**BÀI THỰC TẬP 02: LẬP TRÌNH TẠO LỚP KẾT HỢP MẢNG ĐỐI TƯỢNG VÀ CON TRỎ ĐỐI TƯỢNG**

**A. MỤC TIÊU**

Trang bị cho sinh viên kỹ năng sử dụng mảng các đối tượng và con trỏ đối tượng để viết các chương trình có liên quan đến mảng các đối tượng trong C++.

1. **NỘI DUNG**

* Khai báo mảng đối tượng, con trỏ đối tượng;
* Viết chương trình sử dụng mảng đối tượng và con trỏ đối tương.

**C. YÊU CẦU PHẦN CỨNG, PHẦN MỀM**

* Phần cứng: Dung lượng RAM từ 1GB
* Phần mềm: Hệ điều hành Windows, C-free 5.0

**D. KẾT QUẢ SAU KHI HOÀN THÀNH**

Sinh viên thành thạo cách sử dụng mảng các đối tượng và con trỏ đối tượng để xây dựng chương trình có liên quan đến mảng đối tượng và con trỏ đối tượng bằng ngôn ngữ lập trình C++.

**E. TÓM TẮT LÝ THUYẾT**

**1. Khai báo con trỏ của một đối tượng thuộc lớp**

|  |
| --- |
| Khai báo con trỏ đối tượng:  *Tên\_lớp \*con\_trỏ ;*  Với lớp HCN có thể khai báo:  HCN \*p1, \*p2, \*p3; // khai báo 3 con trỏ p1, p2, p3  HCN h1, h2; // khai báo 2 đối tượng h1, h2  Khi đó có thể thực hiện các câu lệnh sau:  p1 = &h1; // p1 chứa địa chỉ của h1, hay p1 trỏ tới h1  p3 = new HCN; // Tạo 1 đối tượng và chứa địa chỉ của nó vào p3  Sử dụng thuộc tính của đối tượng thông qua con trỏ:  *Tên\_con\_trỏ -> Tên\_thuộc\_tính* |

1. **Khai báo mảng các đối tượng**

|  |
| --- |
| Khai báo mảng các đối tượng:  *tên\_lớp tên\_mảng\_đối\_tượng [số\_phần\_tử] ;*  Thuộc tính của mảng đối tượng  *Tên\_mảng\_đối\_tượng[chỉ\_số] . tên\_thuộc\_tính ;*  Thuộc tính của mảng đối tượng con trỏ  *Tên\_mảng\_đối\_tượng[chỉ\_số] -> Tên\_thuộc\_tính;*  Sử dụng các phương thức  *Tên\_mảng\_đối\_tượng[chỉ\_số] . tên\_phương\_thức (các\_đối\_số) ;*  Sử dụng các phương thức của mảng đối tượng con trỏ  *Tên\_mảng\_đối\_tượng[chỉ\_số] -> tên\_phương\_thức (các\_đối\_số) ;* |

**F. HƯỚNG DẪN CHI TIẾT**

**Bài 1:**

Tạo lớp hình chữ nhật HCN gồm có 2 thành phần dữ liệu là chiều dài d và chiều rộng r, các phương thức gồm khởi tạo, nhập, in, tính chu vi, tính diện tích của hình chữ nhật. Viết chương trình nhập vào kích thước của n hình chữ nhật và in ra mà hình thông tin:

* Hình chữ nhật có chu vi lớn nhất
* Hình chữ nhật có diện tích bé nhất

**Hướng dẫn:**

Khai báo mảng các đối tượng HCN (tối đa 50 đối tượng)

|  |
| --- |
| HCN H[50]; |

Hoặc khai báo 1 con trỏ đối tượng H:

|  |
| --- |
| HCN \*H;  Sau đó cấp phát cho H như sau:  H = new HCN[n]; |

Nhập dữ liệu cho n đối tượng:

|  |
| --- |
| for(i=0;i<n;i++)  H[i].nhap(); |

In thông tin cho n đối tượng:

|  |
| --- |
| for(i=0;i<n;i++)  H[i].inthongtin(); |

Tìm hình chữ nhật có chu vi lớn nhất

|  |
| --- |
| float Maxcv;  int chisocv;  Maxcv=H[0].tinhchuvi();  chisocv = 0;  for(i=1;i<n;i++)  if(Maxcv < H[i].tinhchuvi())  {  Maxcv = H[i].tinhchuvi();  chisocv = i;  }  //*Khi đó hình chữ nhật có chu vi lớn nhất là H[chisocv], //chu vi lớn nhất/là Maxcv.* |

Tìm hình chữ nhật có diện tích nhỏ nhất

|  |
| --- |
| float Mindt;  int chisodt;  Mindt=H[0].tinhdientich();  chisodt = 0;  for(i=1;i<n;i++)  if(Mindt > H[i].tinhdientich())  {  Mindt = H[i].tinhdientich();  chisodt = i;  }  //*Khi đó hình chữ nhật có diện tích nhỏ nhất là H[chisodt], //diện tích nhỏ nhất là Mindt.* |

Cài đặt chương trình như sau:

|  |
| --- |
| #include<iostream.h>  **class HCN**  {  **private:**  float d,r;  **public:**  void nhap() ;  void inthongtin();  float tinhchuvi();  float tinhdientich();  };  **void HCN :: nhap()**  {  cout<<"nhap chieu dai: "; cin>>d;  cout<<"nhap chieu rong: "; cin>>r;  }  **void HCN :: inthongtin()**  {  cout<<"("<<d<<","<<r<<")"<<" ";  }  **float HCN :: tinhchuvi()**  {  return 2\*(d+r);  }  **float HCN :: tinhdientich()**  {  return d\*r ;  }  **int main()**  {  HCN \*H;  int n,i;  cout<<"\nNhap so doi tuong n = ";  cin>>n;  H = new HCN[n];  cout<<"\nNhap thong tin hinh chu nhat:"<<endl;  for(i=0;i<n;i++)  H[i].nhap();  cout<<"\nIn thong tin hinh chu nhat: ";  for(i=0;i<n;i++)  H[i].inthongtin();  //Tim hình chu nhat co chu vi lon nhat  float Maxcv;  int chisocv;  Maxcv = H[0].tinhchuvi();  chisocv = 0;  for(i=1;i<n;i++)  if(Maxcv < H[i].tinhchuvi())  {  Maxcv = H[i].tinhchuvi();  chisocv = i;  }  cout<<"\n\nHinh ch nhat co chu vi lon nhat la: ";  H[chisocv].inthongtin();  cout<<" co chu vi la "<<Maxcv<<endl;  //Tim hình chu nhat co dien tich nho nhat  float Mindt;  int chisodt;  Mindt = H[0].tinhdientich();  chisodt = 0;  for(i=1;i<n;i++)  if(Mindt > H[i].tinhdientich())  {  Mindt = H[i].tinhdientich();  chisodt = i;  }  cout<<"\n\nHinh ch nhat co dien tich nho nhat la: ";  H[chisodt].inthongtin();  cout<<" co dien tich la "<<Mindt<<endl;  cout<<endl;  return 0;  } |

**Bài 2:**

Xây dựng lớp phân số PS gồm các thuộc tính tử số và mẫu số là các số nguyên. Trong lớp có định nghĩa các hàm sau:

* Hàm nhập.
* Hàm rút gọn phân số.
* Hàm hiển thị phân số ra màn hình ở dạng tử số / mẫu số.
* Hàm tính tổng 2 phân số.
* Hàm tính tích 2 phân số.

Viết chương nhập dữ liệu cho n phân số. In ra màn hình tổng, tích các phân số đó[2].

**Hướng dẫn:**

Khai báo mảng các đối tượng PS (tối đa 50 đối tượng)

|  |
| --- |
| PS H[50]; |

Hoặc khai báo 1 con trỏ đối tượng P:

|  |
| --- |
| PS \*P;  Sau đó cấp phát cho P như sau:  P = new PS[n]; |

Nhập và in thông tin cho n đối tượng PS p tương tự như bài 1

Tính tổng n phân số dựa vào phương thức tính tổng của 2 phân số như sau

|  |
| --- |
| PS KQ;  KQ = P[0];  for(int i=1;i<n;i++)  KQ = KQ.tong(P[i]); |

Sau khi tính tổng, rút gọn PS KQ rồi in ra màn hình.

Tính tích hoàn toàn tương tự tính tổng

**Bài 3: Viết chương trình thực hiện:**

Xây dựng lớp thí sinh TS có các thuộc tính họ tên, điểm thi Toán, điểm thi Văn. Các phương thức bao gốm: khởi tạo, hủy, nhập, in và các phương thức khác để thực hiện:

* Nhập một danh sách thí sinh lưu trữ trong một mảng
* Với một thí sinh được nhập từ bàn phím, hãy cho biết trong danh sách có thí sinh đó không, nếu có đưa ra điểm thi các môn của thí sinh đó, sau đó xóa thí sinh đó khỏi danh sách.
* Sắp xếp danh sách thí sinh sau khi xóa theo thứ tự tăng dần của họ tên.

**Hướng dẫn:**

Trong chương trình chính cần lưu ý:

Tạo 1 mảng đối tượng n TS, n nhập từ bàn phím:

|  |
| --- |
| TS \*A;  int n;  cout<<"Nhap so thi sinh: n = ";  cin>>n;  A = new TS[n]; |

Nhập danh sách n thí sinh, phương thức nhập nhap() đã được định nghĩa trước đó trong lớp.

|  |
| --- |
| for(i =0; i<n;i++)  { cout<<endl;  cout<<"Nhap nguoi thu "<<i+1<<endl;  A[i].nhap();  } |

Xuất danh sách n thí sinh được lưu trong mảng đối tượng A:

|  |
| --- |
| for(i=0;i<n;i++)  A[i].xuat(); |

Nhập họ tên thí sinh từ bàn phím để tìm kiếm:

|  |
| --- |
| TS tstk;  char HTTK[30];  cin.ignore(1);  cin.getline(HTTK,30); |

Kiểm tra thí sinh có họ tên lưu trong biến HTTK xem có trong danh sách thí sinh không, nếu có in ra thông tin thí sinh đó, nếu không tìm thấy in ra thông báo thí sinh đó không có trong danh sách.

Mô tả thuật toán:

* Sử dụng 1 biến KT có kiểu Bool để đánh dấu việc tìm kiếm, ban đầu chưa tìm được gán biến này bằng false.
* Duyệt trong danh sách n thí sinh, nếu có thí sinh có họ tên trùng với HTTK thì xuất thông tin thí sinh đó ra và đánh dấu biến KT bằng true.
* Kết thúc quá trình tìm kiếm nếu biến KT không thay đổi giá trị khởi tạo thì kết luận không tìm thấy thí sinh đó trong danh sách.

|  |
| --- |
| bool KT1=false;  for(i=0;i<n;i++)  if(strcmp(HTTK,A[i].getht())==0)  {  cout<<"\nThong tin Thi sinh can tim la: ";  A[i].xuat();  KT1=true;  }  if(!KT1)  cout<<"!!!!Khong co thi sinh nay trong danh sach!!!!"; |

Xóa thí sinh khỏi danh sách:

* Duyệt lần lượt từ thí sinh từ đầu đến hết danh sách
* Nếu thí sinh i nào có họ tên trùng với HTTK thì

+ Giảm số phần tử của mảng đi 1

+ Di chuyển các phần tử sau vị trí thứ i về trước 1 vị trí

|  |
| --- |
| for(i=0;i<n;i++)  if(strcmp(HTTK,A[i].getht())==0)  {  n--;  for(int k=i;k<n;k++)  A[k]=A[k+1];    } |

Sắp xếp danh sách theo thứ tự tăng dần của họ tên: Sử dụng 1 trong các thuật toán sắp xếp đã biết, ví dụ thuật toán sắp xếp lựa chọn.

|  |
| --- |
| for(i=0;i<n-1;i++)  for(j=i+1;j<n;j++)  if(strcmp(A[i].getht(),A[j].getht())>0)  doicho(A[i],A[j]); |

**Lưu ý:** Trong toàn bộ chương trình chính, các đối tương không thể truy xuất trực tiếp đến các thuộc tính private của lớp, vì vậy, việc truy xuất phải gián tiếp thông qua các phương thức public, cụ thể là phương thức getht().

Chương trình cài đặt như sau:

|  |
| --- |
| #include<iostream.h>  #include<iomanip.h>  **class TS**  {  **private:**  char ht[30];  double t,v;  **public:**  void nhap();  void xuat();  char \*getht()  {  return ht;  }  };  **void TS::nhap()**  {  cout<<"Nhap ho ten: ";  cin.ignore(1);  cin.getline(ht,30);  cout<<"nhap diem toan: ";  cin>>t;  cout<<"nhap diem van: ";  cin>>v;  }  **void TS::xuat()**  {  cout<<ht<<setw(10)<<t<<setw(10)<<v<<endl;  }  **void doicho(TS &X,TS &Y)**  {  TS TG;  TG=X;  X=Y;  Y=TG;  }  **int main()**  {  TS \*A;  int i,j,n;  cout<<"Nhap so thi sinh: n = ";  cin>>n;  A = new TS[n];  cout<<"\nNhap danh sách thí sinh: ";  for(i =0; i<n;i++)  { cout<<endl;  cout<<"Nhap nguoi thu "<<i+1<<endl;  A[i].nhap();  }  //Nhap ho ten thi sinh tim kiem  cout<<"\nNhap thi sinh tìm kiem: ";  char HTTK[30];  cin.ignore(1);  cin.getline(HTTK,30);  //KT thi sinh có trong DS khong neu có in thông tin,...  bool KT1=false;  for(i=0;i<n;i++)  if(strcmp(HTTK,A[i].getht())==0)  {  cout<<"\nThong tin Thi sinh can tim la: ";  A[i].xuat();  KT1=true;  }  if(!KT1)  cout<<"!!!!Khong co thi sinh nay trong danh sach!!!!";  // Xoa thi sinh khoi danh sach  for(i=0;i<n;i++)  if(strcmp(HTTK,A[i].getht())==0)  {  n--;  for(int k=i;k<n;k++)  A[k]=A[k+1];  }  cout<<"\nDanh sach thi sinh sau khi xoa là:"<<endl;  for(i=0;i<n;i++)  A[i].xuat();  // Sap xep thi sinh theo ten tang dan  for(i=0;i<n-1;i++)  for(j=i+1;j<n;j++)  if(strcmp(A[i].getht(),A[j].getht())>0)  doicho(A[i],A[j]);  //Xuat danh sach thi sinh sau khi sap xep  cout<<"\nDanh sach thi sinh sau khi sap xep là:"<<endl;  for(i=0;i<n;i++)  A[i].xuat();  return 0;  } |

**Bài 4: Viết chương trình thực hiện:**

Xây dựng lớp sinh viên SVien có 2 thuộc tính là đối tượng SV (SV có kiểu cấu trúc gồm số báo danh kiểu số nguyên, họ tên kiểu chuỗi, điểm toán, điểm lý, điểm hóa, tổng điểm kiểu float). Các phương thức gồm:

* Khởi tạo không tham số
* Khởi tạo có tham số
* Hủy
* Nhập thông tin cho mảng sinh viên
* In thông tin cho mảng sinh viên

Viết chương trình:

* Nhập vào n sinh viên (n nguyên dương nhập từ bàn phím).
* In danh sách sinh viên trúng tuyển biết rằng điểm trúng tuyển là 20.
* In ra thông tin sinh viên thủ khoa.

**Hướng dẫn:**

Tạo cấu trúc SV:

|  |
| --- |
| **struct SV**  {  int sobd;  char ht[30];  float t,l,h,td;  }; |

Tạo lớp có 2 thuộc tính theo yêu cầu:

|  |
| --- |
| **private:**  SV \*sv;  int sosv |

Phương thức khởi tạo không có tham số:

|  |
| --- |
| SVien()  {  sv = NULL;  sosv = 0;  } |

Phương thức khởi tạo có tham số:

|  |
| --- |
| **SVien(int n)**  {  sv = new SV[n+1];  sosv = n;  } |

Phương thức hủy: lớp có thành phần dữ liệu nên cần thiết xây dựng phương thức hủy tường minh để thu hồi vùng nhớ này.

|  |
| --- |
| **~SVien()**  {  if (sosv)  {  sosv = 0;  sv = NULL;  }  } |

Phương thức nhập: Bài này yêu cầu xây dựng phương thức nhập n sinh viên.

|  |
| --- |
| **void SVien::nhap()**  {  if(sosv)  for(int i =0; i<sosv;i++)  {  cout<<"Sinh vien "<<i+1<<":";  cout<<"\nSo bao danh: ";  cin>>sv[i].sobd;  cin.ignore();  cout<<"\nHo ten: ";  cin.get(sv[i].ht,30);  cout<<"\nDiem toan, ly, hoa: ";  cin>>sv[i].t>>sv[i].l>>sv[i].h;  sv[i].td=sv[i].t+sv[i].l+sv[i].h;  }  } |

Phương thức in: Bài này yêu cầu xây dựng phương thức in n sinh viên.

|  |
| --- |
| **void SVien::in()**  {  if(sosv)  {  cout<<"\nDanh sach sinh vien:"<<endl;  for(int i = 0; i<sosv;i++)  cout<<sv[i].ht<<setw(10)<<sv[i].sobd<<  setw(10)<<sv[i].td<<endl;  }  } |

Phương thức in danh sách sinh viên trúng tuyển: Có thể xây dựng phương thức in danh sách sinh viên trúng tuyển ngay trong lớp.

|  |
| --- |
| **void SVien::intrungtuyen()**  {  for(int i = 0; i<sosv;i++)  {  if(sv[i].td>=20)  cout<<sv[i].ht<<setw(10)<<sv[i].sobd<<setw(10)  <<sv[i].td<<endl;  }  } |

Phương thức in thông tin sinh viên thủ khoa:

|  |
| --- |
| **void SVien::inthukhoa()**  {  float Maxtd;  Maxtd = sv[0].td;  int chiso = 0;  for(int i = 1; i<sosv;i++)  if(Maxtd<sv[i].td)  {  Maxtd = sv[i].td;  chiso = i;  }  cout<<sv[chiso].ht<<setw(10)<<sv[chiso].sobd  <<setw(10)<<sv[chiso].td<<endl;  } |

Chương trình chính:

|  |
| --- |
| **int main()**  {  int n;  cout<<"Nhap so sinh vien: ";  cin>>n;  SVien \*s = new SVien(n);  s->nhap();  cout<<"Danh sach sinh vien nhap la:"<<endl;  s->in();  cout<<"Danh sach sinh vien trung tuyen la:"<<endl;  s->intrungtuyen();  cout<<"Thong tin sinh vien do thu khoa là: ";  s->inthukhoa();  delete s;  cout<<endl;  return 0;  } |

**Mở rộng:** Viết các chương trình có yêu cầu tương tự đối với các lớp như lớp Khách hàng, lớp Công nhân, lớp Ô tô, Lớp Con người…

**Bài 5:**

Xây dựng lớp con người NGUOI gồm các thuộc tính: họ tên, tuổi (kiểu số nguyên), nghề nghiệp, quốc tịch. Trong lớp định nghĩa các phương thức sau:

* Khởi tạo không tham số
* Khởi tạo có tham số
* Hủy
* Nhập
* In thông tin

Viết chương trình nhập danh sách n ngưởi (n nguyên dương nhập từ bàn phím). Và in ra màn hình:

* Danh sách theo thứ tự tăng dần của tên
* Danh sách những người có nghề nghiệp là bác sỹ
* Tìm trong danh sách người có quốc tịch “Campuchia”, nếu có xóa người này khỏi danh sách
* Tìm trong danh sách có người có họ tên “Le Đai Nghia” không? Nếu không có chèn người này vào cuối danh sách.
* In ra thông tin người có tuổi cao nhất danh sách và người có tuổi nhỏ nhất trong danh sách.

**Hướng dẫn:**

* Tạo lớp NGUOI gồm các thuộc tính theo yêu cầu
* Các phương thức khởi tạo, hủy, nhập thông tin 1 người, in thông tin 1 người tương tự các bài trên.
* Sắp xếp danh sách theo thứ tự tăng dần của tên: Sử dụng 1 trong các thuật toán sắp xếp, lưu ý, khi so sánh chuỗi họ tên phải dùng hàm **strcmp(...)** để so sánh họ tên. Sau khi sắp xếp in danh sách bằng cách dùng vòng lặp for rồi gọi phương thức in 1 đối tượng đã định nghĩa trong lớp.
* Tìm trong danh sách những người có nghề nghiệp là “bac sy”. Nguyên tắc tìm kiếm như sau:

+ Duyệt lần lượt các phần tử trong mảng

+ So sánh thuộc tính nghề nghiệp của mỗi đối tượng với chuỗi “bac sy”. Nếu kết quả so sánh chuỗi bằng 0 thì in ra thông tin của đối tượng đó (bằng cách gọi phương thức in 1 đối tượng đã định nghĩa trong lớp)

+ Kết thúc vòng lặp sẽ in ra danh sách các đối tượng có nghề nghiệp là bác sỹ nếu có.

* Tìm và xóa trong danh sách người có quốc tịch “Campuchia”:

+ Duyệt lần lượt các phần tử trong mảng

+ So sánh thuộc tính quốc tịch của mỗi đối tượng với chuỗi “Campuchia”. Nếu kết quả so sánh chuỗi bằng 0 thì thực hiện xóa đối tượng đó khỏi danh sách.

+ Xóa đối tượng có vị trí i trong mảng như sau:

* Dịch chuyển các đối tượng từ vị trí (i+1) lùi về trước 1 chỗ
* Giảm số phần tử của mảng đi 1
* Tìm trong danh sách có người có họ tên “Le Đai Nghia” không? Nếu không có chèn người này vào cuối danh sách:

+ Duyệt lần lượt các phần tử trong mảng

+ Nếu có người có họ tên “Le Đai Nghia” thì kết thúc thuật toán, không cần chèn.

+ Nếu tìm đến hết danh sách vẫn không có thì chèn đối tượng vào cuối danh sách. Việc tìm kiếm nên sử dụng 1 biến kiểm tra có kiểu Bool, ban đầu gán biến này bằng false, nếu tìm thấy gán biến này bằng true. Hết vòng lặp sẽ kiểm tra biến này để biết có hay không có đối tượng có tên cần tìm.

+ Việc chèn đối tượng này vào cuối danh sách thực hiện như sau:

* Tăng số phần tử của mảng lên 1
* Gán lần lượt các thuộc tính họ tên, tuổi, nghề nghiệp, quốc tịch của đối tượng cuối cùng trong mảng bằng giá trị như yêu cầu trong đó họ tên là “Le Đai Nghia”.
* In ra thông tin người có tuổi cao nhất danh sách

+ Coi người có tuổi cao nhất là người đầu tiên, tức là Maxtuoi=BiếnMảng[0], chiso=0;

+ Duyệt lần lượt các phần tử trong mảng, nếu có phần tử thứ i nào đó có tuổi lớn hơn Maxtuoi thì:

Maxtuoi = BiếnMảng[i];

Chiso = i;

+ Kết thúc vòng lặp, người có chỉ số là chiso có tuổi cao nhất và in ra thông tin người đó (bằng cách gọi phương thức in() đã định nghĩa trong lớp)

* In ra thông tin người có tuổi nhỏ nhất trong danh sách. (Tương tự như tìm người có tuổi cao nhất).

**Mở rộng**:

* Đây là chương trình tổng hợp các thuật toán tìm kiếm và sắp xếp. Sinh viên có thể sử dụng linh hoạt và đa dạng các thuật toán tìm kiếm và sắp xếp để viết chương trình sao cho phù hợp và hiệu quả nhất.
* Cùng các yêu cầu tương tự, sinh viết vận dụng viết các chương trình thực tế với các lớp đa dạng như lớp ô tô, lớp sách, lớp hàng hóa, lớp động vật,...