ff);
03628.
03629.     cout << "\nDa thuc vua nhap: ";
03630.     Xuat(ff);
03631.     return 1;
03632.  }
03633.
03634.  void Nhap(DATHUC& f)
03635.  {
03636.     cout << "Nhap bac da thuc: ";
03637.     cin >> f.n;
03638.     for (int i = f.n; i >= 0; i--)
03639.     {
03640.        cout << "Nhap he so a[" << i << "]: ";
03641.        cin >> f.a[i];
03642.     }
03643.  }
03644.
03645.  void Xuat(DATHUC f)
03646.  {
03647.     for (int i = f.n; i >= 1; i--)
03648.     {
03649.        cout << f.a[i];
03650.        cout << "x^" << i;
03651.        cout << setw(3);
03652.     }
03653.     cout << f.a[0];
03654.  }
```

01.08 CÁC CHỦ ĐỀ PHỔ THÔNG

01.08.01 Chủ đề điểm trong mặt phẳng Oxy – Point

01.08.01.1 Chương trình minh họa chủ đề điểm

Chương trình 001. Viết chương trình thực hiện các yêu cầu sau:
a. Nhập tọa độ hai điểm A, B trong mặt phẳng Oxy.
b. Tìm tọa độ điểm đối xứng với điểm A qua gốc tọa độ.
c. Tính khoảng cách giữa hai điểm A và B.

– Chương trình

```
03655.  #include <iostream>
03656.  #include <iomanip>
03657.  #include <cmath>
03658.  using namespace std;
03659.
03660.  struct diem
03661.  {
03662.      float x;
03663.      float y;
03664.  };
03665.  typedef struct diem DIEM;
03666.
03667.  void Nhap(DIEM&);
03668.  void Xuat(DIEM);
03669.
03670.  DIEM DoiXungGoc(DIEM);
03671.  float KhoangCach(DIEM, DIEM);
03672.
03673.  int main()
03674.  {
03675.      DIEM A, B;
03676.      cout << "Nhap toa do diem 1: ";
03677.      Nhap(A);
03678.      cout << "Nhap toa do diem 2: ";
03679.      Nhap(B);
03680.      cout << "\nToa do diem 1: ";
03681.      Xuat(A);
03682.      cout << "\nToa do diem 2: ";
03683.      Xuat(B);
03684.
03685.      DIEM C;
03686.      C = DoiXungGoc(A);
03687.      cout << "\nDiem doi xung A qua goc toa do:";
03688.      Xuat(C);
03689.
03690.      float kq = KhoangCach(A, B);
03691.      cout << setw(6);
03692.      cout << setprecision(3);
03693.      cout << "\nKhoang cach hai diem: " << kq;
03694.      return 1;
03695.  }
```

```
03696.
03697.  void Nhap(DIEM &P)
03698.  {
03699.     cout << "\nNhap x: ";
03700.     cin >> P.x;
03701.     cout << "Nhap y: ";
03702.     cin >> P.y;
03703.  }
03704.
03705.  void Xuat(DIEM P)
03706.  {
03707.     cout << setw(6);
03708.     cout << setprecision(3);
03709.     cout << "\nx = " << P.x;
03710.     cout << "\ny = " << P.y;
03711.  }
03712.
03713.  DIEM DoiXungGoc(DIEM P)
03714.  {
03715.     DIEM temp;
03716.     temp.x = -P.x;
03717.     temp.y = -P.y;
03718.     return temp;
03719.  }
03720.
03721.  float KhoangCach(DIEM P, DIEM Q)
03722.  {
03723.     return sqrt(   (Q.x - P.x)*(Q.x - P.x) +
03724.                    (Q.y - P.y)*(Q.y - P.y));
03725.  }
```

01.08.01.2 Bài tập chủ đề điểm

Bài 014.Khai báo kiểu dữ liệu biểu diễn điểm trong mặt phẳng Oxy.

- Kiến thức phổ thông: điểm trong mặt phẳng Oxy có hai thành phần:
  - + Hoành độ: x, kiểu số thực.
  - + Tung độ: y, kiểu số thực.

```
03726.  struct diem
03727.  {
03728.     float x;
03729.     float y;
03730.  };
```

```
03731. typedef struct diem DIEM;
```

Bài 015.Định nghĩa hàm nhập tọa độ điểm trong mặt phẳng Oxy.

- Khai báo hàm.

```
03732. void Nhap(DIEM &);
```

- Định nghĩa hàm.

```
03733. void Nhap(DIEM &P)
03734. {
03735.    cout << "Nhap x: ";
03736.    cin >> P.x;
03737.    cout << "Nhap y: ";
03738.    cin >> P.y;
03739. }
```

Bài 016.Định nghĩa hàm xuất tọa độ điểm.

- Khai báo hàm

```
03740. void Xuat(DIEM);
```

- Định nghĩa hàm.

```
03741. void Xuat(DIEM P)
03742. {
03743.    cout << "x: " << P.x << endl;
03744.    cout << "y: " << P.y << endl;
03745. }
```

Bài 017.Tính khoảng cách giữa hai điểm.

- Kiến thức phổ thông: công thức tính khoảng cách giữa hai điểm $P(P.x, P.y)$ và $Q(Q.x, Q.y)$ là:
- $Khoảng\ cách\ (P,Q) = \sqrt{(Q.x - P.x)^2 + (Q.y - P.y)^2}$.
- Khai báo hàm

```
03746. float KhoangCach(DIEM,DIEM);
```

- Định nghĩa hàm.

```
03747. float KhoangCach(DIEM P, DIEM Q)
03748. {
03749.    return sqrt((P.x - Q.x) * (P.x - Q.x) +
03750.                (P.y - Q.y) * (P.y - Q.y));
03751. }
```

Bài 018.Tính khoảng cách giữa hai điểm theo phương Ox.

- Kiến thức phổ thông: công thức tính khoảng cách giữa hai điểm $P(P.x, P.y)$ và $Q(Q.x, Q.y)$ là: $Khoảng\ cách\ x\ (P,Q) = |Q.x - P.x|$.
- Khai báo hàm

```
03752. float KhoangCachX(DIEM,DIEM);
```

- Định nghĩa hàm.

```
03753. float KhoangCachX(DIEM P, DIEM Q)
03754. {
03755.     return abs(Q.x - P.x);
03756. }
```

Bài 019.Tính khoảng cách giữa hai điểm theo phương Oy.

- Kiến thức phổ thông: công thức tính khoảng cách giữa hai điểm $P(P.x, P.y)$ và $Q(Q.x, Q.y)$ là: $Khoảng\ cách\ y\ (P,Q) = |Q.y - P.y|$.
- Khai báo hàm

```
03757. float KhoangCachY(DIEM,DIEM);
```

- Định nghĩa hàm.

```
03758. float KhoangCachY(DIEM P, DIEM Q)
03759. {
03760.     return abs(Q.y - P.y);
03761. }
```

Bài 020.Tính khoảng cách đến gốc tọa độ.

- Khai báo hàm

```
03762. float KhoangCachGoc(DIEM);
```

- Kiến thức phổ thông: công thức tính khoảng cách từ một điểm $P(P.x, P.y)$ đến gốc tọa độ là: $Khoảng\ cách\ gốc\ (P) = \sqrt{(P.x)^2 + (P.y)^2}$.
- Định nghĩa hàm.

```
03763. float KhoangCachGoc(DIEM P)
03764. {
03765.     return sqrt(P.x*P.x + P.y*P.y);
03766. }
```

Bài 021.Tìm tọa độ điểm đối xứng qua gốc tọa độ.

- Khai báo hàm

```
03767. DIEM DoiXungGoc(DIEM);
```

- Định nghĩa hàm.

```
03768. DIEM DoiXungGoc(DIEM P)
```

```
03769. {
03770.     DIEM temp;
03771.     temp.x = -P.x;
03772.     temp.y = -P.y;
03773.     return temp;
03774. }
```

Bài 022.Tìm tọa độ điểm đối xứng qua trục hoành.

– Khai báo hàm

```
03775. DIEM DoiXungHoanh(DIEM);
```

– Định nghĩa hàm.

```
03776. DIEM DoiXungHoanh(DIEM P)
03777. {
03778.     DIEM temp;
03779.     temp.x = P.x;
03780.     temp.y = -P.y;
03781.     return temp;
03782. }
```

Bài 023.Tìm tọa độ điểm đối xứng qua trục tung.

– Khai báo hàm

```
03783. DIEM DoiXungTung(DIEM);
```

– Định nghĩa hàm.

```
03784. DIEM DoiXungTung(DIEM P)
03785. {
03786.     DIEM temp;
03787.     temp.x = -P.x;
03788.     temp.y = P.y;
03789.     return temp;
03790. }
```

Bài 024.Tìm tọa độ điểm đối xứng qua đường phân giác thứ nhất $y = x$.

– Khai báo hàm

```
03791. DIEM DoiXungPhanGiac1(DIEM);
```

– Định nghĩa hàm.

```
03792. DIEM DoiXungPhanGiac1(DIEM P)
03793. {
03794.     DIEM temp;
03795.     temp.x = P.y;
03796.     temp.y = P.x;
```

```
03797.     return temp;
03798.  }
```

Bài 025.Tìm tọa độ điểm đối xứng qua đường phân giác thứ hai $y = -x$.

– Khai báo hàm

```
03799.  DIEM DoiXungPhanGiac2(DIEM);
```

– Định nghĩa hàm.

```
03800.  DIEM DoiXungPhanGiac2(DIEM P)
03801.  {
03802.     DIEM temp;
03803.     temp.x = -P.y;
03804.     temp.y = -P.x;
03805.     return temp;
03806.  }
```

Bài 026.Định nghĩa hàm kiểm tra hai điểm có trùng nhau nay không?

– Khai báo hàm.

```
03807.  int ktTrung (DIEM,DIEM);
```

– Định nghĩa hàm.

```
03808.  int ktTrung(DIEM P,DIEM Q)
03809.  {
03810.     if(P.x==Q.x && P.y==Q.y)
03811.        return 1;
03812.     return 0;
03813.  }
```

Bài 027.Kiểm tra điểm có thuộc phần tư thứ I không?

– Khai báo hàm

```
03814.  int ktThuoc1(DIEM);
```

– Định nghĩa hàm.

```
03815.  int ktThuoc1(DIEM P)
03816.  {
03817.     if (P.x > 0 && P.y > 0)
03818.        return 1;
03819.     return 0;
03820.  }
```

Bài 028.Kiểm tra điểm có thuộc phần tư thứ II không?

– Khai báo hàm

```
03821. int ktThuoc2(DIEM);
```

– Định nghĩa hàm.

```
03822. int ktThuoc2(DIEM P)
03823. {
03824.    if (P.x < 0 && P.y > 0)
03825.       return 1;
03826.    return 0;
03827. }
```

Bài 029.Kiểm tra điểm có thuộc phần tư thứ III không?

– Khai báo hàm

```
03828. int ktThuoc3(DIEM);
```

– Định nghĩa hàm.

```
03829. int ktThuoc3(DIEM P)
03830. {
03831.    if (P.x < 0 && P.y < 0)
03832.       return 1;
03833.    return 0;
03834. }
```

Bài 030.Kiểm tra điểm có thuộc phần tư thứ IV không?

– Khai báo hàm

```
03835. int ktThuoc4(DIEM);
```

– Định nghĩa hàm.

```
03836. int ktThuoc4(DIEM P)
03837. {
03838.    if (P.x > 0 && P.y < 0)
03839.       return 1;
03840.    return 0;
03841. }
```

01.08.02 Chủ đề điểm trong không gian Oxyz

01.08.02.1 Chương trình minh họa chủ đề điểm trong không gian

Chương trình 002. Viết chương trình thực hiện các yêu cầu sau:
a. Nhập tọa độ hai điểm A, B trong không gian Oxyz.
b. Tìm tọa độ điểm đối xứng với điểm A qua mặt phẳng Oxy.
c. Tính khoảng cách giữa hai điểm A và B theo phương x.

– Chương trình

```
03842.  #include <iostream>
03843.  #include <iomanip>
03844.  #include <cmath>
03845.  using namespace std;
03846.
03847.  struct diemkhonggian
03848.  {
03849.     float x;
03850.     float y;
03851.     float z;
03852.  };
03853.  typedef struct diemkhonggian DIEMKHONGGIAN;
03854.
03855.  void Nhap(DIEMKHONGGIAN&);
03856.  void Xuat(DIEMKHONGGIAN);
03857.
03858.  DIEMKHONGGIAN DoiXungOxy(DIEMKHONGGIAN);
03859.  float KhoangCachX(DIEMKHONGGIAN,DIEMKHONGGIAN);
03860.
03861.  int main()
03862.  {
03863.     DIEMKHONGGIAN A, B;
03864.     cout << "Nhap toa do diem 1: ";
03865.     Nhap(A);
03866.     cout << "Nhap toa do diem 2: ";
03867.     Nhap(B);
03868.     cout << "\nToa do diem 1: ";
03869.     Xuat(A);
03870.     cout << "\nToa do diem 2: ";
03871.     Xuat(B);
03872.
03873.     DIEMKHONGGIAN C;
03874.     C = DoiXungOxy(A);
03875.     cout << "\nDiem doi xung A tren Oxy: ";
03876.     Xuat(C);
03877.
03878.     float kq = KhoangCachX(A, B);
03879.     cout << setw(6);
03880.     cout << setprecision(3);
03881.     cout << "\nKhoang cach x :" << kq;
03882.     return 1;
03883.  }
```

```
03884.
03885.  void Nhap(DIEMKHONGGIAN &P)
03886.  {
03887.     cout << "\nNhap x: ";
03888.     cin >> P.x;
03889.     cout << "Nhap y: ";
03890.     cin >> P.y;
03891.     cout << "Nhap z: ";
03892.     cin >> P.z;
03893.  }
03894.
03895.  void Xuat(DIEMKHONGGIAN P)
03896.  {
03897.     cout << setw(6);
03898.     cout << setprecision(3);
03899.     cout << "\nx = " << P.x;
03900.     cout << "\ny = " << P.y;
03901.     cout << "\ny = " << P.z;
03902.  }
03903.
03904.  DIEMKHONGGIAN DoiXungOxy(DIEMKHONGGIAN P)
03905.  {
03906.     DIEMKHONGGIAN temp;
03907.     temp.x = -P.x;
03908.     temp.y = -P.y;
03909.     temp.z = +P.z;
03910.     return temp;
03911.  }
03912.
03913.  float KhoangCachX(DIEMKHONGGIAN P,
03914.                    DIEMKHONGGIAN Q)
03915.  {
03916.     return abs(Q.x-P.x);
03917.  }
03918.
```

01.08.02.2 Bài tập chủ đề điểm trong không gian

Bài 031.Khai báo kiểu dữ liệu biểu diễn điểm trong không gian $Oxyz$.

– Khai báo kiểu dữ liệu.

```
03919.  struct diemkhonggian
03920.  {
03921.     float x;
```

```
03922.    float y;
03923.    float z;
03924. };
03925. typedef struct diemkhonggian DIEMKHONGGIAN;
```

Bài 032.Định nghĩa hàm nhập tọa độ điểm trong không gian $Oxyz$.

– Khai báo hàm

```
03926. void Nhap(DIEMKHONGGIAN&);
```

– Định nghĩa hàm.

```
03927. void Nhap(DIEMKHONGGIAN& P)
03928. {
03929.    cout << "Nhap x: ";
03930.    cin >> P.x;
03931.    cout << "Nhap y: ";
03932.    cin >> P.y;
03933.    cout << "Nhap z: ";
03934.    cin >> P.z;
03935. }
```

Bài 033.Định nghĩa hàm xuất tọa độ điểm không gian $Oxyz$.

– Khai báo hàm

```
03936. void Xuat(DIEMKHONGGIAN);
```

– Định nghĩa hàm.

```
03937. void Xuat(DIEMKHONGGIAN P)
03938. {
03939.    cout << "\nx: " << P.x;
03940.    cout << "\ny: " << P.y;
03941.    cout << "\nz: " << P.z;
03942. }
```

Bài 034.Tính khoảng cách giữa hai điểm trong không gian.

– Khai báo hàm

```
03943. float KhoangCach(DIEMKHONGGIAN,
03944.                  DIEMKHONGGIAN);
```

– Định nghĩa hàm.

```
03945. float KhoangCach(DIEMKHONGGIAN P,
03946.                  DIEMKHONGGIAN Q)
03947. {
03948.    return sqrt((P.x - Q.x) * (P.x - Q.x) +
03949.                (P.y - Q.y) * (P.y - Q.y) +
```

```
03950.                  (P.z - Q.z) * (P.z - Q.z));
03951. }
```

Bài 035.Tính khoảng cách giữa hai điểm trong không gian theo phương x.

– Khai báo hàm

```
03952. float KhoangCachX(DIEMKHONGGIAN,
03953.                   DIEMKHONGGIAN);
```

– Định nghĩa hàm.

```
03954. float KhoangCachX(DIEMKHONGGIAN P,
03955.                   DIEMKHONGGIAN Q)
03956. {
03957.    return abs(Q.x - P.x);
03958. }
```

Bài 036.Tính khoảng cách giữa hai điểm trong không gian theo phương y.

– Khai báo hàm

```
03959. float KhoangCachY(DIEMKHONGGIAN,
03960.                   DIEMKHONGGIAN);
```

– Định nghĩa hàm.

```
03961. float KhoangCachY(DIEMKHONGGIAN P,
03962.                   DIEMKHONGGIAN Q)
03963. {
03964.    return abs(Q.y - P.y);
03965. }
```

Bài 037.Tính khoảng cách giữa hai điểm trong không gian theo phương z.

– Khai báo hàm

```
03966. float KhoangCachZ(DIEMKHONGGIAN,
03967.                   DIEMKHONGGIAN);
```

– Định nghĩa hàm.

```
03968. float KhoangCachZ(DIEMKHONGGIAN P,
03969.                   DIEMKHONGGIAN Q)
03970. {
03971.    return abs(Q.z - P.z);
03972. }
```

Bài 038.Tính khoảng cách đến gốc tọa độ.

– Kiến thức phổ thông: công thức tính khoảng cách từ một điểm không gian $P(P.x, P.y, P.z)$ đến gốc tọa độ là: $Khoảng\ cách\ gốc\ (P) = \sqrt{(P.x)^2 + (P.y)^2 + (P.z)^2}$.

– Khai báo hàm

```
03973. float KhoangCachGoc(DIEMKHONGGIAN);
```

– Định nghĩa hàm.

```
03974. float KhoangCachGoc(DIEMKHONGGIAN P)
03975. {
03976.    return sqrt(P.x*P.x + P.y*P.y + P.z*P.z);
03977. }
```

Bài 039.Tìm tọa độ điểm đối xứng qua gốc tọa độ.

– Khai báo hàm

```
03978. DIEMKHONGGIAN DoiXungGoc(DIEMKHONGGIAN);
```

– Định nghĩa hàm.

```
03979. DIEMKHONGGIAN DoiXungGoc(DIEMKHONGGIAN P)
03980. {
03981.    DIEMKHONGGIAN temp;
03982.    temp.x = -P.x;
03983.    temp.y = -P.y;
03984.    temp.z = -P.z;
03985.    return temp;
03986. }
```

Bài 040.Tìm tọa độ điểm đối xứng qua mặt phẳng Oxy.

– Khai báo hàm

```
03987. DIEMKHONGGIAN DoiXungOxy(DIEMKHONGGIAN);
```

– Định nghĩa hàm.

```
03988. DIEMKHONGGIAN DoiXungOxy(DIEMKHONGGIAN P)
03989. {
03990.    DIEMKHONGGIAN temp;
03991.    temp.x = P.x;
03992.    temp.y = P.y;
03993.    temp.z = -P.z;
03994.    return temp;
03995. }
```

Bài 041.Tìm tọa độ điểm đối xứng qua mặt phẳng Oxz.

– Khai báo hàm

```
03996. DIEMKHONGGIAN DoiXungOxz(DIEMKHONGGIAN);
```

– Định nghĩa hàm.

```
03997. DIEMKHONGGIAN DoiXungOxz(DIEMKHONGGIAN P)
03998. {
03999.    DIEMKHONGGIAN temp;
04000.    temp.x = P.x;
04001.    temp.y = -P.y;
04002.    temp.z = P.z;
04003.    return temp;
04004. }
```

Bài 042.Tìm tọa độ điểm đối xứng qua mặt phẳng *Oyz*.

– Khai báo hàm

```
04005. DIEMKHONGGIAN DoiXungOyz(DIEMKHONGGIAN);
```

– Định nghĩa hàm.

```
04006. DIEMKHONGGIAN DoiXungOyz(DIEMKHONGGIAN P)
04007. {
04008.    DIEMKHONGGIAN temp;
04009.    temp.x = -P.x;
04010.    temp.y = P.y;
04011.    temp.z = P.z;
04012.    return temp;
04013. }
```

Bài 043.Định nghĩa hàm kiểm tra hai điểm có trùng nhau nay không?

– Khai báo hàm.

```
04014. int ktTrung(DIEMKHONGGIAN,DIEMKHONGGIAN);
```

– Định nghĩa hàm.

```
04015. int ktTrung(DIEMKHONGGIAN P,DIEMKHONGGIAN Q)
04016. {
04017.    if(P.x==Q.x && P.y==Q.y && P.z==Q.z)
04018.       return 1;
04019.    return 0;
04020. }
```

01.08.03 Chủ đề phân số – Fraction

01.08.03.1 Chương trình minh họa chủ đề phân số

Chương trình 003. Viết chương trình thực hiện các yêu cầu sau:
a. Nhập hai phân số.

b. Kiểm tra phân số thứ nhất có phải là phân số dương hay không?
c. Tính tổng hai phân số.

– Chương trình

```
04021. #include <iostream>
04022. #include <cmath>
04023. using namespace std;
04024.
04025. struct phanso
04026. {
04027.    int Tu;
04028.    int Mau;
04029. };
04030. typedef struct phanso PHANSO;
04031.
04032. void Nhap(PHANSO&);
04033. void Xuat(PHANSO);
04034.
04035. int ktDuong(PHANSO);
04036. PHANSO Tong(PHANSO,PHANSO);
04037.
04038. int main()
04039. {
04040.    PHANSO A, B;
04041.    cout << "Nhap phan so 1:";
04042.    Nhap(A);
04043.    cout << "Nhap phan so 2:";
04044.    Nhap(B);
04045.    cout << "Phan so 1:";
04046.    Xuat(A);
04047.    cout << "Phan so 2:";
04048.    Xuat(B);
04049.
04050.    if(ktDuong(A))
04051.       cout << "\nPhan so 1 duong.";
04052.    else
04053.       cout << "\nPhan so 1 ko duong.";
04054.
04055.    PHANSO kq = Tong(A, B);
04056.    cout << "\nTong la: ";
04057.    Xuat(kq);
04058.    return 1;
04059. }
```

```
04060.
04061.  void Nhap(PHANSO& x)
04062.  {
04063.     cout << "\nNhap tu: ";
04064.     cin >> x.Tu;
04065.     cout << "Nhap mau: ";
04066.     cin >> x.Mau;
04067.  }
04068.
04069.  void Xuat(PHANSO x)
04070.  {
04071.     cout << "\nTu =  "<<x.Tu;
04072.     cout << "\nMau = "<<x.Mau;
04073.  }
04074.
04075.  int ktDuong(PHANSO x)
04076.  {
04077.     if(x.Tu*x.Mau > 0)
04078.        return 1;
04079.     return 0;
04080.  }
04081.
04082.  PHANSO Tong(PHANSO x, PHANSO y)
04083.  {
04084.     PHANSO temp;
04085.     temp.Tu = x.Tu * y.Mau + x.Mau * y.Tu;
04086.     temp.Mau = x.Mau * y.Mau;
04087.     return temp;
04088.  }
```

01.08.03.2 Bài tập chủ đề phân số

Bài 044.Khai báo kiểu dữ liệu biểu diễn phân số.

– Khai báo kiểu dữ liệu.

```
04089.  struct phanso
04090.  {
04091.     int Tu;
04092.     int Mau;
04093.  };
04094.  typedef struct phanso PHANSO;
```

Bài 045.Định nghĩa hàm nhập phân số.

– Khai báo hàm

```
04095.  void Nhap(PHANSO &);
```

– Định nghĩa hàm.

```
04096.  void Nhap(PHANSO &x)
04097.  {
04098.     cout << "Nhap tu: ";
04099.     cin >> x.Tu;
04100.     cout << "Nhap mau: ";
04101.     cin >> x.Mau;
04102.  }
```

Bài 046.Định nghĩa hàm xuất phân số.

– Khai báo hàm

```
04103.  void Xuat(PHANSO);
```

– Định nghĩa hàm.

```
04104.  void Xuat(PHANSO x)
04105.  {
04106.     cout << "\nTu: " << x.Tu;
04107.     cout << "\nMau: " << x.Mau;
04108.  }
```

Bài 047.Rút gọn phân số.

– Kiến thức phổ thông: để rút gọn một phân số ta thực hiện theo ba bước như sau:
  + Tìm ước chung lớn nhất của tử và mẫu.
  + Tử chia cho ước chung lớn nhất.
  + Mẫu chia cho ước chung lớn nhất.
– Ví dụ:
  + Dữ liệu vào: $\frac{4}{8}$
  + Dữ liệu ra: $\frac{1}{2}$
– Khai báo hàm

```
04109.  void RutGon(PHANSO &);
```

– Định nghĩa hàm.

```
04110.  int UCLN(int a, int b)
04111.  {
04112.     a = abs(a);
04113.     b = abs(b);
04114.     while (a * b != 0)
04115.        if (a > b)
04116.           a = a - b;
```

```
04117.       else
04118.          b = b - a;
04119.    return a + b;
04120. }
04121.
04122. void RutGon(PHANSO& x)
04123. {
04124.    int kq = UCLN(x.Tu, x.Mau);
04125.    x.Tu = x.Tu / kq;
04126.    x.Mau = x.Mau / kq;
04127. }
```

Bài 048.Tính tổng hai phân số.

- Kiến thức phổ thông
  - Gọi phân số thứ nhất là x có dạng: $\frac{a}{b}$.
  - Gọi phân số thứ hai là y có dạng: $\frac{c}{d}$.
  - Tổng hai phân số là $\frac{a}{b}+\frac{c}{d}=\frac{ad+bc}{bd}$.
- Ví dụ:
  - Dữ liệu vào: $\frac{1}{2}+\frac{1}{4}$
  - Dữ liệu ra: $\frac{3}{4}$
- Khai báo hàm

```
04128. PHANSO Tong(PHANSO,PHANSO);
```

- Định nghĩa hàm.

```
04129. PHANSO Tong(PHANSO x, PHANSO y)
04130. {
04131.    PHANSO temp;
04132.    temp.Tu = x.Tu * y.Mau + y.Tu * x.Mau;
04133.    temp.Mau = x.Mau * y.Mau;
04134.    RutGon(temp);
04135.    return temp;
04136. }
```

Bài 049.Tính hiệu hai phân số.

- Kiến thức phổ thông
  - Gọi phân số thứ nhất là x có dạng: $\frac{a}{b}$.
  - Gọi phân số thứ hai là y có dạng: $\frac{c}{d}$.
  - Hiệu hai phân số là $\frac{a}{b}-\frac{c}{d}=\frac{ad-bc}{bd}$.
- Ví dụ:

+ Dữ liệu vào: $\frac{1}{2} - \frac{1}{4}$.
+ Dữ liệu ra: $\frac{1}{4}$.

– Khai báo hàm

```
04137.  PHANSO Hieu(PHANSO,PHANSO);
```

– Định nghĩa hàm.

```
04138.  PHANSO Hieu(PHANSO x, PHANSO y)
04139.  {
04140.      PHANSO temp;
04141.      temp.Tu = x.Tu * y.Mau - y.Tu * x.Mau;
04142.      temp.Mau = x.Mau * y.Mau;
04143.      RutGon(temp);
04144.      return temp;
04145.  }
```

Bài 050.Tính tích hai phân số.

– Kiến thức phổ thông
+ Gọi phân số thứ nhất là x có dạng: $\frac{a}{b}$.
+ Gọi phân số thứ hai là y có dạng: $\frac{c}{d}$.
+ Tích hai phân số là $\frac{a}{b} \times \frac{c}{d} = \frac{ac}{bd}$.

– Ví dụ:
+ Dữ liệu vào: $\frac{6}{7} \times \frac{9}{8}$.
+ Dữ liệu ra: $\frac{27}{28}$.

– Khai báo hàm

```
04146.  PHANSO Tich(PHANSO,PHANSO);
```

– Định nghĩa hàm.

```
04147.  PHANSO Tich(PHANSO x, PHANSO y)
04148.  {
04149.      PHANSO temp;
04150.      temp.Tu = x.Tu * y.Tu;
04151.      temp.Mau = x.Mau * y.Mau;
04152.      RutGon(temp);
04153.      return temp;
04154.  }
```

Bài 051.Tính thương hai phân số.

– Kiến thức phổ thông
+ Gọi phân số thứ nhất là x có dạng: $\frac{a}{b}$.

+ Gọi phân số thứ hai là y có dạng: $\frac{c}{d}$.
+ Thương hai phân số là $\frac{a}{b} \div \frac{c}{d} = \frac{a}{b} \times \frac{d}{c} = \frac{ad}{bc}$.

– Ví dụ:
+ Dữ liệu vào: $\frac{6}{7} \div \frac{9}{8}$.
+ Dữ liệu ra: $\frac{16}{21}$.

– Khai báo hàm

```
04155. PHANSO Thuong(PHANSO,PHANSO);
```

– Định nghĩa hàm.

```
04156. PHANSO Thuong(PHANSO x, PHANSO y)
04157. {
04158.    PHANSO temp;
04159.    temp.Tu = x.Tu * y.Mau;
04160.    temp.Mau = x.Mau * y.Tu;
04161.    RutGon(temp);
04162.    return temp;
04163. }
```

Bài 052.Kiểm tra phân số có nghĩa

– Kiến thức phổ thông: một phân số được gọi là có nghĩa khi mẫu khác không.

– Khai báo hàm

```
04164. int ktCoNghia(PHANSO);
```

– Định nghĩa hàm.

```
04165. int ktCoNghia(PHANSO x)
04166. {
04167.    if (x.Mau != 0)
04168.       return 1;
04169.    return 0;
04170. }
```

Bài 053.Kiểm tra phân số tối giản.

– Kiến thức phổ thông: một phân số được gọi là tối giản khi ước chung lớn nhất của tử và mẫu là 1.

– Khai báo hàm

```
04171. int ktToiGian(PHANSO);
```

– Định nghĩa hàm.

```
04172. int ktToiGian(PHANSO x)
04173. {
04174.    if (UCLN(x.Tu, x.Mau) == 1)
```

```
04175.       return 1;
04176.    return 0;
04177. }
```

Bài 054.Kiểm tra phân số dương.

- Kiến thức phổ thông: một phân số được gọi là phân số dương khi tử và mẫu cùng dấu.
- Khai báo hàm

```
04178. int ktDuong(PHANSO);
```

- Định nghĩa hàm.

```
04179. int ktDuong(PHANSO x)
04180. {
04181.    if (x.Tu * x.Mau > 0)
04182.       return 1;
04183.    return 0;
04184. }
```

Bài 055.Kiểm tra phân số âm.

- Kiến thức phổ thông: một phân số được gọi là phân số âm khi tử và mẫu trái dấu.
- Khai báo hàm

```
04185. int ktAm(PHANSO);
```

- Định nghĩa hàm.

```
04186. int ktAm(PHANSO x)
04187. {
04188.    if (x.Tu * x.Mau < 0)
04189.       return 1;
04190.    return 0;
04191. }
```

Bài 056.Qui đồng mẫu hai phân số.

- Kiến thức phổ thông
  - + Gọi phân số thứ nhất là x có dạng: $\frac{a}{b}$.
  - + Gọi phân số thứ hai là y có dạng: $\frac{c}{d}$.
  - + Qui đồng tử hai phân số là:
    - x sẽ là: $\frac{ad}{bd}$.
    - y sẽ là: $\frac{cb}{bd}$.
- Khai báo hàm

```
04192. void QuiDongMau(PHANSO&,PHANSO&);
```

– Định nghĩa hàm.

```
04193. void QuiDongMau(PHANSO &x, PHANSO &y)
04194. {
04195.    int mc = x.Mau * y.Mau;
04196.    x.Tu = x.Tu * y.Mau;
04197.    y.Tu = x.Mau * y.Tu;
04198.    x.Mau = mc;
04199.    y.Mau = mc;
04200. }
```

Bài 057.Qui đồng tử hai phân số.

– Kiến thức phổ thông
  + Gọi phân số thứ nhất là x có dạng: $\frac{a}{b}$.
  + Gọi phân số thứ hai là y có dạng: $\frac{c}{d}$.
  + Qui đồng tử hai phân số là:
    - x sẽ là: $\frac{ac}{bc}$.
    - y sẽ là: $\frac{ac}{da}$.

– Khai báo hàm

```
04201. void QuiDongTu(PHANSO&,PHANSO&);
```

– Định nghĩa hàm.

```
04202. void QuiDongTu(PHANSO &x, PHANSO &y)
04203. {
04204.    int tc = x.Tu * y.Tu;
04205.    x.Mau = x.Mau * y.Tu;
04206.    y.Mau = y.Mau * x.Tu;
04207.    x.Tu = tc;
04208.    y.Tu = tc;
04209. }
```

Bài 058.Định nghĩa hàm so sánh hai phân số.

– Hàm trả về một trong ba giá trị.
  + Giá trị 1: Phân số thứ 1 lớn hơn phân số thứ 2.
  + Giá trị 0: Phân số thứ 1 bằng phân số thứ 2.
  + Giá trị -1: Phân số thứ 1 nhỏ hơn phân số thứ 2.

– Khai báo hàm

```
04210. int SoSanh(PHANSO,PHANSO);
```

– Định nghĩa hàm.

```
04211. int SoSanh(PHANSO x, PHANSO y)
04212. {
```

```
04213.     float a = (float)x.Tu / x.Mau;
04214.     float b = (float)y.Tu / y.Mau;
04215.     if (a > b)
04216.        return 1;
04217.     if (a < b)
04218.        return -1;
04219.     return 0;
04220.  }
```

Bài 059.Định nghĩa toán tử cộng (`operator +`) cho hai phân số.

– Khai báo hàm

```
04221.  PHANSO operator+(PHANSO,PHANSO);
```

– Định nghĩa hàm.

```
04222.  PHANSO operator+(PHANSO x, PHANSO y)
04223.  {
04224.     PHANSO temp;
04225.     temp.Tu = x.Tu * y.Mau + y.Tu * x.Mau;
04226.     temp.Mau = x.Mau * y.Mau;
04227.     RutGon(temp);
04228.     return temp;
04229.  }
```

Bài 060.Định nghĩa toán tử hiệu (`operator -`) cho hai phân số.

– Khai báo hàm

```
04230.  PHANSO operator-(PHANSO,PHANSO);
```

– Định nghĩa hàm.

```
04231.  PHANSO operator-(PHANSO x, PHANSO y)
04232.  {
04233.     PHANSO temp;
04234.     temp.Tu = x.Tu * y.Mau - y.Tu * x.Mau;
04235.     temp.Mau = x.Mau * y.Mau;
04236.     RutGon(temp);
04237.     return temp;
04238.  }
```

Bài 061.Định nghĩa toán tử tích (`operator *`) cho hai phân số.

– Khai báo hàm

```
04239.  PHANSO operator*(PHANSO,PHANSO);
```

– Định nghĩa hàm.

```
04240.  PHANSO operator*(PHANSO x, PHANSO y)
```

```
04241. {
04242.     PHANSO temp;
04243.     temp.Tu = x.Tu * y.Tu;
04244.     temp.Mau = x.Mau * y.Mau;
04245.     RutGon(temp);
04246.     return temp;
04247. }
```

Bài 062.Định nghĩa toán tử thương (`operator /`) cho hai phân số.

– Khai báo hàm

```
04248. PHANSO operator/(PHANSO,PHANSO);
```

– Định nghĩa hàm.

```
04249. PHANSO operator/(PHANSO x, PHANSO y)
04250. {
04251.     PHANSO temp;
04252.     temp.Tu = x.Tu * y.Mau;
04253.     temp.Mau = x.Mau * y.Tu;
04254.     RutGon(temp);
04255.     return temp;
04256. }
```

01.08.04 Chủ đề số phức – Complex Numbers

01.08.04.1 Chương trình minh họa chủ đề số phức

Chương trình 004. Viết chương trình thực hiện các yêu cầu sau:
a. Nhập hai số phức.
b. Tính module của số phức thứ nhất.
c. Tính hiệu hai số phức.

– Chương trình

```
04257. #include <iostream>
04258. #include <iomanip>
04259. #include <cmath>
04260. using namespace std;
04261.
04262. struct sophuc
04263. {
04264.     float Thuc;
04265.     float Ao;
04266. };
04267. typedef struct sophuc SOPHUC;
```

```
04268.
04269.  void Nhap(SOPHUC&);
04270.  void Xuat(SOPHUC);
04271.
04272.  float TinhModule(SOPHUC);
04273.  SOPHUC Hieu(SOPHUC, SOPHUC);
04274.
04275.  int main()
04276.  {
04277.     SOPHUC A, B;
04278.     cout << "Nhap so phuc 1:";
04279.     Nhap(A);
04280.     cout << "Nhap so phuc 2:";
04281.     Nhap(B);
04282.     cout << "So phuc 1:";
04283.     Xuat(A);
04284.     cout << "So phuc 2:";
04285.     Xuat(B);
04286.
04287.     float k = TinhModule(A);
04288.     cout << setw(6);
04289.     cout << setprecision(3);
04290.     cout << "\nModule so phuc 1: " << k;
04291.
04292.     SOPHUC kq = Hieu(A, B);
04293.     cout << "\nHieu la: ";
04294.     Xuat(kq);
04295.     return 1;
04296.  }
04297.
04298.  void Nhap(SOPHUC& x)
04299.  {
04300.     cout << "\nNhap thuc: ";
04301.     cin >> x.Thuc;
04302.     cout << "Nhap ao: ";
04303.     cin >> x.Ao;
04304.  }
04305.
04306.  void Xuat(SOPHUC x)
04307.  {
04308.     cout << setw(6);
04309.     cout << setprecision(3);
04310.     cout << "\nThuc =  " << x.Thuc;
```

```
04311.     cout << "\nAo = " << x.Ao;
04312.  }
04313.
04314.  float TinhModule(SOPHUC x)
04315.  {
04316.     return sqrt(x.Thuc*x.Thuc + x.Ao*x.Ao);
04317.  }
04318.
04319.  SOPHUC Hieu(SOPHUC x, SOPHUC y)
04320.  {
04321.     SOPHUC temp;
04322.     temp.Thuc = x.Thuc - y.Thuc;
04323.     temp.Ao = x.Ao - y.Ao;
04324.     return temp;
04325.  }
```

01.08.04.2 Bài tập chủ đề số phức

Bài 063.Khai báo kiểu dữ liệu biểu diễn số phức.

– Khai báo kiểu dữ liệu.

```
04326.  struct sophuc
04327.  {
04328.     float Thuc;
04329.     float Ao;
04330.  };
04331.  typedef struct sophuc SOPHUC;
```

Bài 064.Định nghĩa hàm nhập số phức.

– Khai báo hàm

```
04332.  void Nhap(SOPHUC &);
```

– Định nghĩa hàm.

```
04333.  void Nhap(SOPHUC &x)
04334.  {
04335.     cout << "Nhap phan thuc: ";
04336.     cin >> x.Thuc;
04337.     cout << "Nhap phan ao: ";
04338.     cin >> x.Ao;
04339.  }
```

Bài 065.Định nghĩa hàm xuất số phức.

– Khai báo hàm

```
04340.  void Xuat(SOPHUC);
```

– Định nghĩa hàm.

```
04341. void Xuat(SOPHUC x)
04342. {
04343.     cout << "\nPhan thuc: " << x.Thuc;
04344.     cout << "\nPhan ao: " << x.Ao;
04345. }
```

Bài 066.Tính tổng hai số phức.

– Ôn tập
  + Gọi số phức thứ 1 là: x: $a + ib$.
  + Gọi số phức thứ 2 là: y: $c + id$.
  + Tổng hai số phức là:

$$\begin{array}{r} a + ib \\ \underline{c + id} \\ (a + c) + i(b + d) \end{array}$$

  + Kết quả: $(a + c) + i(b + d)$.
  + Về mặt kiểu dữ liệu thì số phức cộng số phức thì ta được số phức.

– Khai báo hàm

```
04346. SOPHUC Tong(SOPHUC,SOPHUC);
```

– Định nghĩa hàm.

```
04347. SOPHUC Tong(SOPHUC x, SOPHUC y)
04348. {
04349.     SOPHUC temp;
04350.     temp.Thuc = x.Thuc + y.Thuc;
04351.     temp.Ao = x.Ao + y.Ao;
04352.     return temp;
04353. }
```

Bài 067.Tính hiệu hai số phức.

– Ôn tập.
  + Gọi số phức thứ 1 là: x: $a + ib$.
  + Gọi số phức thứ 2 là: y: $c + id$.
  + Hiệu hai số phức là:

$$\begin{array}{r} a + ib \\ \underline{c + id} \\ (a - c) + i(b - d) \end{array}$$

  + Kết quả: $(a - c) + i(b - d)$
  + Về mặt kiểu dữ liệu thì số phức trừ số phức thì ta được số phức.

– Khai báo hàm.

```
04354.  SOPHUC Hieu(SOPHUC,SOPHUC);
```

– Định nghĩa hàm.

```
04355.  SOPHUC Hieu(SOPHUC x, SOPHUC y)
04356.  {
04357.      SOPHUC temp;
04358.      temp.Thuc = x.Thuc - y.Thuc;
04359.      temp.Ao = x.Ao - y.Ao;
04360.      return temp;
04361.  }
```

Bài 068.Tính tích hai số phức.

– Ôn tập.

+ Gọi số phức thứ 1 là: x: $a + ib$.
+ Gọi số phức thứ 2 là: y: $c + id$.
+ Tích hai số phức là:

$$(a + ib)(c + id) = a(c + id) + ib(c + id)$$
$$(a + ib)(c + id) = ac + iad + ibc + i^2bd$$
$$(a + ib)(c + id) = ac + iad + ibc - bd$$
$$(a + ib)(c + id) = ac - bd + iad + ibc$$
$$(a + ib)(c + id) = ac - bd + i(ad + bc)$$
$$(a + ib)(c + id) = (ac - bd) + i(ad + bc)$$

+ Kết quả: $(ac - bd) + i(ad + bc)$.
+ Về mặt kiểu dữ liệu thì số phức nhân số phức thì ta được số phức.

– Khai báo hàm.

```
04362.  SOPHUC Tich(SOPHUC,SOPHUC);
```

– Định nghĩa hàm.

```
04363.  SOPHUC Tich(SOPHUC x, SOPHUC y)
04364.  {
04365.      SOPHUC temp;
04366.      temp.Thuc = x.Thuc * y.Thuc - x.Ao * y.Ao;
04367.      temp.Ao = x.Thuc * y.Ao + x.Ao * y.Thuc;
04368.      return temp;
04369.  }
```

Bài 069.Tính thương hai số phức.

– Ôn tập

+ Gọi số phức thứ 1 là: x: $a + ib$.
+ Gọi số phức thứ 2 là: y: $c + id$.
+ Tích hai số phức là:

$$\frac{(a+ib)}{(c+id)} = \frac{(a+ib)(c-id)}{(c+id)(c-id)}$$
$$\frac{(a+ib)}{(c+id)} = \frac{(a+ib)(c-id)}{c^2-(id)^2}$$
$$\frac{(a+ib)}{(c+id)} = \frac{(a+ib)(c-id)}{c^2-i^2d^2}$$
$$\frac{(a+ib)}{(c+id)} = \frac{(a+ib)(c-id)}{c^2-(-1)d^2}$$
$$\frac{(a+ib)}{(c+id)} = \frac{(a+ib)(c-id)}{c^2+d^2}$$
$$\frac{(a+ib)}{(c+id)} = \frac{a(c-id)+ib(c-id)}{c^2+d^2}$$
$$\frac{(a+ib)}{(c+id)} = \frac{ac-iad+ibc-i^2bd}{c^2+d^2}$$
$$\frac{(a+ib)}{(c+id)} = \frac{ac-iad+ibc-(-1)bd}{c^2+d^2}$$
$$\frac{(a+ib)}{(c+id)} = \frac{ac-iad+ibc+bd}{c^2+d^2}$$
$$\frac{(a+ib)}{(c+id)} = \frac{ac+bd+ibc-iad}{c^2+d^2}$$
$$\frac{(a+ib)}{(c+id)} = \frac{(ac+bd)+i(bc-ad)}{c^2+d^2}$$
$$\frac{(a+ib)}{(c+id)} = \frac{(ac+bd)}{c^2+d^2} + \frac{i(bc-ad)}{c^2+d^2}$$
$$\frac{(a+ib)}{(c+id)} = \frac{(ac+bd)}{c^2+d^2} + i\frac{(bc-ad)}{c^2+d^2}$$

+ Kết quả: $\frac{(ac+bd)}{c^2+d^2} + i\frac{(bc-ad)}{c^2+d^2}$
+ Về mặt kiểu dữ liệu thì số phức chia số phức thì ta được số phức.

– Khai báo hàm

```
04370.  SOPHUC Thuong(SOPHUC,SOPHUC);
```

– Định nghĩa hàm.

```
04371.  SOPHUC Thuong(SOPHUC x, SOPHUC y)
04372.  {
04373.     float mc = y.Thuc*y.Thuc+y.Ao*y.Ao;
04374.     SOPHUC temp;
04375.     temp.Thuc = (x.Thuc*y.Thuc+ x.Ao*y.Ao)/mc;
04376.     temp.Ao = (x.Ao*y.Thuc-x.Thuc*y.Ao)/mc;
04377.     return temp;
04378.  }
```

Bài 070.Tính lũy thừa bậc n của số phức x.

– Khai báo hàm

```
04379. SOPHUC LuyThua(SOPHUC,int);
```

– Định nghĩa hàm.

```
04380. SOPHUC LuyThua(SOPHUC x, int n)
04381. {
04382.     SOPHUC temp = {1,0};
04383.     for (int i = 1; i <= n; i++)
04384.        temp = Tich(temp, x);
04385.     return temp;
04386. }
```

01.08.05 Chủ đề hỗn số – Mixed Numbers

01.08.05.1 Chương trình minh họa chủ đề hỗn số

Chương trình 005. Viết chương trình thực hiện các yêu cầu sau:
a. Nhập hai hỗn số.
b. Kiểm tra hỗn số thứ nhất có phải là hỗn số dương hay không?
c. Tính tổng hai hỗn số.

– Chương trình

```
04387.
```

01.08.05.2 Bài tập chủ đề điểm hỗn số

Bài 071.Khai báo kiểu dữ liệu biểu diễn hỗn số.

– Khai báo kiểu dữ liệu

```
04388. struct honso
04389. {
04390.     int Nguyen;
04391.     int Tu;
04392.     int Mau;
04393. };
04394. typedef struct honso HONSO;
```

Bài 072.Định nghĩa hàm nhập hỗn số.

– Khai báo hàm

```
04395. void Nhap(HONSO&);
```

– Định nghĩa hàm.

```
04396. void Nhap(HONSO &x)
```

```
04397. {
04398.    cout << "Nhap nguyen: ";
04399.    cin >> x.Nguyen;
04400.    cout << "Nhap tu: ";
04401.    cin >> x.Tu;
04402.    cout << "Nhap mau: ";
04403.    cin >> x.Mau;
04404. }
```

Bài 073.Định nghĩa hàm xuất hỗn số.

– Khai báo hàm

```
04405. void Xuat(HONSO);
```

– Định nghĩa hàm.

```
04406. void Xuat(HONSO x)
04407. {
04408.    cout << "\nNguyen: " << x.Nguyen;
04409.    cout << "\nTu: " << x.Tu;
04410.    cout << "\nMau: " << x.Mau;
04411. }
```

Bài 074.Rút gọn hỗn số.

– Khai báo hàm

```
04412. void RutGon(HONSO&);
```

– Định nghĩa hàm.

```
04413. int UCLN(int a, int b)
04414. {
04415.    a = abs(a);
04416.    b = abs(b);
04417.    while (a * b != 0)
04418.       if (a > b)
04419.          a = a - b;
04420.       else
04421.          b = b - a;
04422.    return a + b;
04423. }
04424.
04425. void RutGon(HONSO& x)
04426. {
04427.    int kq = UCLN(x.Tu, x.Mau);
04428.    x.Tu = x.Tu / kq;
04429.    x.Mau = x.Mau / kq;
```

```
04430.     x.Nguyen = x.Nguyen + x.Tu / x.Mau;
04431.     x.Tu = x.Tu % x.Mau;
04432.  }
```

Bài 075.Tính tổng hai hỗn số.

– Khai báo hàm

```
04433.  HONSO Tong(HONSO,HONSO);
```

– Định nghĩa hàm.

```
04434.  HONSO Tong(HONSO x, HONSO y)
04435.  {
04436.     HONSO temp;
04437.     temp.Nguyen = x.Nguyen + y.Nguyen;
04438.     temp.Tu = x.Tu * y.Mau + y.Tu * x.Mau;
04439.     temp.Mau = x.Mau * y.Mau;
04440.     RutGon(temp);
04441.     return temp;
04442.  }
```

Bài 076.Tính hiệu hai hỗn số.

– Khai báo hàm

```
04443.  HONSO Hieu(HONSO,HONSO);
```

– Định nghĩa hàm.

```
04444.  HONSO Hieu(HONSO x, HONSO y)
04445.  {
04446.     HONSO temp;
04447.     temp.Nguyen = x.Nguyen - y.Nguyen;
04448.     temp.Tu = x.Tu * y.Mau - y.Tu * x.Mau;
04449.     temp.Mau = x.Mau * y.Mau;
04450.     RutGon(temp);
04451.     return temp;
04452.  }
```

Bài 077.Tính tích hai hỗn số.

– Khai báo hàm

```
04453.  HONSO Tich(HONSO,HONSO);
```

– Định nghĩa hàm.

```
04454.  HONSO Tich(HONSO x, HONSO y)
04455.  {
04456.     x.Tu = x.Nguyen * x.Mau + x.Tu;
04457.     x.Nguyen = 0;
04458.     y.Tu = y.Nguyen * y.Mau + y.Tu;
```

```
04459.     y.Nguyen = 0;
04460.     HONSO temp;
04461.     temp.Tu = x.Tu * y.Tu;
04462.     temp.Mau = x.Mau * y.Mau;
04463.     temp.Nguyen = 0;
04464.     RutGon(temp);
04465.     return temp;
04466.  }
```

Bài 078.Tính thương hai hỗn số.

– Khai báo hàm

```
04467.  HONSO Thuong(HONSO,HONSO);
```

– Định nghĩa hàm.

```
04468.  HONSO Thuong(HONSO x, HONSO y)
04469.  {
04470.     x.Tu = x.Nguyen * x.Mau + x.Tu;
04471.     x.Nguyen = 0;
04472.     y.Tu = y.Nguyen * y.Mau + y.Tu;
04473.     y.Nguyen = 0;
04474.     HONSO temp;
04475.     temp.Tu = x.Tu * y.Mau;
04476.     temp.Mau = x.Mau * y.Tu;
04477.     temp.Nguyen = 0;
04478.     RutGon(temp);
04479.     return temp;
04480.  }
```

Bài 079.Kiểm tra hỗn số tối giản.

– Khai báo hàm

```
04481.  int ktToiGian(HONSO);
```

– Định nghĩa hàm.

```
04482.  int ktToiGian(HONSO x)
04483.  {
04484.     if(UCLN(x.Tu,x.Mau)==1 && x.Tu/x.Mau==0)
04485.        return 1;
04486.     return 0;
04487.  }
```

Bài 080.Qui đồng mẫu hai hỗn số.

– Khai báo hàm

```
04488.  void QuiDongMau(HONSO&,HONSO&);
```

– Định nghĩa hàm.

```
04489.  void QuiDongMau(HONSO& x, HONSO& y)
04490.  {
04491.      int mc = x.Mau * y.Mau;
04492.      x.Tu = x.Tu * y.Mau;
04493.      y.Tu = x.Mau * y.Tu;
04494.      x.Mau = mc;
04495.      y.Mau = mc;
04496.  }
```

Bài 081.Qui đồng tử hai hỗn số.

– Khai báo hàm

```
04497.  void QuiDongTu(HONSO&,HONSO&);
```

– Định nghĩa hàm.

```
04498.  void QuiDongTu(HONSO &x, HONSO& y)
04499.  {
04500.      int tc = x.Tu * y.Tu;
04501.      x.Mau = x.Mau * y.Tu;
04502.      y.Mau = y.Mau * x.Tu;
04503.      x.Tu = tc;
04504.      y.Tu = tc;
04505.  }
```

01.08.06 Chủ đề thời gian – Time

01.08.06.1 Chương trình minh họa chủ đề thời gian

Chương trình 006. Viết chương trình thực hiện các yêu cầu sau:
a. Nhập một thời gian.
b. Tính số giây trôi qua kể từ khi 00:00:00.
c. Tìm thời gian ở giây kế tiếp.

– Chương trình

```
04506.  #include <iostream>
04507.  #include <iomanip>
04508.  #include <cmath>
04509.  using namespace std;
04510.
04511.  struct thoigian
04512.  {
04513.      int Gio;
04514.      int Phut;
```

```
04515.     int Giay;
04516.  };
04517.  typedef struct thoigian THOIGIAN;
04518.
04519.  void Nhap(THOIGIAN&);
04520.  void Xuat(THOIGIAN);
04521.
04522.  int TinhSoGiay(THOIGIAN);
04523.  THOIGIAN KeTiep(THOIGIAN);
04524.
04525.  int main()
04526.  {
04527.     THOIGIAN A;
04528.     cout << "Nhap thoi gian: ";
04529.     Nhap(A);
04530.
04531.     int kq = TinhSoGiay(A);
04532.     cout << "\nSo giay: " << kq;
04533.
04534.     THOIGIAN B;
04535.     B = KeTiep(A);
04536.     cout << "\nThoi gian ke tiep: ";
04537.     Xuat(B);
04538.     return 1;
04539.  }
04540.
04541.  void Nhap(THOIGIAN &x)
04542.  {
04543.     cout << "\nNhap gio: ";
04544.     cin >> x.Gio;
04545.     cout << "Nhap phut: ";
04546.     cin >> x.Phut;
04547.     cout << "Nhap giay: ";
04548.     cin >> x.Giay;
04549.  }
04550.
04551.  void Xuat(THOIGIAN x)
04552.  {
04553.     cout << "\nGio: " << x.Gio;
04554.     cout << "\nPhut: " << x.Phut;
04555.     cout << "\nGiay: " << x.Giay;
04556.  }
04557.
```

```
04558.  int TinhSoGiay(THOIGIAN x)
04559.  {
04560.     return (x.Gio*3600 + x.Phut*60 + x.Giay);
04561.  }
04562.
04563.  THOIGIAN KeTiep(THOIGIAN x)
04564.  {
04565.     THOIGIAN temp = x;
04566.     temp.Giay = x.Giay + 1;
04567.     if(x.Giay > 59)
04568.     {
04569.        temp.Phut = temp.Phut + 1;
04570.        if(temp.Phut > 59)
04571.        {
04572.           temp.Gio = temp.Gio + 1;
04573.           if(temp.Gio > 23)
04574.              temp.Gio = 0;
04575.           temp.Phut = 0;
04576.        }
04577.        temp.Giay = 0;
04578.     }
04579.     return temp;
04580.  }
```

01.08.06.2 Bài tập chủ đề thời gian

Bài 082.Khai báo kiểu dữ liệu biểu diễn thời gian.

- Kiến thức phổ thông: thời gian có ba thành phần thông tin: giờ, phút, giây.
  - + Một ngày có 24 giờ đánh số từ 0 đến 12.
  - + Một giờ có 60 phút đánh số từ 0 đến 59.
  - + Một phút có 60 giây đánh số từ 0 đến 59.
- Khai báo kiểu dữ liệu.

```
04581.  struct thoigian
04582.  {
04583.     int Gio;
04584.     int Phut;
04585.     int Giay;
04586.  };
04587.  typedef struct thoigian THOIGIAN;
```

Bài 083.Định nghĩa hàm nhập thời gian.

– Khai báo hàm

```
04588.  void Nhap(THOIGIAN&);
```

– Định nghĩa hàm.

```
04589.  void Nhap(THOIGIAN& x)
04590.  {
04591.     cout << "Nhap gio: ";
04592.     cin >> x.Gio;
04593.     cout << "Nhap phut: ";
04594.     cin >> x.Phut;
04595.     cout << "Nhap giay: ";
04596.     cin >> x.Giay;
04597.  }
```

Bài 084.Định nghĩa hàm xuất thời gian.

– Khai báo hàm

```
04598.  void Xuat(THOIGIAN);
```

– Định nghĩa hàm.

```
04599.  void Xuat(THOIGIAN x)
04600.  {
04601.     cout << "\nGio: " << x.Gio;
04602.     cout << "\nPhut: " << x.Phut;
04603.     cout << "\nGiay: " << x.Giay;
04604.  }
```

Bài 085.Định nghĩa hàm kiểm tra tính hợp lệ của một thời gian.

– Khai báo hàm

```
04605.  int ktHopLe(THOIGIAN);
```

– Một thời gian được gọi là hợp lệ khi:
  + Giờ hợp lệ: số nguyên thuộc đoạn [0,23].
  + Phút hợp lệ: số nguyên thuộc đoạn [0,59].
  + Giây hợp lệ: số nguyên thuộc đoạn [0,59].

– Định nghĩa hàm.

```
04606.  int ktHopLe(THOIGIAN x)
04607.  {
04608.     if(!(x.Gio >= 0 && x.Gio<=23))
04609.        return 0;
04610.     if(!(x.Phut >= 0 && x.Phut <=59))
04611.        return 0;
04612.     if(!(x.Giay >= 0 && x.Giay <=59))
04613.        return 0;
04614.     return 1;
```

```
04615. }
```

Bài 086.Tìm thời gian ở giây kế tiếp.

- Ví dụ 01:
  - Dữ liệu vào: 13: 33: 45.
  - Dữ liệu ra: 13: 33: 46.
- Ví dụ 02:
  - Dữ liệu vào: 13: 33: 59.
  - Dữ liệu ra: 13: 34: 00.
- Ví dụ 03:
  - Dữ liệu vào: 13: 59: 59.
  - Dữ liệu ra: 14: 00: 00.
- Ví dụ 04:
  - Dữ liệu vào: 23: 59: 59.
  - Dữ liệu ra: 00: 00: 00.
- Khai báo hàm

```
04616. THOIGIAN KeTiep(THOIGIAN);
```

- Định nghĩa hàm.

```
04617. THOIGIAN KeTiep(THOIGIAN x)
04618. {
04619.    x.Giay++;
04620.    if(x.Giay>59)
04621.    {
04622.       x.Phut++;
04623.       if(x.Phut>59)
04624.       {
04625.          x.Gio++;
04626.          if(x.Gio>23)
04627.             x.Gio = 0;
04628.          x.Phut = 0;
04629.       }
04630.       x.Giay = 0;
04631.    }
04632.    return x;
04633. }
```

Bài 087.Tìm thời gian ở giây trước đó.

- Ví dụ 01:
  - Dữ liệu vào: 13: 33: 46.
  - Dữ liệu ra: 13: 33: 45.
- Ví dụ 02:

+ Dữ liệu vào: 13:34:00.
+ Dữ liệu ra: 13:33:59.

– Ví dụ 03:
+ Dữ liệu vào: 14:00:00.
+ Dữ liệu ra: 13:59:59.

– Ví dụ 04:
+ Dữ liệu vào: 00:00:00.
+ Dữ liệu ra: 23:59:59.

– Khai báo hàm

```
04634.  THOIGIAN TruocDo(THOIGIAN);
```

– Định nghĩa hàm.

```
04635.  THOIGIAN TruocDo(THOIGIAN x)
04636.  {
04637.     x.Giay--;
04638.     if(x.Giay<0)
04639.     {
04640.        x.Phut--;
04641.        if(x.Phut<0)
04642.        {
04643.           x.Gio--;
04644.           if(x.Gio<0)
04645.              x.Gio = 23;
04646.           x.Phut = 59;
04647.        }
04648.        x.Giay = 59;
04649.     }
04650.     return x;
04651.  }
```

Bài 088.Tính số thứ tự giây kể từ lúc 00:00:00.

– Ví dụ:
+ Dữ liệu vào: 13:33:45.
+ Dữ liệu ra: 48.825
+ Giải thích: $13 \times 3600 + 33 \times 60 + 45 = 48.825$

– Khai báo hàm

```
04652.  int SoThuTu(THOIGIAN);
```

– Định nghĩa hàm.

```
04653.  int SoThuTu(THOIGIAN x)
04654.  {
04655.     return x.Gio*3600 + x.Phut*60 + x.Giay;
04656.  }
```

Bài 089.Tính khoảng cách giữa hai thời gian.

- Ví dụ:
  - + Dữ liệu vào:
    - $13:33:45$.
    - $20:21:08$.
  - + Dữ liệu ra: 24.443
  - + Giải thích:
    - $13 \times 3600 + 33 \times 60 + 45 = 48.825$
    - $20 \times 3600 + 21 \times 60 + 08 = 73.268$
- Khai báo hàm

```
04657. int KhoangCach(THOIGIAN,THOIGIAN);
```

- Định nghĩa hàm.

```
04658. int KhoangCach(THOIGIAN x,THOIGIAN y)
04659. {
04660.    int a = SoThuTu(x);
04661.    int b = SoThuTu(y);
04662.    return abs(a-b);
04663. }
```

Bài 090.So sánh hai thời gian.

- Qui ước:

$$Quá\ khứ < Hiện\ tại < Tương\ lai.$$

- Giá trị trả về: hàm trả về một trong 3 giá trị: $+1$, 0, -1.
  - + Giá trị -1: thời gian thứ nhất nhỏ hơn thời gian thứ hai.
  - + Giá trị 0: thời gian thứ nhất bằng thời gian thứ hai.
  - + Giá trị $+1$: thời gian thứ nhất lớn hơn thời gian thứ hai.
- Khai báo hàm.

```
04664. int SoSanh(THOIGIAN,THOIGIAN);
```

- Định nghĩa hàm.

```
04665. int SoSanh(THOIGIAN x, THOIGIAN y)
04666. {
04667.    if(x.Gio > y.Gio)
04668.       return 1;
04669.    if(x.Gio < y.Gio)
04670.       return -1;
04671.    if(x.Phut > y.Phut)
04672.       return 1;
04673.    if(x.Phut < y.Phut)
04674.       return -1;
04675.    if(x.Giay > y.Giay)
```

```
04676.        return 1;
04677.     if(x.Giay < y.Giay)
04678.        return -1;
04679.     return 0;
04680.  }
```

01.08.07 Chủ đề ngày – Date

01.08.07.1 Chương trình minh họa chủ đề ngày

Chương trình 007. Viết chương trình thực hiện các yêu cầu sau:
a. Nhập hai ngày.
b. Tính số ngày trôi qua kể từ ngày 01/01/01.
c. Tìm ngày kế tiếp.

- Chương trình

```
04681.
```

01.08.07.2 Bài tập chủ đề ngày

Bài 091.Khai báo kiểu dữ liệu biểu diễn ngày.

- Kiến thức phổ thông: ngày có ba thành phần thông tin: ngày, tháng, năm.
  - + Năm là số nguyên dương lớn hơn bằng 1.
  - + Tháng là số nguyên dương nằm trong đoạn [1,12].
  - + Ngày là số nguyên dương nằm trong đoạn $[1, x]$ với $x \epsilon \{28,29,30,31\}$.
  - + Ngày đầu tiên theo dương lịch là ngày 01/01/01 và ngày này là ngày thứ hai.
- Khai báo kiểu dữ liệu.

```
04682.  struct ngay
04683.  {
04684.     int ng;
04685.     int th;
04686.     int nm;
04687.  };
04688.  typedef struct ngay NGAY;
```

Bài 092.Định nghĩa hàm nhập ngày.

- Khai báo hàm

```
04689.  void Nhap(NGAY&);
```

- Định nghĩa hàm nhập ngày.

```
04690.    void Nhap(NGAY& x)
04691. {
04692.    cout << "Nhap ngay: ";
04693.    cin >> x.ng;
04694.    cout << "Nhap thang: ";
04695.    cin >> x.th;
04696.    cout << "Nhap nam: ";
04697.    cin >> x.nm;
04698. }
```

Bài 093.Định nghĩa hàm xuất ngày.

- Khai báo hàm.

```
04699. void Xuat(NGAY);
```

- Định nghĩa hàm xuất ngày.

```
04700. void Xuat(NGAY x)
04701. {
04702.    cout << "\nNgay: " << x.ng;
04703.    cout << "\nThang: " << x.th;
04704.    cout << "\nNam: " << x.nm;
04705. }
```

Bài 094.Kiểm tra năm của ngày x có là năm nhuận hay không.

- Một năm được gọi là năm nhuận khi thoả một trong hai điều kiện.
  - Điều kiện 1: Năm chia hết cho 4 và không chia hết cho 100.
  - Điều kiện 2: Năm chia hết cho 400.
- Ví dụ 01:
  - Dữ liệu vào: 1996.
  - Dữ liệu ra: 1.
  - Giải thích: nhuận theo điều kiện 1.
- Ví dụ 02:
  - Dữ liệu vào: 2000.
  - Dữ liệu ra: 1.
  - Giải thích: nhuận theo điều kiện 2.
- Ví dụ 03:
  - Dữ liệu vào: 1900.
  - Dữ liệu ra: 0.
- Khai báo hàm.

```
04706. int ktNhuan(NGAY);
```

- Định nghĩa hàm kiểm tra năm của ngày x có là năm nhuận hay không.

```
04707.  int ktNhuan(NGAY x)
04708.  {
04709.     if(x.nm%4==0 && x.nm%100!=0)
04710.        return 1;
04711.     if(x.nm%400==0)
04712.        return 1;
04713.     return 0;
04714.  }
```

Bài 095.Định nghĩa hàm tìm số ngày tối đa trong tháng của ngày x.

- Kiến thức phổ thông.
  - Các tháng có 31 ngày: 1, 3, 5, 7, 8, 10, 12.
  - Các tháng có 30 ngày: 4, 6, 9, 11.
  - Tháng 2 có 28 hoặc 29 ngày (năm nhuận).
- Ví dụ 01:
  - Dữ liệu vào: 13/05/2014.
  - Dữ liệu ra: 31.
- Ví dụ 02:
  - Dữ liệu vào: 21/09/2000.
  - Dữ liệu ra: 30.
- Ví dụ 03:
  - Dữ liệu vào: 21/02/2000.
  - Dữ liệu ra: 29.
- Ví dụ 04:
  - Dữ liệu vào: 03/02/1992.
  - Dữ liệu ra: 29.
- Ví dụ 05:
  - Dữ liệu vào: 13/02/1990.
  - Dữ liệu ra: 28.
- Ví dụ 06:
  - Dữ liệu vào: 13/02/1900.
  - Dữ liệu ra: 28.
- Khai báo hàm

```
04715.  int SoNgayToiDaTrongThang(NGAY);
```

- Định nghĩa hàm tìm số ngày tối đa trong tháng của ngày x.

```
04716.  int SoNgayToiDaTrongThang(NGAY x)
04717.  {
04718.     int ngaythang[12]={31, 28, 31, 30, 31, 30,
04719.                        31, 31, 30, 31, 30, 31};
04720.     if(ktNhuan(x))
04721.        ngaythang[1] = 29;
```

```
04722.     return ngaythang[x.th-1];
04723. }
```

Bài 096.Định nghĩa hàm tìm số ngày tối đa trong năm của ngày x.

- Kiến thức phổ thông
  - + Năm nhuận có 366 ngày.
  - + Năm không nhuận có 365 ngày.
- Ví dụ 01:
  - + Dữ liệu vào: 13/05/2014.
  - + Dữ liệu ra: 365.
- Ví dụ 02:
  - + Dữ liệu vào: 21/09/2000.
  - + Dữ liệu ra: 366.
- Ví dụ 03:
  - + Dữ liệu vào: 03/02/1992.
  - + Dữ liệu ra: 366.
- Ví dụ 04:
  - + Dữ liệu vào: 13/02/1900.
  - + Dữ liệu ra: 365.
- Khai báo hàm

```
04724. int SoNgayToiDaTrongNam(NGAY);
```

- Định nghĩa hàm tìm số ngày tối đa trong năm của ngày x.

```
04725. int SoNgayToiDaTrongNam(NGAY x)
04726. {
04727.     if(ktNhuan(x))
04728.        return 366;
04729.     return 365;
04730. }
```

Bài 097.Định nghĩa hàm kiểm tra tính hợp lệ của một ngày.

- Một ngày được gọi là hợp lệ khi thỏa các điều kiện:
  - + Năm hợp lệ: năm lớn hơn 1.
  - + Tháng hợp lệ: tháng nằm trong đoạn [1,12].
  - + Ngày hợp lệ: ngày lớn hơn 1 và nhỏ hơn bằng số ngày tối đa của tháng và năm hợp lệ.
- Ví dụ 01:
  - + Dữ liệu vào: 13/05/2014.
  - + Dữ liệu ra: 1.
- Ví dụ 02:
  - + Dữ liệu vào: 31/09/2000.
  - + Dữ liệu ra: 0.

- Ví dụ 03:
  + Dữ liệu vào: $27/15/2000$.
  + Dữ liệu ra: 0.
- Ví dụ 04:
  + Dữ liệu vào: $27/11/-2000$.
  + Dữ liệu ra: 0.
- Khai báo hàm

```
04731. int ktHopLe(NGAY);
```

- Định nghĩa hàm kiểm tra tính hợp lệ của một ngày.

```
04732. int ktHopLe(NGAY x)
04733. {
04734.    if(x.nam<1)
04735.       return 0;
04736.    if(x.th<1)
04737.       return 0;
04738.    if(x.th>12)
04739.       return 0;
04740.    if(x.ng<1)
04741.       return 0;
04742.    if(x.ng>SoNgayToiDaTrongThang[x.th-1])
04743.       return 0;
04744.    return 1;
04745. }
```

Bài 098.So sánh hai ngày.

- Qui ước:

$$Quá\ khứ < Hiện\ tại < Tương\ lai.$$

- Giá trị trả về: hàm trả về một trong 3 giá trị: $+1$, 0, -1.
  + Giá trị -1: ngày thứ nhất nhỏ hơn ngày thứ hai.
  + Giá trị 0: ngày thứ nhất bằng ngày thứ hai.
  + Giá trị $+1$: ngày thứ nhất lớn hơn ngày thứ hai.
- Ví dụ 01:
  + Dữ liệu vào:
    - $27/06/2002$.
    - $14/03/1989$.
  + Dữ liệu ra: $+1$.
- Ví dụ 02:
  + Dữ liệu vào:
    - $27/06/2002$.
    - $14/03/2014$.
  + Dữ liệu ra: -1.

– Ví dụ 03:
  + Dữ liệu vào:
    • 17/08/2012.
    • 17/08/2012.
  + Dữ liệu ra: +0.
– Khai báo hàm.

```
04746.  int SoSanh(NGAY,NGAY);
```

– Định nghĩa hàm so sánh hai ngày.

```
04747.  int SoSanh(NGAY x,NGAY y)
04748.  {
04749.     if(x.nm > y.nm)
04750.        return 1;
04751.     if(x.nm < y.nm)
04752.        return -1;
04753.     if(x.th > y.th)
04754.        return 1;
04755.     if(x.th < y.th)
04756.        return -1;
04757.     if(x.ng > y.ng)
04758.        return 1;
04759.     if(x.ng < y.ng)
04760.        return -1;
04761.     return 0;
04762.  }
```

Bài 099.Tính số thứ tự ngày trong năm.

– Ví dụ 01:
  + Dữ liệu vào: 20/10/2007.
  + Dữ liệu ra: 293.
  + Giải thích: $31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 + 30 + 20 = 293$.

Tháng	01	02	03	04	05	06	07	08	09	10
Số ngày tối đa	31	28	31	30	31	30	31	31	30	20

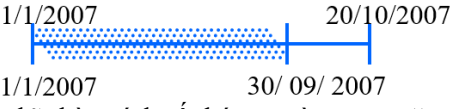
– Ví dụ 02:
  + Dữ liệu vào: 13/04/2020.
  + Dữ liệu ra: 104.
  + Giải thích: $31 + 29 + 31 + 13 = 104$.

Tháng	01	02	03	04
Số ngày tối đa	31	29	31	13

– Khai báo hàm.

```
04763.  int SoThuTuTrongNam(NGAY);
```

- Cách làm: việc tính số thứ tự của ngày trong năm thực hiện qua hai giai đoạn.
  - Giai đoạn 01: tính tổng số ngày trong những tháng đã qua.
  - Giai đoạn 02: tính ngày trong tháng hiện hành.
- Hình vẽ minh họa.



- Định nghĩa hàm tính số thứ tự ngày trong năm.

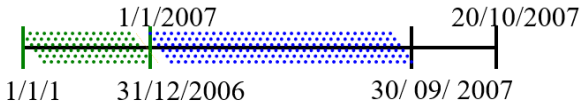
```
04764. int SoThuTuTrongNam(NGAY x)
04765. {
04766.     int stt = 0;
04767.     for(int i=1;i<=x.th-1;i++)
04768.     {
04769.        NGAY temp = {1,i,x.nm};
04770.        stt += SoNgayToiDaTrongThang(temp);
04771.     }
04772.     return (stt + x.ng);
04773. }
```

Bài 100.Tính số thứ tự ngày kể từ ngày 01/01/01.

- Ví dụ 01:
  - Dữ liệu vào: 20/10/2007.
  - Dữ liệu ra: 732.969.
- Ví dụ 02:
  - Dữ liệu vào: 13/04/2020.
  - Dữ liệu ra: 737.528.
- Khai báo hàm.

```
04774. int SoThuTu(NGAY);
```

- Cách làm: việc tính số thứ tự của ngày kể tử ngày 01/01/01 thực hiện qua hai giai đoạn.
  - Giai đoạn 01: tính tổng số ngày trong những năm đã qua (trong ví dụ 01 trên ta tính tổng số ngày từ năm 01 tới năm 2006).
  - Giai đoạn 02: tính số thứ tự ngày trong năm hiện hành (năm 2007).
- Hình vẽ minh họa.



- Định nghĩa hàm tính số thứ tự ngày kể từ ngày 01/01/01.

```
04775.  int SoThuTu(NGAY x)
04776.  {
04777.     int stt = 0;
04778.     for(int i=1;i<=x.nm-1;i++)
04779.     {
04780.        NGAY temp = {1,1,i};
04781.        stt = stt + SoNgayToiDaTrongNam(temp);
04782.     }
04783.     return (stt + SoThuTuTrongNam(x));
04784.  }
```

Bài 101.Xuất thứ của một ngày. Biết rằng ngày 01/01/01 là ngày thứ hai.

- Ví dụ:
  - Dữ liệu vào: 19/07/2019.
  - Dữ liệu ra: "thứ sáu".
- Kiến thức phổ thông.
  - Ngày dương lịch đầu tiên là ngày 01/01/01.
  - Ngày 01/01/01 có số thứ tự ngày kể từ ngày 01/01/01 là 1 và ngày này là ngày thứ hai.
  - Suy luận tương tự, ngày 02/01/01 có số thứ tự ngày kể từ ngày 01/01/01 là 2 và ngày này là ngày thứ ba.
  - Ngày 03/01/01 có số thứ tự ngày kể từ ngày 01/01/01 là 3 và ngày này là ngày thứ tư.
  - Ngày 04/01/01 có số thứ tự ngày kể từ ngày 01/01/01 là 4 và ngày này là ngày thứ năm.
  - Ngày 05/01/01 có số thứ tự ngày kể từ ngày 01/01/01 là 5 và ngày này là ngày thứ sáu.
  - Ngày 06/01/01 có số thứ tự ngày kể từ ngày 01/01/01 là 6 và ngày này là ngày thứ bảy.
  - Ngày 07/01/01 có số thứ tự ngày kể từ ngày 01/01/01 là 7 và ngày này là ngày chủ nhật.
  - Ngày 08/01/01 có số thứ tự ngày kể từ ngày 01/01/01 là 8 và ngày này là ngày thứ hai.
  - …
- Khai báo hàm.

```
04785.  void XuatThu(NGAY);
```

- Định nghĩa hàm xuất thứ của một ngày.

```
04786.  void XuatThu(NGAY x)
04787.  {
04788.     int stt = SoThuTu(x);
```

```
04789.     switch (stt % 7)
04790.     {
04791.        case 0: cout << "Chu Nhat";
04792.           break;
04793.        case 1: cout << "Thu Hai";
04794.           break;
04795.        case 2: cout << "Thu Ba";
04796.           break;
04797.        case 3: cout << "Thu Tu";
04798.           break;
04799.        case 4: cout << "Thu Nam";
04800.           break;
04801.        case 5: cout << "Thu Sau";
04802.           break;
04803.        case 6: cout << "Thu Bay";
04804.           break;
04805.     }
04806. }
```

Bài 102.Khoảng cách giữa hai ngày.

- Ví dụ:
  - + Dữ liệu vào:
    - 27/06/2002.
    - 14/03/1989.
  - + Dữ liệu ra: 4.853.
- Giải thuật.
  - + Tính khoảng cách ngày thứ nhất đến ngày 01/01/01.
  - + Tính khoảng cách ngày thứ hai đến ngày 01/01/01.
  - + Khoảng cách giữa hai ngày là trị tuyệt đối của hiệu hai khoảng cách trên.
- Khai báo hàm

```
04807. int KhoangCach(NGAY,NGAY);
```

- Định nghĩa hàm khoảng cách giữa hai ngày..

```
04808. int KhoangCach(NGAY x,NGAY y)
04809. {
04810.    int a = SoThuTu(x);
04811.    int b = SoThuTu(y);
04812.    return abs(a-b);
04813. }
```

Bài 103.Tìm ngày khi biết năm và số thứ tự của ngày trong năm.

- Ví dụ:
  - Dữ liệu vào: năm 2002, stt = 178.
  - Dữ liệu ra: 27/06/2002.
- Khai báo hàm.

```
04814.  NGAY TimNgay(int,int);
```

- Định nghĩa hàm tìm ngày khi biết năm và số thứ tự của ngày trong năm..

```
04815.  NGAY TimNgay(int nam,int stt)
04816.  {
04817.     NGAY temp = {1,1,nam};
04818.     temp.th = 1;
04819.     while(stt-SoNgayToiDaTrongThang(temp)>0)
04820.     {
04821.        stt = stt - SoNgayToiDaTrongThang(temp);
04822.        temp.th++;
04823.     }
04824.     temp.ng = stt;
04825.     return temp;
04826.  }
```

Bài 104.Tìm ngày khi biết số thứ tự ngày kể từ ngày 01/01/01.

- Ví dụ 01:
  - Dữ liệu vào: stt = 1734.
  - Dữ liệu ra: 30/09/05.
- Ví dụ 02:
  - Dữ liệu vào: stt = 783.456.
  - Dữ liệu ra: 11/01/2146.
- Khai báo hàm.

```
04827.  NGAY TimNgay(int);
```

- Định nghĩa hàm tìm ngày khi biết số thứ tự ngày kể từ ngày 01/01/01..

```
04828.  NGAY TimNgay(int stt)
04829.  {
04830.     int nam = 1;
04831.     int sn = 365;
04832.     while(stt-sn>0)
04833.     {
04834.        stt = stt – sn;
04835.        nam++;
04836.        NGAY temp = {1,1,nam};
04837.        sn = SoNgayToiDaTrongNam(temp);
```

```
04838.     }
04839.     return TimNgay(nam,stt);
04840.  }
```

Bài 105.Tìm ngày kế tiếp.

- Ví dụ 01:
  - Dữ liệu vào: $12/07/2019$.
  - Dữ liệu ra: $13/07/2019$.
- Ví dụ 02:
  - Dữ liệu vào: $31/07/2019$.
  - Dữ liệu ra: $01/08/2019$.
- Ví dụ 03:
  - Dữ liệu vào: $30/04/2019$.
  - Dữ liệu ra: $01/05/2019$.
- Ví dụ 04:
  - Dữ liệu vào: $28/02/2019$.
  - Dữ liệu ra: $01/03/2019$.
- Ví dụ 05:
  - Dữ liệu vào: $29/02/2000$.
  - Dữ liệu ra: $01/03/2000$.
- Ví dụ 06:
  - Dữ liệu vào: $31/12/2014$.
  - Dữ liệu ra: $01/01/2015$.
- Khai báo hàm.

```
04841.  NGAY KeTiep(NGAY);
```

- Cách làm:
  - Gọi ngày cần tìm ngày kế tiếp là ngày x.
  - Tính số thứ tự ngày kể từ ngày $01/01/01$ tới ngày x (gọi là stt).
  - Ngày kế tiếp của ngày x sẽ có số thứ tự ngày kể từ ngày $01/01/01$ là $stt + 1$.
  - Có số thứ tự ngày kể từ ngày ngày $01/01/01$ của ngày kế tiếp ta chỉ cần đổi ra ngày sẽ được ngày kết tiếp.
- Định nghĩa hàm tìm ngày kế tiếp.

```
04842.  NGAY KeTiep(NGAY x)
04843.  {
04844.     int stt = SoThuTu(x);
04845.     stt = stt + 1;
04846.     return TimNgay(stt);
04847.  }
```

Bài 106.Tìm ngày trước đó.

- Ví dụ 01:
  - Dữ liệu vào: 13/07/2019.
  - Dữ liệu ra: 12/07/2019.
- Ví dụ 02:
  - Dữ liệu vào: 01/08/2019.
  - Dữ liệu ra: 31/07/2019.
- Ví dụ 03:
  - Dữ liệu vào: 01/05/2019.
  - Dữ liệu ra: 30/04/2019.
- Ví dụ 04:
  - Dữ liệu vào: 01/03/2019.
  - Dữ liệu ra: 28/02/2019.
- Ví dụ 05:
  - Dữ liệu vào: 01/03/2000.
  - Dữ liệu ra: 29/02/2000.
- Ví dụ 06:
  - Dữ liệu vào: 01/01/2015.
  - Dữ liệu ra: 31/12/2014.
- Ví dụ 07:
  - Dữ liệu vào: 01/01/01.
  - Dữ liệu ra: 01/01/01.
- Khai báo hàm tìm ngày trước đó.

```
04848. NGAY TruocDo(NGAY);
```

- Định nghĩa hàm tìm ngày trước đó.

```
04849. NGAY TruocDo(NGAY x)
04850. {
04851.     if (x.ng == 1 && x.th == 1 && x.nm == 1)
04852.         return x;
04853.     int stt = SoThuTu(x);
04854.     stt = stt - 1;
04855.     return TimNgay(stt);
04856. }
```

Bài 107.Tìm ngày kế tiếp k ngày.

- Ví dụ 01:
  - Dữ liệu vào: 13/07/2019 và $k = 187$.
  - Dữ liệu ra: 16/01/2020.
- Ví dụ 02:
  - Dữ liệu vào: 24/09/1983 và $k = 18.701$.

+ Dữ liệu ra: 06/12/2034.

– Khai báo hàm tìm ngày kế tiếp k ngày.

```
04857. NGAY KeTiep(NGAY,int);
```

– Định nghĩa hàm tìm ngày kế tiếp k ngày.

```
04858. NGAY KeTiep(NGAY x,int k)
04859. {
04860.     int stt = SoThuTu(x);
04861.     stt = stt + k;
04862.     return TimNgay(stt);
04863. }
```

Bài 108.Tìm ngày trước đó k ngày.

– Ví dụ 01:

+ Dữ liệu vào: 13/07/2019 và $k = 872.247$.
+ Dữ liệu ra: 01/01/01.

– Ví dụ 02:

+ Dữ liệu vào: 11/12/1974 và $k = 2018$.
+ Dữ liệu ra: 02/06/1969.

– Khai báo hàm tìm ngày trước đó k ngày.

```
04864. NGAY TruocDo(NGAY,int);
```

– Định nghĩa hàm tìm ngày trước đó k ngày.

```
04865. NGAY TruocDo(NGAY x,int k)
04866. {
04867.    NGAY temp = x;
04868.    for(int i=1;i<=k;i++)
04869.       temp = TruocDo(temp);
04870.    return temp;
04871. }
```

01.08.08 Chủ đề đơn thức – Monomial

01.08.08.1 Chương trình minh họa chủ đề đơn thức

Chương trình 008. Viết chương trình thực hiện các yêu cầu sau:
a. Nhập một đơn thức.
b. Tính giá trị đơn thức tại vị trí $x = x_0$.
c. Tìm đạo hàm cấp 1 của đơn thức.

– Chương trình

```
04872. #include <iostream>
04873. #include <iomanip>
```

```
04874. #include <cmath>
04875. using namespace std;
04876.
04877. struct donthuc
04878. {
04879.    float a;
04880.    int n;
04881. };
04882. typedef struct donthuc DONTHUC;
04883.
04884. void Nhap(DONTHUC&);
04885. void Xuat(DONTHUC);
04886.
04887. float GiaTri(DONTHUC,float);
04888. DONTHUC DaoHam(DONTHUC);
04889.
04890. int main()
04891. {
04892.    DONTHUC ff;
04893.    cout << "Nhap don thuc: ";
04894.    Nhap(ff);
04895.
04896.    float x = 2;
04897.    float kq = GiaTri(ff, x);
04898.    cout << setw(6);
04899.    cout << setprecision(3);
04900.    cout << "\nGia tri tai x la: "<<kq;
04901.
04902.    DONTHUC gg = DaoHam(ff);
04903.    cout << "\nDao ham cap 1:";
04904.    Xuat(gg);
04905.    return 1;
04906. }
04907.
04908. void Nhap(DONTHUC &f)
04909. {
04910.    cout << "\nNhap he so: ";
04911.    cin >> f.a;
04912.    cout << "Nhap so mu: ";
04913.    cin >> f.n;
04914. }
04915. void Xuat(DONTHUC f)
04916. {
```

```
04917.    cout << setw(6);
04918.    cout << setprecision(3);
04919.    cout << "\n a = " << f.a;
04920.    cout << "\n n = " << f.n;
04921. }
04922.
04923. float GiaTri(DONTHUC f, float x)
04924. {
04925.    return(f.a*pow(x, f.n));
04926. }
04927.
04928. DONTHUC DaoHam(DONTHUC f)
04929. {
04930.    if(f.n==0)
04931.    {
04932.       DONTHUC temp = {0, 0};
04933.       return temp;
04934.    }
04935.    DONTHUC temp;
04936.    temp.a = f.a * f.n;
04937.    temp.n = f.n - 1;
04938.    return temp;
04939. }
```

01.08.08.2 Bài tập chủ đề đơn thức

Bài 109.Khai báo kiểu dữ liệu biểu diễn đơn thức.

- Kiến thức phổ thông.
  - + Dạng toán học: $f(x) = ax^n$.
  - + Ví dụ 1: f(x) = $10x^5$.
  - + Ví dụ 2: g(x) = $2x^2$.
  - + Ví dụ 3: h(x) = $12x^7$.
- Khai báo kiểu dữ liệu.

```
04940. struct donthuc
04941. {
04942.    float a;
04943.    int n;
04944. };
04945. typedef struct donthuc DONTHUC;
```

Bài 110.Định nghĩa hàm nhập đơn thức.

- Khai báo hàm

```
04946. void Nhap(DONTHUC &);
```

– Định nghĩa hàm.

```
04947. void Nhap(DONTHUC &f)
04948. {
04949.    cout << "Nhap he so: ";
04950.    cin >> f.a;
04951.    cout << "Nhap so mu: ";
04952.    cin >> f.n;
04953. }
```

Bài 111.Định nghĩa hàm xuất đơn thức.

– Khai báo hàm

```
04954. void Xuat(DONTHUC);
```

– Định nghĩa hàm.

```
04955. void Xuat(DONTHUC f)
04956. {
04957.    cout << "\nHe so: " << f.a;
04958.    cout << "\nSo mu: " << f.n;
04959. }
```

Bài 112.Tính tích hai đơn thức.

– Ôn tập
  + Gọi đơn thức thứ nhất là $f(x) = ax^n$.
  + Gọi đơn thức thứ nhất là $g(x) = bx^m$.
  + Tích của hai đơn thức trên là: $h(x) = fg = abx^{m+n}$.

– Ví dụ:
  + $f(x) = 10x^7$.
  + $g(x) = 2x^4$.
  + Kết quả: $h(\mathrm{x}) = 20x^{11}$.

– Khai báo hàm

```
04960. DONTHUC Tich(DONTHUC,DONTHUC);
```

– Định nghĩa hàm.

```
04961. DONTHUC Tich(DONTHUC f, DONTHUC g)
04962. {
04963.    DONTHUC temp;
04964.    temp.a = f.a * g.a;
04965.    temp.n = f.n + g.n;
04966.    return temp;
04967. }
```

Bài 113.Tính thương hai đơn thức.

- Ôn tập
  - Gọi đơn thức thứ nhất là $f(x) = ax^n$.
  - Gọi đơn thức thứ nhất là $g(x) = bx^m$.
  - Thương của hai đơn thức trên là: $h(x) = \frac{f}{g} = \frac{a}{b}x^{m-n}$.
- Ví dụ:
  - $f(x) = 10x^7$.
  - $g(x) = 2x^4$.
  - Kết quả: $h(\mathrm{x}) = 5x^3$.
- Khai báo hàm

```
04968. DONTHUC Thuong(DONTHUC,DONTHUC);
```

- Định nghĩa hàm.

```
04969. DONTHUC Thuong(DONTHUC f, DONTHUC g)
04970. {
04971.     DONTHUC temp;
04972.     temp.a = f.a / g.a;
04973.     temp.n = f.n - g.n;
04974.     return temp;
04975. }
```

Bài 114.Tính đạo hàm cấp 1 đơn thức.

- Ôn tập
  - Gọi đơn thức ban đầu là $f(x) = ax^n$.
  - Đạo hàm cấp 1 của đơn thức trên là: $f^{(1)}(x) = a.nx^{n-1}$.
- Ví dụ:
  - $f(x) = 10x^7$.
  - Kết quả: $f^{(1)}(x) = 70x^6$.
- Khai báo hàm

```
04976. DONTHUC DaoHam(DONTHUC);
```

- Định nghĩa hàm.

```
04977. DONTHUC DaoHam(DONTHUC f)
04978. {
04979.     DONTHUC temp;
04980.     temp.a = f.a * f.n;
04981.     temp.n = f.n - 1;
04982.     return temp;
04983. }
```

Bài 115.Tính đạo hàm cấp k đơn thức.

- Ôn tập
  - Gọi đơn thức ban đầu là $f(x) = ax^n$.
  - Đạo hàm cấp 1: $f^{(1)}(x) = a.nx^{n-1}$.
  - Đạo hàm cấp 2: $f^{(2)}(x) = a.n(n-1)x^{n-2}$.
  - …
  - Đạo hàm cấp k: $f^{(k)}(x) = a.n.(n-1)\ldots(n-k)x^{n-k}$.
- Ví dụ:Tính đạo hàm cấp 2 của đơn thức $f(x) = 10x^7$.
  - Đạo hàm cấp 1: $f^{(1)}(x) = 70x^6$.
  - Đạo hàm cấp 2: $f^{(2)}(x) = 420x^5$.
- Khai báo hàm

```
04984. DONTHUC DaoHam(DONTHUC,int);
```

- Định nghĩa hàm.

```
04985. DONTHUC DaoHam(DONTHUC f, int k)
04986. {
04987.    DONTHUC temp = f;
04988.    for (int i = 1; i <= k; i++)
04989.       temp = DaoHam(temp);
04990.    return temp;
04991. }
```

Bài 116.Tính giá trị đơn thức tại vị trí $x = x_0$.

- Khai báo hàm

```
04992. float TinhGiaTri(DONTHUC,float);
```

- Định nghĩa hàm.

```
04993. float TinhGiaTri(DONTHUC f, float x)
04994. {
04995.    float T = 1;
04996.    for (int i = 1; i <= f.n; i++)
04997.       T = T * x;
04998.    T = T * f.a;
04999.    return T;
05000. }
```

Bài 117.Định nghĩa toán tử tích (*operator* *) cho hai đơn thức.

- Khai báo hàm

```
05001. DONTHUC operator*(DONTHUC,DONTHUC);
```

- Định nghĩa hàm.

```
05002. DONTHUC operator*(DONTHUC f, DONTHUC g)
05003. {
05004.    DONTHUC temp;
```

```
05005.     temp.a = f.a * g.a;
05006.     temp.n = f.n + g.n;
05007.     return temp;
05008.  }
```

Bài 118.Định nghĩa toán tử thương (*operator /*) cho hai đơn thức.

– Khai báo hàm

```
05009.  DONTHUC operator/(DONTHUC,DONTHUC);
```

– Định nghĩa hàm.

```
05010.  DONTHUC operator/(DONTHUC f, DONTHUC g)
05011.  {
05012.     DONTHUC temp;
05013.     temp.a = f.a / g.a;
05014.     temp.n = f.n - g.n;
05015.     return temp;
05016.  }
```

01.08.09 Chủ đề đường tròn trong mặt phẳng Oxy – Circle

01.08.09.1 Chương trình minh họa chủ đề đường tròn

Chương trình 009. Viết chương trình thực hiện các yêu cầu sau:
a. Nhập tọa độ tâm và bán kính của một đường tròn trong mặt phẳng Oxy.
b. Tính diện tích đường tròn.
c. Tính chu vi đường tròn.

```
05017.  #include <iostream>
05018.  #include <iomanip>
05019.  using namespace std;
05020.
05021.  struct diem
05022.  {
05023.     float x;
05024.     float y;
05025.  };
05026.  typedef struct diem DIEM;
05027.
05028.  struct duongtron
05029.  {
05030.     DIEM I;
05031.     float R;
```

```
05032.  };
05033.  typedef struct duongtron DUONGTRON;
05034.
05035.  void Nhap(DIEM &);
05036.  void Xuat(DIEM);
05037.
05038.  void Nhap(DUONGTRON &);
05039.  void Xuat(DUONGTRON);
05040.
05041.  float DienTich(DUONGTRON);
05042.  float ChuVi(DUONGTRON);
05043.
05044.  int main()
05045.  {
05046.     DUONGTRON dt;
05047.     Nhap(dt);
05048.     Xuat(dt);
05049.     float kq = DienTich(dt);
05050.     cout << "Dien tich = " << kq;
05051.
05052.     kq = ChuVi(dt);
05053.     cout << "Dien tich = " << kq;
05054.     return 1;
05055.  }
05056.
05057.  float DienTich(DUONGTRON c)
05058.  {
05059.     return 3.14 * c.R * c.R;
05060.  }
05061.  float ChuVi(DUONGTRON c)
05062.  {
05063.     return 2 * 3.14 * c.R;
05064.  }
```

01.08.09.2 Đường tròn trong mặt phẳng Oxy

Bài 119.Khai báo kiểu dữ liệu biểu diễn đường tròn trong mặt phẳng Oxy.

– Khai báo kiểu dữ liệu.

```
05065.  struct diem
05066.  {
05067.     float x;
05068.     float y;
```

```
05069. };
05070. typedef struct diem DIEM;
05071.
05072. struct duongthang
05073. {
05074.    float a;
05075.    float b;
05076.    float c;
05077. };
05078. typedef struct duongthang DUONGTHANG;
05079.
05080. struct duongtron
05081. {
05082.    DIEM I;
05083.    float R;
05084. };
05085. typedef struct duongtron DUONGTRON;
```

Bài 120.Định nghĩa hàm nhập tọa độ đường tròn trong mặt phẳng Oxy.

– Khai báo hàm

```
05086. void Nhap(DUONGTRON&);
```

– Định nghĩa hàm.

```
05087. void Nhap(DUONGTRON& c)
05088. {
05089.    cout << "Nhap tam:\n";
05090.    Nhap(c.I);
05091.    cout << "Nhap ban kinh: ";
05092.    cin >> c.R;
05093. }
```

Bài 121.Định nghĩa hàm xuất tọa độ đường tròn.

– Khai báo hàm

```
05094. void Xuat(DUONGTRON&);
```

– Định nghĩa hàm.

```
05095. void Xuat(DUONGTRON c)
05096. {
05097.    cout << "\nTam:";
05098.    Xuat(c.I);
05099.    cout << "\nBan kinh: " << c.R;
05100. }
```

Bài 122.Tính chu vi đường tròn.

- Khai báo hàm

```
05101.  float ChuVi(DUONGTRON);
```

- Định nghĩa hàm.

```
05102.  float ChuVi(DUONGTRON c)
05103.  {
05104.     return float(2 * 3.14 * c.R);
05105.  }
```

Bài 123.Tính diện tích đường tròn.

- Khai báo hàm

```
05106.  float DienTich(DUONGTRON);
```

- Định nghĩa hàm.

```
05107.  float DienTich(DUONGTRON c)
05108.  {
05109.     return float(3.14 * c.R * c.R);
05110.  }
```

Bài 124.Kiểm tra một tọa độ điểm có nằm trong đường tròn hay không.

- Khai báo hàm

```
05111.  int KtThuoc(DUONGTRON,DIEM);
```

- Định nghĩa hàm.

```
05112.  int ktThuoc(DUONGTRON c, DIEM P)
05113.  {
05114.     float kc = KhoangCach(c.I, P);
05115.     if (kc <= c.R)
05116.        return 1;
05117.     return 0;
05118.  }
```

Bài 125.Xét vị trí tương đối giữa một điểm và một đường tròn (nằm trong đường tròn, nằm trên đường tròn, nằm ngoài đường tròn).

- Giá trị trả về: Hàm trả về một trong các giá trị sau:
  + 0. Điểm nằm trong đường tròn.
  + 1. Điểm nằm trên đường tròn.
  + 2. Điểm nằm ngoài đường tròn.
- Khai báo hàm

```
05119.  int TuongDoi(DUONGTRON,DIEM);
```

- Định nghĩa hàm.

```
05120.  int TuongDoi(DUONGTRON c, DIEM P)
05121.  {
05122.     float kc = KhoangCach(c.I, P);
05123.     if (kc > c.R)
05124.        return 2;
05125.     if (kc < c.R)
05126.        return 0;
05127.     return 1;
05128.  }
```

Bài 126.Xét vị trí tương đối giữa một đường thẳng và một đường tròn (không cắt, tiếp tuyến, cắt đường tròn).

- Giá trị trả về: Hàm trả về một trong các giá trị sau:
  - 0. Đường thẳng không cắt đường tròn.
  - 1. Đường thẳng tiếp xúc với đường tròn.
  - 2. Đường thẳng cắt đường trong tại hai điểm.
- Khai báo hàm

```
05129.  int TuongDoi(DUONGTRON,DUONGTHANG);
```

- Định nghĩa hàm.

```
05130.  int TuongDoi(DUONGTRON c, DUONGTHANG d)
05131.  {
05132.     float kc = KhoangCach(d, c.I);
05133.     if (kc > c.R)
05134.        return 0;
05135.     if (kc < c.R)
05136.        return 2;
05137.     return 1;
05138.  }
```

Bài 127.Xét vị trí tương đối giữa hai đường tròn (trùng nhau, không cắt nhau, tiếp xúc ngoài, cắt nhau, tiếp xúc trong, chứa trong nhau).

- Giá trị trả về: Hàm trả về một trong các giá trị sau:
  - 0. Hai đường tròn trùng nhau.
  - 1. Hai đường tròn rời nhau.
  - 2. Hai đường tròn tiếp xúc ngoài.
  - 3. Hai đường tròn cắt nhau.
  - 4. Hai đường tròn tiếp xúc trong.
  - 5. Hai đường tròn chứa trong nhau.
- Khai báo hàm

```
05139.  int TuongDoi(DUONGTRON,DUONGTRON);
```

- Định nghĩa hàm.

```
05140.  int TuongDoi(DUONGTRON c1, DUONGTRON c2)
05141.  {
05142.     float kc = KhoangCach(c1.I, c2.I);
05143.     if (kc == 0 && c1.R == c2.R)
05144.        return 0;
05145.     if (kc > (c1.R + c2.R))
05146.        return 1;
05147.     if (kc == (c1.R + c2.R))
05148.        return 2;
05149.     if (kc < (c1.R + c2.R) && kc > abs(c1.R -
c2.R))
05150.        return 3;
05151.     if (kc == abs(c1.R - c2.R))
05152.        return 4;
05153.     return 5;
05154.  }
```

01.08.10 Chủ đề hình cầu trong không gian Oxyz – Sphere

01.08.10.1 Chương trình minh họa chủ đề hình cầu

Chương trình 010. Viết chương trình thực hiện các yêu cầu sau:
a. Nhập một hình cầu.
b. Nhập điểm M.
c. Kiểm tra điểm M có thuộc hình cầu không?
d. Tính thể tích của hình cầu.

– Chương trình

```
05155.  #include <iostream>
05156.  #include <iomanip>
05157.  #include <cmath>
05158.  using namespace std;
05159.
05160.  struct diemkhonggian
05161.  {
05162.     float x;
05163.     float y;
05164.     float z;
05165.  };
05166.  typedef struct diemkhonggian DIEMKHONGGIAN;
05167.
05168.  struct hinhcau
05169.  {
```

```
05170.     DIEMKHONGGIAN I;
05171.     float R;
05172.  };
05173.  typedef struct hinhcau HINHCAU;
05174.
05175.  void Nhap(DIEMKHONGGIAN&);
05176.  void Xuat(DIEMKHONGGIAN);
05177.
05178.  void Nhap(HINHCAU&);
05179.  void Xuat(HINHCAU);
05180.
05181.  float KhoangCach(DIEMKHONGGIAN,DIEMKHONGGIAN);
05182.  int ktThuoc(HINHCAU,DIEMKHONGGIAN);
05183.  float TheTich(HINHCAU);
05184.
05185.  int main()
05186.  {
05187.     DIEMKHONGGIAN M;
05188.     HINHCAU hc;
05189.
05190.     cout << "Nhap hinh cau: ";
05191.     Nhap(hc);
05192.     cout << "Nhap diem M: ";
05193.     Nhap(M);
05194.
05195.     if(ktThuoc(hc, M))
05196.        cout << "\nM thuoc hinh cau.";
05197.     else
05198.        cout << "\nM ko thuoc hinh cau.";
05199.
05200.     float kq = TheTich(HC);
05201.     cout << setw(6);
05202.     cout << setprecision(3);
05203.     cout << "\nThe tich hinh cau: " << kq;
05204.     return 1;
05205.  }
05206.
05207.  void Nhap(DIEMKHONGGIAN &P)
05208.  {
05209.     cout << "\nNhap x: ";
05210.     cin >> P.x;
05211.     cout << "Nhap y: ";
05212.     cin >> P.y;
```

```
05213.    cout << "Nhap y: ";
05214.    cin >> P.z;
05215. }
05216.
05217. float KhoangCach(DIEMKHONGGIAN P,
05218.                   DIEMKHONGGIAN Q)
05219. {
05220.    return sqrt((Q.x - P.x)*(Q.x - P.x) +
05221.                (Q.y - P.y)*(Q.y - P.y) +
05222.                (Q.z - P.z)*(Q.z - P.z));
05223. }
05224.
05225. void Nhap(HINHCAU& C)
05226. {
05227.    cout << "\nNhap tam I:";
05228.    Nhap(C.I);
05229.    cout << "Nhap R:";
05230.    cin >> C.R;
05231. }
05232.
05233. int ktThuoc(HINHCAU C, DIEMKHONGGIAN P)
05234. {
05235.    float kc = KhoangCach(C.I, P);
05236.    if(kc <= C.R)
05237.       return 1;
05238.    return 0;
05239. }
05240.
05241. float TheTich(HINHCAU C)
05242. {
05243.    return ((float(4) / 3)*3.14*pow(C.R, 3));
05244. }
```

01.08.10.2 Bài tập chủ đề hình cầu

Bài 128.Khai báo kiểu dữ liệu biểu diễn hình cầu trong mặt phẳng Oxy.

– Khai báo kiểu dữ liệu.

```
05245. struct diemkhonggian
05246. {
05247.    float x;
05248.    float y;
05249.    float z;
05250. };
```

```
05251. typedef struct diemkhonggian DIEMKHONGGIAN;
05252.
05253. struct hinhcau
05254. {
05255.    DIEMKHONGGIAN I;
05256.    float R;
05257. };
05258. typedef struct hinhcau HINHCAU;
```

Bài 129.Định nghĩa hàm nhập hình cầu.

– Khai báo hàm

```
05259. void Nhap(HINHCAU&);
```

– Định nghĩa hàm.

```
05260. void Nhap(HINHCAU& c)
05261. {
05262.    cout << "Nhap tam:\n";
05263.    Nhap(c.I);
05264.    cout << "Nhap ban kinh: ";
05265.    cin >> c.R;
05266. }
```

Bài 130.Định nghĩa hàm xuất tọa độ hình cầu.

– Khai báo hàm

```
05267. void Xuat(HINHCAU);
```

– Định nghĩa hàm.

```
05268. void Xuat(HINHCAU c)
05269. {
05270.    cout << "\nTam:";
05271.    Xuat(c.I);
05272.    cout << "\nBan kinh: " << c.R;
05273. }
```

Bài 131.Tính diện tích xung quanh hình cầu.

– Khai báo hàm

```
05274. float DienTich(HINHCAU);
```

– Định nghĩa hàm.

```
05275. float DienTich(HINHCAU c)
05276. {
05277.    return float(4 * 3.14 * c.R * c.R);
05278. }
```

Bài 132.Tính thể tích hình cầu.

- Khai báo hàm

```
05279. float TheTich(HINHCAU);
```

- Định nghĩa hàm.

```
05280. float TheTich(HINHCAU c)
05281. {
05282.    return float((float)4 / 3 * 3.14 * c.R * c.R *
c.R);
05283. }
```

Bài 133.Kiểm tra một tọa độ điểm có nằm bên trong hình cầu hay không.

- Khai báo hàm

```
05284. int KtThuoc(HINHCAU,DIEMKHONGGIAN);
```

- Định nghĩa hàm.

```
05285. int ktThuoc(HINHCAU c, DIEMKHONGGIAN P)
05286. {
05287.    float kc = KhoangCach(c.I, P);
05288.    if (kc <= c.R)
05289.       return 1;
05290.    return 0;
05291. }
```

Bài 134.Xét vị trí tương đối giữa hai hình cầu (trùng nhau, rời nhau, tiếp xúc ngoài, cắt nhau, tiếp xúc trong, chứa trong nhau).

- Giá trị trả về: Hàm trả về một trong các giá trị sau:
  + 0. Hai hình cầu trùng nhau.
  + 1. Hai hình cầu rời nhau.
  + 2. Hai hình cầu tiếp xúc ngoài.
  + 3. Hai hình cầu cắt nhau.
  + 4. Hai hình cầu tiếp xúc trong.
  + 5. Hai hình cầu chứa trong nhau.
- Khai báo hàm

```
05292. float TuongDoi(HINHCAU,HINHCAU);
```

- Định nghĩa hàm.

```
05293. int TuongDoi(HINHCAU c1, HINHCAU c2)
05294. {
05295.    float kc = KhoangCach(c1.I, c2.I);
05296.    if (kc == 0 && c1.R == c2.R)
05297.       return 0;
```

```
05298.     if (kc > (c1.R + c2.R))
05299.        return 1;
05300.     if (kc == (c1.R + c2.R))
05301.        return 2;
05302.     if (kc < (c1.R+c2.R) && kc>abs(c1.R-c2.R))
05303.        return 3;
05304.     if (kc == abs(c1.R - c2.R))
05305.        return 4;
05306.     return 5;
05307. }
```

01.08.11 Chủ đề tam giác – Triangle

01.08.11.1 Chương trình minh họa chủ đề tam giác

Chương trình 011. Viết chương trình nhập tọa độ 3 đỉnh của một tam giác trong mặt phẳng Oxy. Tính diện tích, chu vi và tọa độ trọng tâm của tam giác và xuất kết quả.

```
05308. #include <iostream>
05309. #include <iomanip>
05310. #include <cmath>
05311. using namespace std;
05312.
05313. struct diem
05314. {
05315.    float x;
05316.    float y;
05317. };
05318. typedef struct diem DIEM;
05319. struct tamgiac
05320. {
05321.    DIEM A;
05322.    DIEM B;
05323.    DIEM C;
05324. };
05325. typedef struct tamgiac TAMGIAC;
05326.
05327. void Nhap(DIEM &);
05328. void Xuat(DIEM );
05329.
05330. void Nhap(TAMGIAC &);
05331. void Xuat(TAMGIAC);
```

```
05332.
05333.  float KhoangCach(DIEM,DIEM);
05334.  float DienTich(TAMGIAC);
05335.  float ChuVi(TAMGIAC);
05336.  DIEM TrongTam(TAMGIAC);
05337.
05338.  int main()
05339.  {
05340.     TAMGIAC tg;
05341.     Nhap(tg);
05342.     Xuat(tg);
05343.
05344.     float kq = DienTich(tg);
05345.     cout << "Dien tich = " << kq;
05346.     kq = ChuVi(tg);
05347.     cout << "Chu vi = " << kq;
05348.
05349.     DIEM G = trongtam(tg);
05350.     cout << "Trong tam = " << endl;
05351.     Xuat(G);
05352.     return 1;
05353.  }
05354.
05355.  float DienTich(TAMGIAC x)
05356.  {
05357.     float a = KhoangCach(x.B,x.C);
05358.     float b = KhoangCach(x.C,x.A);
05359.     float c = KhoangCach(x.A,x.B);
05360.     float p = (a+b+c)/2;
05361.     return sqrt(p*(p-a)*(p-b)*(p-c));
05362.  }
05363.  float ChuVi(TAMGIAC x)
05364.  {
05365.     float a = KhoangCach(x.B,x.C);
05366.     float b = KhoangCach(x.C,x.A);
05367.     float c = KhoangCach(x.A,x.B);
05368.     return (a+b+c);
05369.  }
05370.  DIEM TrongTam(TAMGIAC t)
05371.  {
05372.     DIEM G;
05373.     G.x = (t.A.x + t.B.x + t.C.x)/3;
05374.     G.y = (t.A.y + t.B.y + t.C.y)/3;
```

```
05375.    return G;
05376. }
```

01.08.11.2 Bài tập chủ đề tam giác

Hãy viết các hàm thực hiện các yêu cầu sau:

Bài 135.Khai báo kiểu dữ liệu biểu diễn tam giác trong mặt phẳng Oxy.

– Khai báo kiểu dữ liệu tam giác.

```
05377. struct diem
05378. {
05379.    float x;
05380.    float y;
05381. };
05382. typedef struct diem DIEM;
05383. struct tamgiac
05384. {
05385.    DIEM A;
05386.    DIEM B;
05387.    DIEM C;
05388. };
05389. typedef struct tamgiac TAMGIAC;
```

Bài 136.Định nghĩa hàm nhập tam giác trong mặt phẳng Oxy.

– Khai báo hàm

```
05390. void Nhap(TAMGIAC&);
```

– Định nghĩa hàm.

```
05391. void Nhap(TAMGIAC& t)
05392. {
05393.    cout << "Nhap dinh A:\n";
05394.    Nhap(t.A);
05395.    cout << "Nhap dinh B:\n";
05396.    Nhap(t.B);
05397.    cout << "Nhap dinh C:\n";
05398.    Nhap(t.C);
05399. }
```

Bài 137.Định nghĩa hàm xuất tam giác.

– Khai báo hàm

```
05400. void Xuat(TAMGIAC);
```

– Định nghĩa hàm.

```
05401. void Xuat(TAMGIAC t)
```

```
05402. {
05403.    cout << "\nToa do dinh A:";
05404.    Xuat(t.A);
05405.    cout << "\nToa do dinh B:";
05406.    Xuat(t.B);
05407.    cout << "\nToa do dinh C:";
05408.    Xuat(t.C);
05409. }
```

Bài 138.Kiểm tra tọa độ 3 đỉnh có thật sự lập thành 3 đỉnh của một tam giác hay không?

– Khai báo hàm

```
05410. int KiemTra(TAMGIAC);
```

– Định nghĩa hàm.

```
05411. int KiemTra(TAMGIAC t)
05412. {
05413.    float a = KhoangCach(t.B, t.C);
05414.    float b = KhoangCach(t.A, t.C);
05415.    float c = KhoangCach(t.A, t.B);
05416.    if (a + b > c && b + c > a && a + c > b)
05417.       return 1;
05418.    return 0;
05419. }
```

Bài 139.Tính chu vi tam giác.

– Khai báo hàm

```
05420. float ChuVi(TAMGIAC);
```

– Định nghĩa hàm.

```
05421. float ChuVi(TAMGIAC t)
05422. {
05423.    float a = KhoangCach(t.B, t.C);
05424.    float b = KhoangCach(t.A, t.C);
05425.    float c = KhoangCach(t.A, t.B);
05426.    return a + b + c;
05427. }
```

Bài 140.Tính diện tích tam giác.

– Khai báo hàm

```
05428. float DienTich(TAMGIAC);
```

– Định nghĩa hàm.

```
05429. float DienTich(TAMGIAC t)
```

```
05430. {
05431.    float a = KhoangCach(t.B, t.C);
05432.    float b = KhoangCach(t.A, t.C);
05433.    float c = KhoangCach(t.A, t.B);
05434.    float p = (a + b + c) / 2;
05435.    return sqrt(p * (p - a) * (p - b) * (p - c));
05436. }
```

Bài 141.Tìm tọa độ trọng tâm của tam giác.

– Khai báo hàm

```
05437. DIEM TrongTam(TAMGIAC);
```

– Định nghĩa hàm.

```
05438. DIEM TrongTam(TAMGIAC t)
05439. {
05440.    DIEM temp;
05441.    temp.x = (t.A.x + t.B.x + t.C.x) / 3;
05442.    temp.y = (t.A.y + t.B.y + t.C.y) / 3;
05443.    return temp;
05444. }
```

Bài 142.Tìm một đỉnh trong tam giác có hoành độ lớn nhất.

– Khai báo hàm

```
05445. DIEM HoanhLonNhat(TAMGIAC);
```

– Định nghĩa hàm.

```
05446. DIEM HoanhLonNhat(TAMGIAC t)
05447. {
05448.    DIEM lc = t.A;
05449.    if (t.B.x > lc.x)
05450.       lc = t.B;
05451.    if (t.C.x > lc.x)
05452.       lc = t.C;
05453.    return lc;
05454. }
```

Bài 143.Tìm một đỉnh trong tam giác có tung độ nhỏ nhất.

– Khai báo hàm

```
05455. DIEM TungNhoNhat(TAMGIAC);
```

– Định nghĩa hàm.

```
05456. DIEM TungNhoNhat(TAMGIAC t)
05457. {
05458.    DIEM lc = t.A;
```

```
05459.    if (t.B.y < lc.y)
05460.       lc = t.B;
05461.    if (t.C.y < lc.y)
05462.       lc = t.C;
05463.    return lc;
05464. }
```

Bài 144.Tính tổng khoảng cách từ điểm $P(x,y)$ tới 3 đỉnh của tam giác.

– Khai báo hàm

```
05465. float TongKhoangCach(TAMGIAC,DIEM);
```

– Định nghĩa hàm.

```
05466. float TongKhoangCach(TAMGIAC t, DIEM P)
05467. {
05468.    float a = KhoangCach(t.A, P);
05469.    float b = KhoangCach(t.B, P);
05470.    float c = KhoangCach(t.C, P);
05471.    return a + b + c;
05472. }
```

Bài 145.Hãy cho biết dạng của tam giác (*không là tam giác, đều, vuông, vuông cân, cân, thường*).

– Giá trị trả về: Hàm trả về một trong các giá trị sau:
  + 0. Không là tam giác.
  + 1. Tam giác đều.
  + 2. Tam giác vuông cân.
  + 3. Tam giác vuông.
  + 4. Tam giác cân.
  + 5. Tam giác thường.

– Khai báo hàm

```
05473. int DangTamGiac(TAMGIAC);
```

– Định nghĩa hàm.

```
05474. int DangTamGiac(TAMGIAC t)
05475. {
05476.    float a = KhoangCach(t.B, t.C);
05477.    float b = KhoangCach(t.A, t.C);
05478.    float c = KhoangCach(t.A, t.B);
05479.    if(!(a+b>c && b+c>a && c+a>b))
05480.       return 0;
05481.    if (a==b && b==c)
05482.       return 1;
```

```
05483.     if(a*a == b*b + c*c ||
05484.        b*b == a*a + c*c ||
05485.        c*c == a*a + b*b)
05486.     {
05487.        if (a == b || b == c || a == c)
05488.           return 2;
05489.        return 3;
05490.     }
05491.     if (a == b || b == c || a == c)
05492.        return 4;
05493.     return 5;
05494.  }
```

01.08.12 Chủ đề đường thẳng trong mặt phẳng Oxy – Line

01.08.12.1 Chương trình minh họa chủ đề đường thẳng

Chương trình 012. Viết chương trình thực hiện các yêu cầu sau:
a. Nhập hai đường thẳng $d1$ và $d2$.
b. Kiểm tra hai đường thẳng có trùng nhau không?

– Chương trình

```
05495.  #include <iostream>
05496.  #include <iomanip>
05497.  #include <cmath>
05498.  using namespace std;
05499.
05500.  struct diemkhonggian
05501.  {
05502.     float x;
05503.     float y;
05504.     float z;
05505.  };
05506.  typedef struct diemkhonggian DIEMKHONGGIAN;
05507.
05508.  void Nhap(DIEMKHONGGIAN&);
05509.  void Xuat(DIEMKHONGGIAN);
05510.
05511.  DIEMKHONGGIAN DoiXungOxy(DIEMKHONGGIAN);
05512.  float KhoangCachX(DIEMKHONGGIAN,DIEMKHONGGIAN);
05513.
05514.  int main()
05515.  {
```

```
05516.     DIEMKHONGGIAN A, B;
05517.     cout << "Nhap toa do diem 1: ";
05518.     Nhap(A);
05519.     cout << "Nhap toa do diem 2: ";
05520.     Nhap(B);
05521.     cout << "\nToa do diem 1: ";
05522.     Xuat(A);
05523.     cout << "\nToa do diem 2: ";
05524.     Xuat(B);
05525.
05526.     DIEMKHONGGIAN C;
05527.     C = DoiXungOxy(A);
05528.     cout << "\nDiem doi xung A tren Oxy: ";
05529.     Xuat(C);
05530.
05531.     float kq = KhoangCachX(A, B);
05532.     cout << setw(6);
05533.     cout << setprecision(3);
05534.     cout << "\nKhoang cach x: " << kq;
05535.     return 1;
05536. }
05537.
05538. void Nhap(DIEMKHONGGIAN &P)
05539. {
05540.     cout << "\nNhap x: ";
05541.     cin >> P.x;
05542.     cout << "Nhap y: ";
05543.     cin >> P.y;
05544.     cout << "Nhap y: ";
05545.     cin >> P.z;
05546. }
05547.
05548. void Xuat(DIEMKHONGGIAN P)
05549. {
05550.     cout << setw(6);
05551.     cout << setprecision(3);
05552.     cout << "\nx = " << P.x;
05553.     cout << "\ny = " << P.y;
05554.     cout << "\nz = " << P.z;
05555. }
05556.
05557. DIEMKHONGGIAN DoiXungOxy(DIEMKHONGGIAN P)
05558. {
```

```
05559.    DIEMKHONGGIAN temp;
05560.    temp.x = -P.x;
05561.    temp.y = -P.y;
05562.    temp.z = +P.z;
05563.    return temp;
05564. }
05565.
05566. float KhoangCachX(DIEMKHONGGIAN P,
05567.       DIEMKHONGGIAN Q)
05568. {
05569.    return abs(Q.x-P.x);
05570. }
```

01.08.12.2 Bài tập chủ đề đường thẳng

Bài 146.Hãy khai báo kiểu dữ liệu biểu diễn đường thẳng $ax + by + c = 0$ trong mặt phẳng Oxy.

– Khai báo kiểu dữ liệu.

```
05571. struct duongthang
05572. {
05573.    float a;
05574.    float b;
05575.    float c;
05576. };
05577. typedef struct duongthang DUONGTHANG;
```

Bài 147.Định nghĩa hàm nhập đườngng thẳng trong mặt phẳng Oxy.

– Khai báo hàm

```
05578. void Nhap(DUONGTHANG&);
```

– Định nghĩa hàm nhập đườngng thẳng trong mặt phẳng Oxy.

```
05579. void Nhap(DUONGTHANG &d)
05580. {
05581.    cout << "Nhap a: ";
05582.    cin >> d.a;
05583.    cout << "Nhap b: ";
05584.    cin >> d.b;
05585.    cout << "Nhap c: ";
05586.    cin >> d.c;
05587. }
```

Bài 148.Định nghĩa hàm xuất đường thẳng.

– Khai báo hàm

```
05588. void Xuat(DUONGTHANG);
```

– Định nghĩa hàm xuất đường thẳng.

```
05589. void Xuat(DUONGTHANG d)
05590. {
05591.    cout << "\na: " << d.a;
05592.    cout << "\nb: " << d.b;
05593.    cout << "\nc: " << d.c;
05594. }
```

Bài 149.Kiểm tra một điểm có thuộc đường thẳng hay không.

– Kiến thức phổ thông: một điểm thuộc đường thẳng khi ta thế tọa độ điểm ấy vào phương trình đường thẳng và đẳng thức vẫn được thỏa.
  + Gọi đường thẳng là Δ: $ax + by + c = 0$.
  + Gọi điểm cần kiểm tra thuộc là P: $P(x_P, y_P)$.
  + Điểm $P(x_P, y_P)$ được gọi là thuộc đường thẳng Δ khi $ax_P + by_P + c = 0$.

– Khai báo hàm.

```
05595. int ktThuoc(DUONGTHANG,DIEM);
```

– Định nghĩa hàm kiểm tra một điểm có thuộc đường thẳng hay không.

```
05596. int ktThuoc(DUONGTHANG d,DIEM P)
05597. {
05598.    if((d.a*P.x + d.b*P.y + d.c)==0)
05599.       return 1;
05600.    return 0;
05601. }
```

Bài 150.Tính khoảng cách giữa một điểm và một đường thẳng.

– Kiến thức phổ thông: khoảng cách giữa một điểm và một đường thẳng.
  + Gọi đường thẳng là Δ: $ax + by + c = 0$.
  + Gọi điểm cần tính khoảng cách là $P(x_P, y_P)$.
  + Khoảng cách từ điểm $P(x_P, y_P)$ đến đường thẳng Δ là: $d(P, \Delta) = \frac{|ax_P + by_P + c|}{\sqrt{a^2 + b^2}}$.

– Khai báo hàm

```
05602. float KhoangCach(DUONGTHANG,DIEM);
```

– Định nghĩa hàm tính khoảng cách giữa một điểm và một đường thẳng.

```
05603. float KhoangCach(DUONGTHANG d,DIEM P)
```

```
05604. {
05605.     float tu = abs(d.a*P.x + d.b*P.y + d.c);
05606.     float mau = sqrt(d.a*d.a + d.b*d.b);
05607.     return tu/mau;
05608. }
```

Bài 151.Kiểm tra hai đường thẳng có trùng nhau hay không.

- Kiến thức phổ thông
  - + Đường thẳng thứ nhất: $a_1x + b_1y + c_1 = 0$.
  - + Đường thẳng thứ hai: $a_2x + b_2y + c_2 = 0$.
  - + Ta có hệ phương trình
    $$\begin{cases} a_1x + b_1y + c_1 = 0 \\ a_2x + b_2y + c_2 = 0 \end{cases}$$
  - + Chuyển vế đổi dấu ta được
    $$\begin{cases} a_1x + b_1y = -c_1 \\ a_2x + b_2y = -c_2 \end{cases}$$
  - + Tính các định thức:
    - $\mathrm{D} = \begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix} = a_1b_2 - a_2b_1$
    - $\mathrm{D}_x = \begin{vmatrix} -c_1 & b_1 \\ -c_2 & b_2 \end{vmatrix} = (-c_1)b_2 - (-c_2)b_1$
    - $\mathrm{D}_x = -c_1b_2 + c_2b_1$
    - $\mathrm{D}_y = \begin{vmatrix} a_1 & -c_1 \\ a_2 & -c_2 \end{vmatrix} = a_1(-c_2) - a_2(-c_1)$
    - $\mathrm{D}_y = -a_1c_2 + a_2c_1$
  - + Nếu $\mathrm{D} \neq 0$ hệ phương có nghiệm duy nhất.
    - $x = \frac{\mathrm{D}_x}{D}$.
    - $y = \frac{\mathrm{D}_y}{D}$.
    - Khi đó hai đường cắt nhau tại duy nhất một điểm.
    - Tọa độ giao điểm là $(x, y) \equiv \left(\frac{\mathrm{D}_x}{D}, \frac{\mathrm{D}_y}{D}\right)$.
  - + Nếu $\mathrm{D} = 0$ và $\mathrm{D}_x = 0$ thì hệ phương có vô số nghiệm. Khi đó hai đường thẳng trùng nhau.
  - + Nếu $\mathrm{D} = 0$ và $\mathrm{D}_x \neq 0$ thì hệ phương vô nghiệm. Khi đó hai đường thẳng song song.
  - + Nhắc lại:
    - $\mathrm{D} = a_1b_2 - a_2b_1$
    - $\mathrm{D}_x = -c_1b_2 + c_2b_1$
    - $\mathrm{D}_y = -a_1c_2 + a_2c_1$
- Khai báo hàm.

```
05609.  int ktTrung(DUONGTHANG,DUONGTHANG);
```

- Định nghĩa hàm kiểm tra hai đường thẳng có trùng nhau hay không.

```
05610.  int ktTrung(DUONGTHANG d1,DUONGTHANG d2)
05611.  {
05612.     float D = d1.a*d2.b – d2.a*d1.b;
05613.     float Dx = -d1.c*d2.b + d2.c*d1.b;
05614.     if(D==0 && Dx==0)
05615.        return 1;
05616.     return 0;
05617.  }
```

Bài 152.Kiểm tra hai đường thẳng có song song với nhau hay không.

- Nhắc lại:
  - $D = a_1b_2 - a_2b_1$
  - $D_x = -c_1b_2 + c_2b_1$
  - $D_y = -a_1c_2 + a_2c_1$
  - Nếu $D = 0$ và $D_x \neq 0$ thì hệ phương vô nghiệm. Khi đó hai đường thẳng song song.
- Khai báo hàm

```
05618.  int ktSongSong(DUONGTHANG,DUONGTHANG);
```

- Định nghĩa hàm kiểm tra hai đường thẳng có song song với nhau hay không.

```
05619.  int ktSongSong(DUONGTHANG d1,DUONGTHANG d2)
05620.  {
05621.     float D = d1.a*d2.b – d2.a*d1.b;
05622.     float Dx = -d1.c*d2.b + d2.c*d1.b;
05623.     if(D==0 && Dx!=0)
05624.        return 1;
05625.     return 0;
05626.  }
```

Bài 153.Kiểm tra hai đường thẳng có cắt nhau hay không.

- Nhắc lại:
  - $D = a_1b_2 - a_2b_1$
  - $D_x = -c_1b_2 + c_2b_1$
  - $D_y = -a_1c_2 + a_2c_1$
  - Nếu $D \neq 0$ hệ phương có nghiệm duy nhất. Khi đó hai đường cắt nhau tại duy nhất một điểm.
- Khai báo hàm.

```
05627.  int ktCat(DUONGTHANG,DUONGTHANG);
```

– Định nghĩa hàm kiểm tra hai đường thẳng có cắt nhau hay không.

```
05628.  int ktCat(DUONGTHANG d1,DUONGTHANG d2)
05629.  {
05630.     float D = d1.a*d2.b – d2.a*d1.b;
05631.     if(D!=0)
05632.        return 1;
05633.     return 0;
05634.  }
```

Bài 154.Tìm tọa độ giao điểm của hai đường thẳng (biết rằng hai đường thẳng cắt nhau).

– Nhắc lại:
  + $D = a_1b_2 - a_2b_1$
  + $D_x = -c_1b_2 + c_2b_1$
  + $D_y = -a_1c_2 + a_2c_1$
  + Nếu $D \neq 0$ hệ phương có nghiệm duy nhất. Khi đó hai đường cắt nhau tại duy nhất một điểm.

– Khai báo hàm

```
05635.  DIEM GiaoDiem(DUONGTHANG,DUONGTHANG);
```

– Định nghĩa hàm tìm tọa độ giao điểm của hai đường thẳng.

```
05636.  DIEM GiaoDiem(DUONGTHANG d1,DUONGTHANG d2)
05637.  {
05638.     float D = d1.a*d2.b – d2.a*d1.b;
05639.     float Dx = -d1.c*d2.b + d2.c*d1.b;
05640.     float Dy = -d1.a*d2.c + d2.a*d1.c;
05641.
05642.     DIEM temp;
05643.     temp.x = Dx/D;
05644.     temp.y = Dy/D;
05645.     return temp;
05646.  }
```

Bài 155.Kiểm tra hai đường thẳng có vuông góc với nhau hay không.

– Kiến thức phổ thông
  + Gọi đường thẳng thứ nhất là Δ_1: $a_1x + b_1y + c_1 = 0$.
  + Gọi đường thẳng thứ hai là Δ_2: $a_2x + b_2y + c_2 = 0$.
  + $\Delta_1 \perp \Delta_2$ khi $a_1a_2 + b_1b_2 = 0$.

– Khai báo hàm

```
05647.  int ktVuongGoc(DUONGTHANG,DUONGTHANG);
```

– Định nghĩa hàm kiểm tra hai đường thẳng có vuông góc với nhau hay không.

```
05648.  int ktVuongGoc(DUONGTHANG d1,DUONGTHANG d2)
05649.  {
05650.     if((d1.a*d2.a+d1.b*d2.b)==0)
05651.        return 1;
05652.     return 0;
05653.  }
```

01.08.13 Chủ đề đa thức – Polynomial

01.08.13.1 Bài tập chủ đề đa thức

Bài 156.Khai báo kiểu dữ liệu biểu diễn đa thức.

– Khai báo kiểu dữ liệu.

Bài 157.Định nghĩa hàm nhập đa thức.

– Khai báo hàm.

```
05654.  void Nhap(DATHUC &);
```

Bài 158.Định nghĩa hàm xuất đa thức.

– Khai báo hàm.

```
05655.  void Xuat(DATHUC );
```

Bài 159.Định nghĩa hàm điều chỉnh bậc của đa thức cho chính xác.

– Khai báo hàm.

```
05656.  void DieuChinhBac(DATHUC &);
```

Bài 160.Tính tổng hai đa thức.

– Khai báo hàm.

```
05657.  DATHUC Tong(DATHUC,DATHUC);
```

Bài 161.Tính hiệu hai đa thức.

– Khai báo hàm.

```
05658.  DATHUC Hieu(DATHUC,DATHUC);
```

Bài 162.Tính tích hai đa thức.

– Khai báo hàm.

```
05659.  DATHUC Tich(DATHUC,DATHUC);
```

Bài 163.Tính thương hai đa thức.

– Khai báo hàm.

```
05660. DATHUC Thuong(DATHUC,DATHUC);
```

Bài 164.Tính đa thức dư của phép chia đa thức thứ nhất cho đa thức thứ hai.

– Khai báo hàm.

```
05661. DATHUC ThuongLayDu(DATHUC,DATHUC);
```

Bài 165.Tính đạo hàm cấp 1 của đa thức.

– Khai báo hàm.

```
05662. DATHUC DaoHam(DATHUC);
```

Bài 166.Tính đạo hàm cấp k của đa thức.

– Khai báo hàm.

```
05663. DATHUC DaoHam(DATHUC,int);
```

Bài 167.Tính giá trị của đa thức tại vị trí $x = x_0$.

– Khai báo hàm.

```
05664. float TinhGiaTri(DATHUC,float);
```

Bài 168.Định nghĩa toán tử cộng (`operator +`) cho hai đa thức.

– Khai báo hàm.

```
05665. DATHUC operator+(DATHUC,DATHUC);
```

Bài 169.Định nghĩa toán tử hiệu (`operator -`) cho hai đa thức.

– Khai báo hàm.

```
05666. DATHUC operator-(DATHUC,DATHUC);
```

Bài 170.Định nghĩa toán tử tích (`operator *`) cho hai đa thức.

– Khai báo hàm.

```
05667. DATHUC operator*(DATHUC,DATHUC);
```

Bài 171.Định nghĩa toán tử thương (`operator /`) cho hai đa thức.

– Khai báo hàm.

```
05668. DATHUC operator/(DATHUC,DATHUC);
```

Bài 172.Tìm nghiệm của đa thức trong đoạn [a, b] cho trước.

– Khai báo hàm.

```
05669.  DATHUC operator%(DATHUC,DATHUC);
```

01.09 MẢNG MỘT CHIỀU CẤU TRÚC

01.09.01 Mảng một chiều tọa độ điểm

Bài 173.Viết hàm nhập mảng tọa độ các điểm.

– Khai báo hàm.

```
05670.  void Nhap(DIEM [],int &);
```

– Định nghĩa hàm.

```
05671.  void Nhap(DIEM a[],int &n)
05672.  {
05673.     cout << "Nhap n: ";
05674.     cin >> n;
05675.     for(int i=0; i<n; i++)
05676.     {
05677.        cout << "Nhap a[" << i << "]: ";
05678.        Nhap(a[i]);
05679.     }
05680.  }
```

Bài 174.Viết hàm xuất mảng tọa độ các điểm.

– Khai báo hàm.

```
05681.  void Xuat(DIEM [],int);
```

– Định nghĩa hàm.

```
05682.  void Xuat(DIEM a[],int n)
05683.  {
05684.     for(int i=0; i<=n-1; i++)
05685.        Xuat(a[i]);
05686.  }
```

Bài 175.Đếm số lượng điểm trong mảng có hoành độ dương.

– Khai báo hàm.

```
05687.  int DemDiem(DIEM [],int);
```

– Định nghĩa hàm.

```
05688.  int DemDiem(DIEM a[],int n)
05689.  {
05690.     int dem=0;
```

```
05691.     for(int i=0; i<=n-1; i++)
05692.        if(a[i].x > 0)
05693.           dem++;
05694.     return dem;
05695.  }
```

Bài 176.Tìm một điểm trong mảng có tung độ lớn nhất trong mảng.

– Khai báo hàm.

```
05696.  DIEM TungLonNhat(DIEM [],int);
```

– Định nghĩa hàm.

```
05697.  DIEM TungLonNhat(DIEM a[],int n)
05698.  {
05699.     DIEM lc = a[0];
05700.     for(int i=0; i<=n-1; i++)
05701.        if(a[i].y > lc.y)
05702.           lc = a[i];
05703.     return lc;
05704.  }
```

Bài 177.Tìm một điểm trong mảng gần gốc tọa độ nhất.

– Khai báo hàm.

```
05705.  float KhoangCachGoc(DIEM);
05706.  DIEM GanGocNhat(DIEM [],int);
```

– Định nghĩa hàm.

```
05707.  float KhoangCachGoc(DIEM P)
05708.  {
05709.     return sqrt(P.x*P.x + P.y*P.y);
05710.  }
```

– Định nghĩa hàm.

```
05711.  DIEM GanGocNhat(DIEM a[],int n)
05712.  {
05713.     DIEM lc = a[0];
05714.     for(int i=0; i<=n-1; i++)
05715.        if(KhoangCachGoc(a[i])<
05716.           KhoangCachGoc(lc))
05717.           lc = a[i];
05718.     return lc;
05719.  }
```

Bài 178.Giả sử mảng nhiều hơn 2 phần tử và các phần tử trong mảng đôi một không trùng nhau. Hãy viết hàm tìm tọa độ hai điểm gần nhau nhất trong mảng.

– Khai báo hàm.

```
05720. float KhoangCach(DIEM,DIEM);
05721. void GanNhauNhat(DIEM [],int,DIEM&,DIEM&);
```

– Định nghĩa hàm.

```
05722. float KhoangCach(DIEM P, DIEM Q)
05723. {
05724.    return sqrt((P.x - Q.x) * (P.x - Q.x) +
05725.                (P.y - Q.y) * (P.y - Q.y));
05726. }
```

– Định nghĩa hàm.

```
05727. void GanNhauNhat(DIEM a[],int n,
05728.                    DIEM &P,DIEM &Q)
05729. {
05730.    P = a[0];
05731.    Q = a[1];
05732.    for(int i=0; i<=n-2; i++)
05733.       for(int j=i+1; j<=n-1; j++)
05734.          if(KhoangCach(a[i],a[j])<
05735.             KhoangCach(P,Q))
05736.          {
05737.             P = a[i];
05738.             Q = a[j];
05739.          }
05740. }
```

Bài 179.Đếm số lượng phần tử không trùng với phần tử khác trong mảng.

– Khai báo hàm.

```
05741. int ktTrung (DIEM,DIEM);
05742. int TanSuat(DIEM [],int,DIEM);
05743. int DemDiem(DIEM [],int);
```

– Định nghĩa hàm.

```
05744. int ktTrung(DIEM P,DIEM Q)
05745. {
05746.    if(P.x==Q.x && P.y==Q.y)
05747.       return 1;
05748.    return 0;
```

```
05749. }
```

– Định nghĩa hàm.

```
05750. int TanSuat(DIEM a[],int n,DIEM P)
05751. {
05752.    int dem = 0;
05753.    for (int i = 0; i < n; i++)
05754.       if(ktTrung(a[i],P)==1)
05755.          dem++;
05756.    return dem;
05757. }
```

– Định nghĩa hàm.

```
05758. int DemDiem(DIEM a[],int n)
05759. {
05760.    int dem = 0;
05761.    for (int i = 0; i < n; i++)
05762.       if(TanSuat(a,n,a[i])==1)
05763.          dem++;
05764.    return dem;
05765. }
```

Bài 180.(*) Cho n điểm $P_1, P_2, P_3, \cdots, P_n$ trong mặt phẳng Oxy trong đó không có 2 điểm nào trùng nhau. Một tam giác với các đỉnh thuộc các điểm $P_1, P_2, P_3, \cdots, P_n$ được gọi là độc lập nếu nó không chứa $(n-3)$ điểm còn lại. Hãy viết hàm tìm một tam giác độc lập trong n điểm này.

Bài 181.(*) Cho n điểm $P_1, P_2, P_3, \cdots, P_n$ trong mặt phẳng Oxy trong đó không có 2 điểm nào trùng nhau. Hãy tìm một đa giác lồi với các đỉnh là các điểm trong số n điểm cho trước, sao cho đa giác lồi chứa tất cả các điểm còn lại.

01.09.02 Mảng một chiều phân số

Bài 182.Viết hàm nhập mảng một chiều các phân số.

– Khai báo hàm.

```
05766. void Nhap(PHANSO [],int &);
```

– Định nghĩa hàm.

```
05767. void Nhap(PHANSO a[],int &n)
05768. {
05769.    cout << "Nhap n: ";
05770.    cin >> n;
```

```
05771.    for(int i=0; i<n; i++)
05772.    {
05773.       cout << "Nhap a[" << i << "]: ";
05774.       Nhap(a[i]);
05775.    }
05776. }
```

Bài 183.Viết hàm xuất mảng một chiều các phân số.

– Khai báo hàm.

```
05777. void Xuat(PHANSO [],int);
```

– Định nghĩa hàm.

```
05778. void Xuat(PHANSO a[],int n)
05779. {
05780.    for(int i=0; i<=n-1; i++)
05781.       Xuat(a[i]);
05782. }
```

Bài 184.Viết hàm đếm số lượng phân số dương trong mảng.

– Khai báo hàm.

```
05783. int DemDuong(PHANSO [],int);
```

– Định nghĩa hàm.

```
05784. int DemDuong(PHANSO a[],int n)
05785. {
05786.    int dem = 0;
05787.    for(int i=0; i<=n-1; i++)
05788.       if(ktDuong(a[i]))
05789.          dem++;
05790.    return dem;
05791. }
```

Bài 185.Tìm phân số lớn nhất trong mảng.

– Khai báo hàm.

```
05792. PHANSO LonNhat(PHANSO [],int);
```

– Định nghĩa hàm.

```
05793. PHANSO LonNhat(PHANSO a[],int n)
05794. {
05795.    PHANSO lc = a[0];
05796.    for(int i=0; i<=n-1; i++)
05797.       if(SoSanh(a[i],lc)==1)
05798.          lc = a[i];
05799.    return lc;
```

```
05800. }
```

Bài 186.Tìm "một vị trí mà giá trị tại vị trí đó là giá trị nhỏ nhất" trong mảng.

– Khai báo hàm.

```
05801. int ViTriNhoNhat(PHANSO [],int);
```

– Định nghĩa hàm.

```
05802. int ViTriNhoNhat(PHANSO a[],int n)
05803. {
05804.    int lc = 0;
05805.    for(int i=0; i<=n-1; i++)
05806.       if(SoSanh(a[i],a[lc])==-1)
05807.          lc = i;
05808.    return lc;
05809. }
```

Bài 187.Tìm giá trị dương đầu tiên trong mảng. Nếu mảng không có giá trị dương thì trả về giá trị phân số 0/1.

– Khai báo hàm.

```
05810. PHANSO DuongDau(PHANSO [],int);
```

– Định nghĩa hàm.

```
05811. PHANSO DuongDau(PHANSO a[],int n)
05812. {
05813.    for(int i=0; i<=n-1; i++)
05814.       if(ktDuong(a[i]))
05815.          return a[i];
05816.    PHANSO lc = {0,1};
05817.    return lc;
05818. }
```

Bài 188.Hãy tìm giá trị dương nhỏ nhất trong mảng. Nếu mảng không có giá trị dương thì trả về phân số không dương là – 1/1.

– Khai báo hàm.

```
05819. PHANSO DuongNhoNhat(PHANSO [],int);
```

– Định nghĩa hàm.

```
05820. PHANSO DuongNhoNhat(PHANSO a[],int n)
05821. {
05822.    if(n==0)
05823.    {
05824.       PHANSO lc = {-1,1};
05825.       return lc;
```

```
05826.     }
05827.     PHANSO lc = DuongNhoNhat(a,n-1);
05828.     if(ktDuong(a[n-1])==0)
05829.           return lc;
05830.     if(ktDuong(lc)==0)
05831.           return a[n-1];
05832.     if(SoSanh(a[n-1],lc)==1)
05833.        lc = a[n-1];
05834.     return lc;
05835. }
```

Bài 189.Hãy tìm vị trí giá trị âm lớn nhất trong mảng. Nếu mảng không có giá trị âm thì trả về một giá trị ngoài đoạn $[0, n-1]$ là -1 nhằm mô tả không có vị trí nào thỏa điều kiện.

– Khai báo hàm.

```
05836. int TimViTri(PHANSO [],int);
```

– Định nghĩa hàm.

```
05837. int TimViTri(PHANSO a[],int n)
05838. {
05839.     if(n==0)
05840.        return -1;
05841.     int lc = TimViTri(a,n-1);
05842.     if(ktAm(a[n-1])==0)
05843.        return lc;
05844.     if(ktAm(lc)==0)
05845.        return n-1;
05846.     if(SoSanh(a[n-1],a[lc])==-1)
05847.        lc = n-1;
05848.     return lc;
05849. }
```

Bài 190.Sắp xếp mảng tăng dần.

– Khai báo hàm.

```
05850. void SapTang(PHANSO [],int);
```

– Định nghĩa hàm.

```
05851. void SapTang(PHANSO a[],int n)
05852. {
05853.     for(int i=0;i<=n-2;i++)
05854.        for(int j=i+1;j<=n-1;j++)
05855.           if(SoSanh(a[i],a[j])==1)
05856.              HoanVi(a[i],a[j]);
```

```
05857. }
```

01.09.03 Mảng một chiều số phức

Bài 191.Viết hàm nhập mảng một chiều các số phức.

– Khai báo hàm.

```
05858. void Nhap(SOPHUC [],int &);
```

– Định nghĩa hàm.

```
05859. void Nhap(SOPHUC a[],int &n)
05860. {
05861.    cout << "Nhap n: ";
05862.    cin >> n;
05863.    for(int i=0; i<n; i++)
05864.    {
05865.       cout << "Nhap a[" << i << "]: ";
05866.       Nhap(a[i]);
05867.    }
05868. }
```

Bài 192.Viết hàm xuất mảng một chiều các số phức.

– Khai báo hàm.

```
05869. void Xuat(SOPHUC [],int);
```

– Định nghĩa hàm.

```
05870. void Xuat(SOPHUC a[],int n)
05871. {
05872.    for(int i=0; i<=n-1; i++)
05873.       Xuat(a[i]);
05874. }
```

Bài 193.Tính tổng tất cả các số phức có trong mảng.

– Khai báo hàm.

```
05875. SOPHUC Tong(SOPHUC [],int);
```

– Định nghĩa hàm.

```
05876. SOPHUC Tong(SOPHUC a[],int n)
05877. {
05878.    SOPHUC s = {0,0};
05879.    for(int i=0; i<=n-1; i++)
05880.       s = Tong(s,a[i]);
05881.    return s;
05882. }
```

Bài 194.Tìm số phức đầu tiên trong mảng có phần thực dương và phần ảo dương.

– Khai báo hàm.

```
05883.  SOPHUC DauTien(SOPHUC [],int);
```

– Định nghĩa hàm.

```
05884.  SOPHUC DauTien(SOPHUC a[],int n)
05885.  {
05886.     for(int i=0; i<=n-1; i++)
05887.        if(a[i].Thuc>0 && a[i].Ao>0)
05888.           return a[i];
05889.     SOPHUC lc = {0,0};
05890.     return lc;
05891.  }
```

Bài 195.Sắp các số phức trong mảng tăng dần theo phần thực.

– Khai báo hàm.

```
05892.  void SapTang(SOPHUC [],int);
```

– Định nghĩa hàm.

```
05893.  void SapTang(SOPHUC a[],int n)
05894.  {
05895.     for(int i=0;i<=n-2;i++)
05896.        for(int j=i+1;j<=n-1;j++)
05897.           if(a[i].Thuc > a[j].Thuc)
05898.              HoanVi(a[i],a[j]);
05899.  }
```

01.09.04 Mảng một chiều các đường tròn

Bài 196.Viết hàm nhập mảng một chiều các đường tròn.

– Khai báo hàm.

```
05900.  void Nhap(DUONGTRON [],int &);
```

– Định nghĩa hàm.

```
05901.  void Nhap(DUONGTRON a[],int &n)
05902.  {
05903.     cout << "Nhap n: ";
05904.     cin >> n;
05905.     for(int i=0; i<n; i++)
05906.     {
05907.        cout << "Nhap a[" << i << "]: ";
05908.        Nhap(a[i]);
```

```
05909.     }
05910.  }
```

Bài 197.Viết hàm xuất mảng một chiều các đường tròn.

– Khai báo hàm.

```
05911.  void Xuat(DUONGTRON [],int);
```

– Định nghĩa hàm.

```
05912.  void Xuat(DUONGTRON a[],int n)
05913.  {
05914.     for(int i=0; i<=n-1; i++)
05915.        Xuat(a[i]);
05916.  }
```

Bài 198.Cho n đường tròn. Tìm một đường tròn gần gốc tọa độ nhất?

– Khai báo hàm.

```
05917.  float KhoangCachGoc(DUONGTRON);
05918.  DUONGTRON GanGocNhat(DUONGTRON [],int);
```

– Định nghĩa hàm.

```
05919.  float KhoangCachGoc(DUONGTRON c)
05920.  {
05921.     float kc = KhoangCachGoc(c.I);
05922.     return abs(kc-c.R);
05923.  }
```

– Định nghĩa hàm.

```
05924.  DUONGTRON GanGocNhat(DUONGTRON a[], int n)
05925.  {
05926.     DUONGTRON lc = a[0];
05927.     for (int i = 1; i < n; i++)
05928.        if(KhoangCachGoc(a[i]) <
05929.           KhoangCachGoc(lc))
05930.           lc = a[i];
05931.     return lc;
05932.  }
```

Bài 199.Cho n đường tròn. Hãy kiểm tra xem có một đường tròn nào trong n đường tròn đi qua gốc tọa độ hay không?

– Khai báo hàm.

```
05933.  int ktQuaGoc(DUONGTRON);
05934.  int ktTonTai(DUONGTRON [],int);
```

– Định nghĩa hàm.

```
05935.  int ktQuaGoc(DUONGTRON c)
05936.  {
05937.     float kc = KhoangCachGoc(c.I);
05938.     if(kc==c.R)
05939.        return 1;
05940.     return 0;
05941.  }
```

– Định nghĩa hàm.

```
05942.  int ktTonTai(DUONGTRON a[], int n)
05943.  {
05944.     int flag = 0;
05945.     for (int i = 0; i < n; i++)
05946.        if(ktQuaGoc(a[i])==1)
05947.           flag = 1;
05948.     return flag;
05949.  }
```

Bài 200.Cho n đường tròn. Tìm một đường tròn trong n đường tròn đó gần trục Ox nhất?

– Khai báo hàm.

```
05950.  float KhoangCachOx(DUONGTRON);
05951.  DUONGTRON GanOxNhat(DUONGTRON [],int);
```

– Định nghĩa hàm.

```
05952.  float KhoangCachOx(DUONGTRON c)
05953.  {
05954.     if(abs(c.I.y)<c.R)
05955.        return 0;
05956.     return abs(abs(c.I.y)-c.R);
05957.  }
```

– Định nghĩa hàm.

```
05958.  DUONGTRON GanOxNhat(DUONGTRON a[], int n)
05959.  {
05960.     DUONGTRON lc = a[0];
05961.     for (int i = 0; i < n; i++)
05962.        if(KhoangCachOx(a[i])<KhoangCachOx(lc))
05963.           lc = a[i];
05964.     return lc;
05965.  }
```

Bài 201.Cho n đường tròn. Hãy kiểm tra xem có đường tròn nào tiếp xúc trục tung hay không?

– Khai báo hàm.

```
05966.  int ktTiepXucOy(DUONGTRON);
05967.  int KiemTra(DUONGTRON [],int);
```

– Định nghĩa hàm.

```
05968.  int ktTiepXucOy(DUONGTRON c)
05969.  {
05970.     if(abs(c.I.x)==c.R)
05971.        return 1;
05972.     return 0;
05973.  }
```

– Định nghĩa hàm.

```
05974.  int KiemTra(DUONGTRON a[], int n)
05975.  {
05976.     int flag = 0;
05977.     for (int i = 0; i < n; i++)
05978.        if(ktTiepXucOy(a[i])==1)
05979.           flag = 1;
05980.     return flag;
05981.  }
```

Bài 202.Cho n đường tròn. Hãy kiểm tra xem các đường tròn này có đôi một cắt nhau không?

– Khai báo hàm.

```
05982.  int ktDoiMotCatNhau(DUONGTRON [],int);
```

– Định nghĩa hàm.

```
05983.  int ktDoiMotCatNhau(DUONGTRON a[],int n)
05984.  {
05985.     int flag = 1;
05986.     for(int i=0;i<=n-2;i++)
05987.        for(int j=i+1;j<=n-1;j++)
05988.           if(TuongDoi(a[i],a[j])!=3)
05989.              flag = 0;
05990.     return flag;
05991.  }
```

Bài 203.Cho n đường tròn đôi một không trùng nhau. Hãy liệt kê tất cả các cặp đường tròn tiếp xúc nhau.

– Khai báo hàm.

```
05992.  void LietKe(DUONGTRON [],int);
```

– Định nghĩa hàm.

```
05993.  void LietKe(DUONGTRON a[],int n)
05994.  {
05995.     for(int i=0;i<=n-2;i++)
05996.        for(int j=i+1;j<=n-1;j++)
05997.           if(TuongDoi(a[i],a[j])==2||
05998.               TuongDoi(a[i],a[j])==4)
05999.           {
06000.              Xuat(a[i]);
06001.              Xuat(a[j]);
06002.           }
06003.  }
```

Bài 204.(*) Cho n đường tròn. Hãy tính diện tích phần mặt phẳng bị phủ bởi n đường tròn này.

Bài 205.(*) Cho n đường tròn. Hỏi có tồn tại một điểm trong mặt phẳng Oxy thuộc tất cả các đường tròn này hay không? Nếu có hãy chỉ ra tọa độ một điểm.

01.09.05 Mảng một chiều các đường thẳng

Bài 206.Viết hàm nhập mảng một chiều các đường thẳng.

– Khai báo hàm.

```
06004.  void Nhap(DUONGTHANG [],int &);
```

– Định nghĩa hàm.

```
06005.  void Nhap(DUONGTHANG a[], int& n)
06006.  {
06007.     cout << "Nhap n: ";
06008.     cin >> n;
06009.     for (int i = 0; i < n; i++)
06010.     {
06011.        cout << "Nhap duong thang A["<<i<<"]:\n";
06012.        Nhap(a[i]);
06013.     }
06014.  }
```

Bài 207.Viết hàm xuất mảng một chiều các đường thẳng.

– Khai báo hàm.

```
06015.  void Xuat(DUONGTHANG [],int);
```

– Định nghĩa hàm.

```
06016.  void Xuat(DUONGTHANG a[], int n)
```

```
06017. {
06018.    for (int i = 0; i < n; i++)
06019.    {
06020.       cout << "Duong thang A[" << i << "]:";
06021.       Xuat(a[i]);
06022.       cout << endl;
06023.    }
06024. }
```

Bài 208.Cho *n* đường thẳng. Hãy kiểm tra các đường thẳng này có cùng song song với nhau không.

– Khai báo hàm.

```
06025. int ktSongSong(DUONGTHANG [],int);
```

– Định nghĩa hàm.

```
06026. int ktSongSong(DUONGTHANG a[], int n)
06027. {
06028.    int flag = 1;
06029.    for (int i = 0; i <=n-1; i++)
06030.       if (!ktSongSong(a[i],a[0]))
06031.          flag = 0;;
06032.    return flag;
06033. }
```

– Định nghĩa hàm.

```
06034. int ktSongSong(DUONGTHANG a[], int n)
06035. {
06036.    for (int i = 0; i <=n-1; i++)
06037.       if(!ktSongSong(a[i],a[0]))
06038.          return 0;
06039.    return 1;
06040. }
```

– Định nghĩa hàm (Phan Thị Hồng Cúc).

```
06041. int ktSongSong(DUONGTHANG a[], int n)
06042. {
06043.    for (int i = 0; i <=n-2; i++)
06044.       if(!ktSongSong(a[i],a[i+1]))
06045.          return 0;
06046.    return 1;
06047. }
```

Bài 209.Cho *n* đường thẳng. Hãy kiểm tra các đường thẳng này có cặp đường thẳng cùng song song với nhau không.

– Khai báo hàm.

```
06048.  int ktCapSongSong(DUONGTHANG [],int);
```

– Định nghĩa hàm.

```
06049.  int ktCapSongSong(DUONGTHANG a[], int n)
06050.  {
06051.     int flag = 0;
06052.     for (int i=0; i<=n-2; i++)
06053.        for (int j=i+1; j<=n-1; j++)
06054.           if(ktSongSong(a[i], a[j]))
06055.              flag = 1;
06056.     return flag;
06057.  }
```

Bài 210.Cho *n* đường thẳng. Hãy kiểm tra các đường thẳng này có cặp đường thẳng trùng nhau không.

– Khai báo hàm.

```
06058.  int ktCapTrungNhau(DUONGTHANG [],int);
```

– Định nghĩa hàm.

```
06059.  int ktCapTrungNhau(DUONGTHANG a[], int n)
06060.  {
06061.     int flag = 0;
06062.     for (int i=0; i<=n-2; i++)
06063.        for (int j=i+1; j<=n-1; j++)
06064.           if(ktTrung(a[i], a[j]))
06065.              flag = 1;
06066.     return flag;
06067.  }
```

Bài 211.Cho *n* đường thẳng và tọa độ một điểm *P*. Hãy kiểm tra xem có đường thẳng nào đi qua điểm *P* hay không.

– Khai báo hàm.

```
06068.  int ktQuaDiem(DUONGTHANG [], int, DIEM);
```

– Định nghĩa hàm.

```
06069.  int ktQuaDiem(DUONGTHANG a[], int n, DIEM P)
06070.  {
06071.     int flag = 0;
06072.     for (int i = 0; i < n; i++)
06073.        if (ktThuoc(a[i], P))
06074.           flag = 1;
06075.     return flag;
06076.  }
```

Bài 212.Cho *n* đường thẳng và tọa độ một điểm *P* không thuộc bất cứ đường thẳng nào. Hãy tìm một đường thẳng gần với điểm *P* nhất.

– Khai báo hàm.

```
06077.  DUONGTHANG GanDiemNhat(DUONGTHANG [],int,
06078.                                                DIEM);
```

– Định nghĩa hàm.

```
06079.  DUONGTHANG GanDiemNhat(DUONGTHANG a[],int n,
06080.                                                DIEM P)
06081.  {
06082.     DUONGTHANG lc = a[0];
06083.     for (int i=0; i<n; i++)
06084.        if(KhoangCach(a[i],P)<KhoangCach(lc,P))
06085.           lc = a[i];
06086.     return lc;
06087.  }
```

Bài 213.Cho *n* đường thẳng đôi một không trùng nhau. Hãy kiểm tra xem có ba đường thẳng nào đồng qui hay không.

– Khai báo hàm.

```
06088.  int ktDongQui(DUONGTHANG [],int);
```

– Định nghĩa hàm.

```
06089.  int ktDongQui(DUONGTHANG a[], int n)
06090.  {
06091.     for (int i=0; i<=n-2; i++)
06092.        for (int j=i+1; j<=n-1; j++)
06093.           if(ktCat(a[i],a[j]))
06094.           {
06095.              DIEM PP = GiaoDiem(a[i],a[j]);
06096.              for (int k=0; k<n; k++)
06097.                 if (k!=i && k!=j &&
06098.                                    ktThuoc(a[k],PP))
06099.                    return 1;
06100.           }
06101.     return 0;
06102.  }
```

01.09.06 Mảng một chiều các ngày

Bài 214.Viết hàm nhập mảng một chiều các ngày.

– Khai báo hàm.

```
06103. void Nhap(NGAY [],int &);
```

– Định nghĩa hàm.

```
06104. void Nhap(NGAY a[], int& n)
06105. {
06106.    cout << "Nhap n: ";
06107.    cin >> n;
06108.    for (int i = 0; i < n; i++)
06109.    {
06110.       cout << "Nhap duong thang A["<<i<<"]:\n";
06111.       Nhap(a[i]);
06112.    }
06113. }
```

Bài 215.Viết hàm xuất mảng một chiều các ngày.

– Khai báo hàm.

```
06114. void Xuat(NGAY [],int);
```

– Định nghĩa hàm.

```
06115. void Xuat(NGAY a[], int n)
06116. {
06117.    for (int i = 0; i < n; i++)
06118.    {
06119.       cout << "Ngay a[" << i << "]:";
06120.       Xuat(a[i]);
06121.       cout << endl;
06122.    }
06123. }
```

Bài 216.Viết hàm tìm ngày hai ngày gần nhau nhất.

– Khai báo hàm.

```
06124. void GanNhauNhat(NGAY [],int,NGAY&,NGAY &);
```

– Định nghĩa hàm.

```
06125. void GanNhauNhat(NGAY a[],int n,
06126.                     NGAY &x,NGAY &y)
06127. {
06128.    x = a[0];
06129.    y = a[1];
06130.    for(int i=0; i<=n-2; i++)
06131.       for(int j=i+1; j<=n-1; j++)
06132.          if(KhoangCach(a[i],a[j])<
06133.             KhoangCach(x,y))
06134.          {
```

```
06135.             x = a[i];
06136.             y = a[j];
06137.          }
06138.  }
```

Bài 217.Viết hàm tìm ngày hai ngày xa nhau nhất.

– Khai báo hàm.

```
06139.  void XaNhauNhat(NGAY [],int,NGAY&,NGAY &);
```

– Định nghĩa hàm.

```
06140.  void XaNhauNhat(NGAY a[],int n,
06141.                    NGAY &x,NGAY &y)
06142.  {
06143.     x = a[0];
06144.     y = a[1];
06145.     for(int i=0; i<=n-2; i++)
06146.        for(int j=i+1; j<=n-1; j++)
06147.           if(KhoangCach(a[i],a[j])>
06148.              KhoangCach(x,y))
06149.           {
06150.              x = a[i];
06151.              y = a[j];
06152.           }
06153.  }
```

01.10 MA TRẬN CẤU TRÚC

01.10.01 Ma trận tọa độ điểm

Bài 218.Viết hàm nhập ma trận tọa độ điểm trong mặt phẳng Oxy.

– Khai báo hàm.

```
06154.  void Nhap(DIEM [][100],int &,int &);
```

– Định nghĩa hàm.

```
06155.  void Nhap(DIEM a[][100],int &m,int &n)
06156.  {
06157.     cout << "Nhap so hang: ";
06158.     cin >> m;
06159.     cout << "Nhap so cot: ";
06160.     cin >> n;
06161.     for (int i = 0; i < m; i++)
06162.        for (int j = 0; j < n; j++)
```

```
06163.       {
06164.          cout << "Nhap a["<<i<<"]["<<j<< "]:\n";
06165.          Nhap(a[i][j]);
06166.       }
06167. }
```

Bài 219.Viết hàm xuất ma trận tọa độ điểm trong mặt phẳng Oxy.

– Khai báo hàm.

```
06168. void Xuat(DIEM [][100],int,int);
```

– Định nghĩa hàm.

```
06169. void Xuat(DIEM a[][100], int m, int n)
06170. {
06171.    for (int i = 0; i < m; i++)
06172.       for (int j = 0; j < n; j++)
06173.       {
06174.          cout << "Diem A["<<i<<"]["<<j<< "]:";
06175.          Xuat(a[i][j]);
06176.          cout << endl;
06177.       }
06178. }
```

Bài 220.Đếm số lượng điểm trong ma trận thuộc góc phần tư thứ 3 trong mặt phẳng tọa độ Oxy.

– Khai báo hàm.

```
06179. int DemThuoc3(DIEM [][100],int,int);
```

– Định nghĩa hàm.

```
06180. int DemThuoc3(DIEM a[][100], int m, int n)
06181. {
06182.    int dem = 0;
06183.    for (int i = 0; i < m; i++)
06184.       for (int j = 0; j < n; j++)
06185.          if (ktThuoc3(a[i][j]))
06186.             dem++;
06187.    return dem;
06188. }
```

Bài 221.Đếm tần suất xuất hiện của tọa độ một điểm trong ma trận.

– Khai báo hàm.

```
06189. int ktTrung (DIEM,DIEM);
06190. int TanSuat(DIEM [][100],int,int,DIEM);
```

– Định nghĩa hàm.

```
06191.  int ktTrung(DIEM P,DIEM Q)
06192.  {
06193.     if(P.x==Q.x && P.y==Q.y)
06194.        return 1;
06195.     return 0;
06196.  }
```

– Định nghĩa hàm.

```
06197.  int TanSuat(DIEM a[][100],int m,int n,DIEM P)
06198.  {
06199.     int dem = 0;
06200.     for (int i = 0; i < m; i++)
06201.        for (int j = 0; j < n; j++)
06202.           if(ktTrung(a[i][j],P))
06203.              dem++;
06204.     return dem;
06205.  }
```

Bài 222.Đếm số lượng điểm không trùng với điểm khác trong ma trận.

– Khai báo hàm.

```
06206.  int TanSuat(DIEM [][100],int,int,DIEM);
06207.  int DemKhongTrung(DIEM [][100],int,int);
```

– Định nghĩa hàm.

```
06208.  int TanSuat(DIEM a[][100],int m,int n,DIEM P)
06209.  {
06210.     int dem = 0;
06211.     for (int i = 0; i < m; i++)
06212.        for (int j = 0; j < n; j++)
06213.           if(ktTrung(a[i][j],P))
06214.              dem++;
06215.     return dem;
06216.  }
```

– Định nghĩa hàm.

```
06217.  int DemKhongTrung(DIEM a[][100],int m,int n)
06218.  {
06219.     int dem = 0;
06220.     for (int i = 0; i < m; i++)
06221.        for (int j = 0; j < n; j++)
06222.           if (TanSuat(a, m, n, a[i][j]) == 1)
06223.              dem++;
06224.     return dem;
06225.  }
```

Bài 223.Đếm số lượng điểm trong ma trận thuộc đường thẳng $ax + by + c = 0$.

– Khai báo hàm.

```
06226.  int ktThuoc(DUONGTHANG,DIEM);
06227.  int DemDiem(DIEM [][100],int,int,DUONGTHANG);
```

– Định nghĩa hàm.

```
06228.  int ktThuoc(DUONGTHANG d, DIEM P)
06229.  {
06230.     if (d.a * P.x + d.b * P.y + d.c == 0)
06231.        return 1;
06232.     return 0;
06233.  }
```

– Định nghĩa hàm.

```
06234.  int DemDiem(DIEM a[][100], int m, int n,
06235.                DUONGTHANG d)
06236.  {
06237.     int dem = 0;
06238.     for (int i = 0; i < m; i++)
06239.        for (int j = 0; j < n; j++)
06240.           if (ktThuoc(d, a[i][j]))
06241.              dem++;
06242.     return dem;
06243.  }
```

Bài 224.Đếm số lượng điểm trong ma trận nằm trong đường tròn $c(I, R)$ (DUONGTRON).

– Khai báo hàm.

```
06244.  int ktThuoc(DUONGTRON,DIEM);
06245.  int DemDiem(DIEM [][100],int,int,DUONGTRON);
```

– Định nghĩa hàm.

```
06246.  int ktThuoc(DUONGTRON c, DIEM P)
06247.  {
06248.     float kc = KhoangCach(c.I, P);
06249.     if (kc <= c.R)
06250.        return 1;
06251.     return 0;
06252.  }
```

– Định nghĩa hàm.

```
06253.  int DemDiem(DIEM a[][100],int m,int n,
06254.                DUONGTRON c)
```

```
06255. {
06256.    int dem = 0;
06257.    for (int i = 0; i < m; i++)
06258.       for (int j = 0; j < n; j++)
06259.          if (ktThuoc(c, a[i][j]))
06260.             dem++;
06261.    return dem;
06262. }
```

Bài 225.(*) Tìm một điểm trong ma trận gần đường thẳng $ax + by + c = 0$ (DUONGTHANG) nhất và không nằm trên đường thẳng này.

Bài 226.(*) Tìm điểm gần với tọa độ điểm $P(x, y)$ nhất trong ma trận (không trùng với điểm P).

01.10.02 Ma trận phân số

Bài 227.Viết hàm nhập ma trận các phân số.

– Khai báo hàm.

```
06263. void Nhap(PHANSO [][100],int &,int &);
```

– Định nghĩa hàm.

```
06264. void Nhap(PHANSO a[][100],int &m,int &n)
06265. {
06266.    cout << "Nhap so hang: ";
06267.    cin >> m;
06268.    cout << "Nhap so cot: ";
06269.    cin >> n;
06270.    for (int i = 0; i < m; i++)
06271.       for (int j = 0; j < n; j++)
06272.       {
06273.          cout << "Nhap a["<<i<<"]["<<j<< "]:\n";
06274.          Nhap(a[i][j]);
06275.       }
06276. }
```

Bài 228.Viết hàm xuất ma trận các phân số.

– Khai báo hàm.

```
06277. void Xuat(PHANSO [][100],int,int);
```

– Định nghĩa hàm.

```
06278. void Xuat(PHANSO a[][100], int m, int n)
```

```
06279. {
06280.    for (int i = 0; i < m; i++)
06281.       for (int j = 0; j < n; j++)
06282.       {
06283.          cout << "A["<<i<<"]["<<j<< "]:";
06284.          Xuat(a[i][j]);
06285.          cout << endl;
06286.       }
06287. }
```

Bài 229.Tìm phân số lớn nhất trong ma trận.

– Khai báo hàm.

```
06288. PHANSO LonNhat(PHANSO [][100],int,int);
```

– Định nghĩa hàm.

```
06289. PHANSO LonNhat(PHANSO a[][100],int m,int n)
06290. {
06291.    PHANSO lc = a[0][0];
06292.    for (int i=0; i<m; i++)
06293.       for (int j=0; j<n; j++)
06294.          if(SoSanh(a[i][j],lc)==1)
06295.             lc = a[i][j];
06296.    return lc;
06297. }
```

Bài 230.Đếm số lượng phân số nhỏ nhất trong ma trận.

– Khai báo hàm.

```
06298. PHANSO NhoNhat(PHANSO [][100],int,int);
06299. int DemNhoNhat(PHANSO [][100],int,int);
```

– Định nghĩa hàm.

```
06300. PHANSO NhoNhat(PHANSO a[][100],int m,int n)
06301. {
06302.    PHANSO lc = a[0][0];
06303.    for (int i=0; i<m; i++)
06304.       for (int j=0; j<n; j++)
06305.          if(SoSanh(a[i][j],lc)==-1)
06306.             lc = a[i][j];
06307.    return lc;
06308. }
```

– Định nghĩa hàm.

```
06309. int DemNhoNhat(PHANSO a[][100],int m,int n)
06310. {
```

```
06311.     PHANSO lc = NhoNhat(a,m,n);
06312.     int dem = 0;
06313.     for (int i=0; i<m; i++)
06314.        for (int j=0; j<n; j++)
06315.           if(SoSanh(a[i][j],lc)==0)
06316.              dem++;
06317.     return dem;
06318.  }
```

Bài 231.Tìm phân số âm lớn nhất trong ma trận.

– Khai báo hàm.

```
06319.  PHANSO AmLonNhat(PHANSO [][100],int,int);
```

– Định nghĩa hàm.

```
06320.  PHANSO AmLonNhat(PHANSO a[][100],int m,int n)
06321.  {
06322.     PHANSO temp = {0,1};
06323.     PHANSO lc = temp;
06324.     for (int i = 0; i<m; i++)
06325.        for (int j = 0; j<n; j++)
06326.           if(ktAm(a[i][j]))
06327.              if(SoSanh(lc,temp) == 0 ||
06328.                  (SoSanh(a[i][j],lc) == 1)))
06329.                 lc = a[i][j];
06330.     return lc;
06331.  }
```

Bài 232.Sắp xếp ma trận tăng dần từ trái sang phải và từ trên xuống dưới.

– Khai báo hàm.

```
06332.  void HoanVi(PHANSO &, PHANSO &);
06333.  void SapTang(PHANSO [][100],int,int);
```

– Định nghĩa hàm.

```
06334.  void HoanVi(PHANSO &x, PHANSO &y)
06335.  {
06336.     PHANSO temp = x;
06337.     x = y;
06338.     y = temp;
06339.  }
```

– Định nghĩa hàm.

```
06340.  void SapTang(PHANSO a[][100], int m, int n)
06341.  {
```

```
06342.    for (int i=0; i<=m*n-2; i++)
06343.       for (int j=i+1; j<=m*n-1; j++)
06344.          if(SoSanh(a[i/n][i%n],
06345.                    a[j/n][j%n])==1)
06346.             HoanVi(a[i/n][i%n],a[j/n][j%n]);
06347. }
```

Bài 233.Liệt kê các phân số tối giản trong ma trận theo thứ tự tăng dần.

– Khai báo hàm.

```
06348. int ktToiGian(PHANSO);
06349. void Xuat(PHANSO);
06350. void HoanVi(PHANSO &, PHANSO &);
06351. void SapTang(PHANSO [],int);
06352. void Xuat(PHANSO [],int);
06353.
06354. void LietKe(PHANSO [][100],int,int);
```

– Định nghĩa hàm.

```
06355. void LietKe(PHANSO a[][100], int m, int n)
06356. {
06357.    PHANSO b[1000];
06358.    int k = 0;
06359.    for (int i = 0; i < m; i++)
06360.       for (int j = 0; j < n; j++)
06361.          if (ktToiGian(a[i][j]))
06362.             b[k++] = a[i][j];
06363.    SapTang(b,k);
06364.    Xuat(b,k);
06365. }
```

01.10.03 Ma trận số phức

Bài 234.Viết hàm nhập ma trận số phức.

– Khai báo hàm.

```
06366. void Nhap(SOPHUC [][100],int &,int &);
```

– Định nghĩa hàm.

```
06367. void Nhap(SOPHUC a[][100],int &m,int &n)
06368. {
06369.    cout << "Nhap so hang: ";
06370.    cin >> m;
06371.    cout << "Nhap so cot: ";
06372.    cin >> n;
```

```
06373.    for (int i = 0; i < m; i++)
06374.       for (int j = 0; j < n; j++)
06375.       {
06376.          cout << "Nhap a["<<i<<"]["<<j<< "]:\n";
06377.          Nhap(a[i][j]);
06378.       }
06379. }
```

Bài 235.Viết hàm xuất ma trận số phức.

– Khai báo hàm.

```
06380. void Xuat(SOPHUC [][100],int,int);
```

– Định nghĩa hàm.

```
06381. void Xuat(SOPHUC a[][100], int m, int n)
06382. {
06383.    for (int i = 0; i < m; i++)
06384.       for (int j = 0; j < n; j++)
06385.       {
06386.          cout << "A["<<i<<"]["<<j<< "]:";
06387.          Xuat(a[i][j]);
06388.          cout << endl;
06389.       }
06390. }
```

Bài 236.Tìm số phức đầu tiên trong ma trận có phần thực dương và phần ảo dương.

– Khai báo hàm.

```
06391. SOPHUC ThucAoDuongDau(SOPHUC [][100],
06392.                                          int,int);
```

– Định nghĩa hàm.

```
06393. SOPHUC ThucAoDuongDau(SOPHUC a[][100],int m,
06394.                                              int n)
06395. {
06396.    for (int i = 0; i < m; i++)
06397.       for (int j = 0; j < n; j++)
06398.          if(a[i][j].Thuc>0 && a[i][j].Ao>0)
06399.             return a[i][j];
06400.    return {0,0};
06401. }
```

Bài 237.Tìm số phức cuối cùng trong ma trận có phần thực âm và phần ảo âm.

– Khai báo hàm.

```
06402.  SOPHUC ThucAoAmCuoi(SOPHUC [][100],
06403.                                          int,int);
```

– Định nghĩa hàm.

```
06404.  SOPHUC ThucAoAmCuoi(SOPHUC a[][100],int m,
06405.                                              int n)
06406.  {
06407.     for (int i = m-1; i >=0 ; i--)
06408.        for (int j = n-1; j >= 0; j--)
06409.           if(a[i][j].Thuc<0 && a[i][j].Ao<0)
06410.              return a[i][j];
06411.     return {0,0};
06412.  }
```

Bài 238.Đếm số lượng dòng trong ma trận có chứa toàn số phức thỏa điều kiện: "phần thực và phần ảo trái dấu nhau".

– Khai báo hàm.

```
06413.  int ktDong(SOPHUC [][100],int,int,int);
06414.  int DemDong(SOPHUC [][100],int,int);
```

– Định nghĩa hàm.

```
06415.  int ktDong(SOPHUC a[][100],int m,int n,int d)
06416.  {
06417.     int flag = 1;
06418.     for (int j = 0; j < n; j++)
06419.        if (a[d][j].Thuc * a[d][j].Ao >= 0)
06420.           flag = 0;
06421.     return flag;
06422.  }
```

– Định nghĩa hàm.

```
06423.  int DemDong(SOPHUC a[][100],int m,int n)
06424.  {
06425.     int dem = 0;
06426.     for (int i = 0; i < m; i++)
06427.        if (ktDong(a,m,n,i))
06428.           dem++;
06429.     return dem;
06430.  }
```

Bài 239.Tìm số phức có phần thực lớn nhất trong ma trận.

– Khai báo hàm.

```
06431.  SOPHUC ThucLonNhat(SOPHUC [][100],int,int);
```

– Định nghĩa hàm.

```
06432. SOPHUC ThucLonNhat(SOPHUC a[][100],int m,
06433.                                              int n)
06434. {
06435.    SOPHUC lc = a[0][0];
06436.    for (int i = 0; i < m; i++)
06437.       for (int j = 0; j < n; j++)
06438.          if (a[i][j].Thuc > lc.Thuc)
06439.             lc = a[i][j];
06440.    return lc;
06441. }
```