**BÀI THỰC TẬP 01: LẬP TRÌNH TẠO LỚP, ĐỐI TƯỢNG CƠ BẢN VÀ CON TRỎ THIS**

**A. MỤC TIÊU**

Trang bị cho sinh viên kỹ năng lập trình tạo lớp cơ bản trong lập trình hướng đối tượng bằng C++: Khai báo lớp, khai báo thuộc tính, định nghĩa và sử dụng hàm tạo, hàm hủy, con trỏ this, khai báo đối tượng và xây dựng một chương trình hướng đối tượng cơ bản.

**B. NỘI DUNG**

* Khai báo lớp, thuộc tính, hàm tạo, hàm hủy, sử dụng con trỏ this;
* Khai báo và sử dụng đối tượng hoàn thiện chương trình.

**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 tạo và sử dụng lớp để xây dựng chương trình hướng đối tượng cơ bản bằng ngôn ngữ lập trình C++.

**E. TÓM TẮT LÝ THUYẾT**

1. **Khai báo lớp cơ bản.**

* Cú pháp định nghĩa lớp:

|  |
| --- |
| class <tên\_lớp>  {  [phạm\_vi\_truy\_xuất:]  <khai\_báo\_các\_thành\_phần\_dữ\_liệu(thuộc\_tính)>  [phạm\_vi\_truy\_xuất:]  <khai\_báo\_các\_hàm\_thành\_phần(phương\_thức)>  }; |

Khai báo phương thức trong lớp và định nghĩa phương thức ngoài lớp.

Khai báo phương thức trong lớp:

|  |
| --- |
| <kiểu\_trả\_về> <tên\_phương\_thức> ([đối\_số]) ; |

Định nghĩa phương thức ngoài lớp:

|  |
| --- |
| **<kiểu\_trả\_về> tên\_lớp :: <tên\_phương\_thức> ([đối\_số])**  **{**  // <thân phương thức>  **}** |

1. **Khai báo thuộc tính cơ bản**

|  |
| --- |
| **<kiểu\_dữ\_liệu > <tên\_thành\_phần>;** |

1. **Khai báo hàm tạo hàm hủy**

**Hàm tạo:**

|  |
| --- |
| * Cú pháp:   <tên\_lớp> ([ds\_tham\_số]);   * Định nghĩa hàm khởi tạo bên ngoài khai báo lớp như sau:   <tên\_lớp> :: <tên\_lớp> ([ds tham số])  {  //thân hàm  } |

**Hàm hủy:**

|  |
| --- |
| Hàm hủy được viết theo quy tắc sau:  ~ <tên\_lớp> (); |

1. **Xây dựng hàm tạo hàm hủy**

**Hàm tạo:**

|  |
| --- |
| Định nghĩa hàm khởi tạo bên ngoài khai báo lớp như sau:  **<tên\_lớp> :: <tên\_lớp> ([ds\_tham\_số])**  **{**  // Gán các thuộc tính bằng các tham số tương ứng  **}** |

**Hàm hủy:**

|  |
| --- |
| Hàm hủy được định nghĩa ngoài khai báo lớp như sau:  ~ <tên\_lớp> :: ~ <tên\_lớp> ()  {  //thu hồi vùng nhớ cho các đối tượng dữ liệu động nếu có  } |

1. **Khai báo đối tượng thuộc một lớp**

Khai báo đối tượng:

|  |
| --- |
| tên\_lớp tên\_biến\_đối\_tượng; |

Sử dụng đối tượng:

|  |
| --- |
| tên\_biến\_đối\_tượng . tên\_thành\_phần\_dữ\_liệu  tên\_biến\_đối\_tượng . tên\_phương\_thức() |

1. **Khai báo thuộc tính cơ bản dùng con trỏ this.**

|  |
| --- |
| Phương thức **nhap()** của lớp **HCN**  **void HCN :: nhap()**  {  cout<<"nhap chieu dai: "; **cin>>d;**  cout<<"nhap chieu rong: "; **cin>>r;**  }  Có thể viết một cách tường minh như sau:  **void HCN :: nhap()**  {  cout<<"nhap chieu dai: "; **cin>>this->d;**  cout<<"nhap chieu rong: "; **cin>>this->r;**  }  C++ cho phép sử dụng con trỏ đặc biệt this trong các phương thức. Các thuộc tính viết trong phương thức được hiểu là thuộc một đối tượng do con trỏ this trỏ tới  Lời gọi tới phương thức **nhap()** trong lớp **HCN**:  HCN h1;  h1.nhap();  Tham số truyền cho con trỏ **this** chính là **địa chỉ của h1:**  **this = &h1**  Và:  **this -> d** chính là **h1.d**  **this -> r** chính là **h1.r**  Câu lệnh **h1.nhap();** sẽ nhập dữ liệu cho các thuộc tính của h1.  Đối thứ nhất của phương thức bao giờ cũng là con trỏ **this** |

1. **Hoàn thiện chương trình đơn giản**

Cấu trúc một chương trình hướng đối tượng trong C++

|  |
| --- |
| * *Khai báo các thư viện dùng trong chương trình* * *Khai báo các hằng, các biến toàn cục (nếu có)* * *Khai báo, định nghĩa các kiểu dữ liệu (nếu có)* * ***Khai báo, định nghĩa các lớp*** * *Định nghĩa các hàm tự do, các phương thức (nếu có)* * ***Chương trình chính*** * *Định nghĩa các hàm tự do(nhưng phải khai báo hàm đó trước hàm main) (nếu có)* * *Định nghĩa các phương thức của lớp (nếu có)* |

**F. HƯỚNG DẪN CHI TIẾT**

**Bài 1**: Định nghĩa lớp hình chữ nhật HCN gồm có 2 thành phần dữ liệu là chiều dài d và chiều rộng r, các phương thức gồm nhập, in, tính chu vi, tính diện tích của hình chữ nhật. Viết chương trình nhập vào kích thước một hình chữ nhật, in ra màn hình chu vi, diện tích hình chữ nhật đó.

**Hướng dẫn:**

Xây dựng lớp hình chữ nhật

Cách 1: Định nghĩa các phương thức ngay trong lớp

|  |
| --- |
| **class HCN**  **{**  **private:**  float **d,r**;  **public:**  **void nhap()**  {  cout<<“nhap chieu dai: ”; cin>>d;  cout<<“nhap chieu rong: ”; cin>>r;  }  **void inthongtin()**  {  cout<<”(”<<d<<”,”<<r<<”)”;  }  **float tinhchuvi()**  {  return 2\*(d+r);  }  **float tinhdientich()**  {  return d\*r ;  }  **};** |

Cách 2: Khai báo các phương thức trong lớp và định nghĩa ngoài lớp

|  |
| --- |
| **class HCN**  **{**  **private:**  float **d,r**;  **public:**  **void nhap() ;**  **void inthongtin(HCN h);**  **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 ;  } |

Viết chương trình chính:

|  |
| --- |
| int main()  {  **HCN h;**  cout<<"Nhap thong tin hinh chu nhat:"<<endl;  **h.nhap();**  cout<<"In thong tin hinh chu nhat: ";  **h.inthongtin();**  cout<<”co chu vi la :"<<**h.tinhchuvi()**<<” va co dien tich la :"<<**h.tinhdientich()**;  return 0;  } |

Hoàn thiện chương trình như sau:

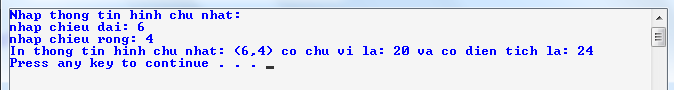
Cách 1:

|  |
| --- |
| **#include<iostream.h>**  **class HCN**  {  **private:**  float d,r;  **public:**  **void nhap()**  {  cout<<"nhap chieu dai: "; cin>>d;  cout<<"nhap chieu rong: "; cin>>r;  }  **void inthongtin()**  {  cout<<"("<<d<<","<<r<<")";  }  **float tinhchuvi()**  {  return 2\*(d+r);  }  **float tinhdientich()**  {  return d\*r ;  }  };  **int main()**  {  HCN h;  cout<<"Nhap thong tin hinh chu nhat:"<<endl;  h.nhap();  cout<<"In thong tin hinh chu nhat: ";  h.inthongtin();  cout<<" co chu vi la: "<<h.tinhchuvi()<<" va co dien tich la: "<<h.tinhdientich();  cout<<endl;  return 0;  } |

Cách 2: Định nghĩa phương thức ngoài lớp 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;  cout<<"Nhap thong tin hinh chu nhat:"<<endl;  h.nhap();  cout<<"In thong tin hinh chu nhat: ";  h.inthongtin();  cout<<" co chu vi la: "<<h.tinhchuvi()<<" va co dien tich la: "<<h.tinhdientich();  cout<<endl;  return 0;  } |

**Lưu bài, dịch chương trình, nếu có lỗi phải sửa lỗi, nếu chương trình chạy đúng sẽ cho kết quả như sau (**trong đó nhập chiều dài 6 và chiều rộng 4 từ bàn phím):

****

**Mở rộng:** Viết hàm khởi tạo cho lớp HCN:

**Hàm tạo không có tham số:**

|  |
| --- |
| **HCN()**{  d = 0;  r = 0;  } |

Hàm tạo có 2 tham số:

|  |
| --- |
| **HCN(float dx, float rx)**  {  d = dx;  r = rx;  } |

Hàm tạo 2 tham số có 1 tham số có giá trị ngầm định bằng 0

|  |
| --- |
| **HCN(float dx,float rx=0)**  {  d = dx;  r = rx;  } |

Hàm tạo 2 tham số có có giá trị ngầm định bằng 0

|  |
| --- |
| **HCN(float dx=0,float rx=0)**  {  d = dx;  r = rx;  } |

Khởi tạo các thuộc tính trong chương trình chính:

|  |
| --- |
| **int main()**  {  **HCN h1();**  **HCN h2(10,6);**  **HCN h3(5);**  cout<<"In thong tin hinh chu nhat h1: ";  **h1.inthongtin();**  cout<<”\nco chu vi la :"<<**h1.tinhchuvi()**<<” va co dien tich la :"<<**h1.tinhdientich()** ;  cout<<"In thong tin hinh chu nhat h2: ";  **h2.inthongtin();**  cout<<”\nco chu vi la :"<<**h2.tinhchuvi()**<<” va co dien tich la :"<<**h2.tinhdientich()** ;  cout<<"In thong tin hinh chu nhat h3: ";  **h3.inthongtin();**  cout<<”\nco chu vi la :"<<**h3.tinhchuvi()**<<” va co dien tich la :"<<**h3.tinhdientich()** ;  **//…**  } |

**Bài 2:**

Xây dựng lớp đa thức bậc 2 DaThuc2 gồm các thuộc tính hệ số, bậc 2, bậc 1 và bậc 0 lần lượt là a, b, c (a, b, c nguyên). Trong lớp có định nghĩa sau:

* Hàm khởi tạo với các tham số có giá trị ngầm định bằng 0
* Hàm hủy
* Hàm hiển thị đa thức ra màn hình ở dạng (ax2 + bx + c)
* Hàm tính tổng 2 đa thức

Viết chương trình tạo 2 đa thức và in ra màn hình tổng 2 đa thức đó[2].

**Hướng dẫn:**

Xây dựng lớp đa thức bậc 2 DaThuc2:

|  |
| --- |
| **class DaThuc2**  {  **private**:  int a, b, c;  **public**:  **DaThuc2 (int da=0,int db=0,int dc=0)**  {  a=da; b=db; c= dc;  }  **~ DaThuc2 (){}**  **void in();**  **DaThuc2 tong(DaThuc2 dt1);**  }; |

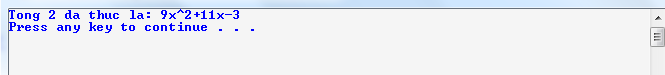
Chương trình chính:

|  |
| --- |
| **int main()**  **{**  **DaThuc2 dt1**(3,4,5);  **DaThuc2 dt2**(6,7,-8);  **DaThuc2 tong2**;  **tong2=dt1.tong(dt2);**  cout<<"Tong 2 da thuc la: ";  **tong2.in();**  cout<<endl;  return 1;  **}** |

**Hoàn thiện chương trình như sau:**

|  |
| --- |
| #include<iostream.h>  class DaThuc2  {  private:  int a, b, c;  public:  DaThuc2 (int da=0,int db=0,int dc=0)  {  a=da; b=db; c= dc;  }  ~ DaThuc2 ()  {  }  void in();  DaThuc2 tong(DaThuc2 dt1);  };  void DaThuc2::in()  {  cout<<a<<"x^2";  if (b>=0)  cout<<"+"<<b<<"x";  else  cout<<"-"<<b<<"x";  if (c>=0)  cout<<"+"<<c;  else  cout<<c;  }  DaThuc2 DaThuc2::tong(DaThuc2 dt)  {  DaThuc2 t;  t.a=a+dt.a;  t.b=b+dt.b;  t.c=c+dt.c;  return t;  }  **int main()**  **{**  **DaThuc2 dt1**(3,4,5);  **DaThuc2 dt2**(6,7,-8);  **DaThuc2 tong2**;  **tong2=dt1.tong(dt2);**  cout<<"Tong 2 da thuc la: ";  **tong2.in();**  cout<<endl;  return 1;  **}** |

**Kết quả chạy đúng chương trình như sau:**

****

**Bài 3:**

Xây dựng lớp số phức SP có 2 thuộc tính là phần thực và phần ảo có kiểu số nguyên. Trong lớp SP có các phương thức sau:

* Khởi tạo
* Hủy
* In số phức ở dạng phần thực + phần ảo\*j
* Tính tổng 2 số phức

Viết chương trình tạo 3 số phức và in ra màn hình tổng 3 số phức đó.

**Hướng dẫn:**

Xây dựng lớp SP:

|  |
| --- |
| **class SP**  **{**  **private:**  int **pt**;  int **pa**;  **public:**  **SP**();  **SP**(int **pt**,int **pa**);  void **in**();  SP **tong2**(SP p);  **};** |

Định nghĩa phương thức trong lớp sử dụng con trỏ **this**

*Phương thức khởi tạo không tham số:*

|  |
| --- |
| SP::SP()  **{**  **this->pt** = 0;  **this->pa** = 0;  **}** |

*Phương thức khởi tạo có tham số:*

|  |
| --- |
| SP::SP(int pt,int pa)  {  **this->pt =** pt**;**  **this->pa =** pa**;**  } |

*Phương thức in:*

|  |
| --- |
| void SP::in()  {  if(**this->pa**>=0)  cout<<**this->pt**<<" + "<<**this->pa**<<"\*j";  else  cout<<**this->pt**<<"-"<<**abs(this->pa**)<<"\*j";  } |

*Phương thức tính tổng 2 số phức:*

|  |
| --- |
| SP SP::tong2(SP p)  {  SP tong;  tong.pt=**this->pt**+p.pt;  tong.pa=**this->pa**+p.pa;  return tong;  } |

*Chương trình hoàn thiện như sau:*

|  |
| --- |
| #include<iostream.h>  **class SP**  **{**  **private:**  int **pt;**  int **pa;**  **public:**  **SP();**  **SP(**int **pt,**int **pa);**  void **in();**  SP **tong2(**SP p**);**  **};**  SP::SP()  {  **this->pt**=0**;**  **this->pa**=0**;**  }  SP::SP(int pt,int pa)  {  **this->pt**=pt**;**  **this->pa**=pa**;**  }  void SP::in()  {  if(**this->pa**>=0)  cout<<" "<<**this->pt**<<" + "<<**this->pa**<<"\*j";  else  cout<<" "<<**this->pt**<<" - "<<abs(**this->pa**)<<" \* j ";  }  SP SP::tong2(SP p)  {  SP tong;  tong.pt=**this->pt**+p.pt;  tong.pa=**this->pa**+p.pa;  return tong;  }  **int main()**  **{**  SP **p1**(5,-8);  SP **p2**(-7,3);  SP **p3**(6,-9);  SP **p4**,**p5**;  **p4=p1.tong2(p2);**  **p5=p3.tong2(p4);**  cout<<"Tong 3 so phuc vua khoi tao la:";  **p5.in();**  cout<<endl;  return 0;  **}** |

**Mở rộng:** Viết hàm nhập dữ liệu cho lớp số phức để có thể nhập dữ liệu từ bàn phím

**Bài 4:**

Xây dựng lớp vector 3 chiều VecTor3 gồm các thuộc tính tọa độ a, b, c (a, b, c nguyên). Trong lớp có định nghĩa các hàm sau:

* Hàm khởi tạo với các tham số có giá trị ngầm định bằng 0
* Hàm hủy
* Hàm nhập thông tin cho 1 vector
* Hàm hiển thị tọa độ vector ra màn hình ở dạng (a, b, c)
* Hàm tính tổng 2 vector

Viết chương trình tạo 2 vector và in ra màn hình tổng 2 vector đó[1].

**Hướng dẫn**: *Bài 4 làm tương tự như bài số 3, có thêm hàm nhập thông tin cho một* vector*, trong chương trình chính có thể tạo 2 vector bằng hàm khởi tạo hoặc nhập trực tiếp từ bàn phím.*

**Bài 5:**

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 khởi tạo các tham số có giá trị ngầm định
* Hàm hủy
* Hàm rút gọn phân số

Hàm hiển thị phân số ra màn hình ở dạng tử số / mẫu số

Viết chương trình khởi tạo 2 phân số. In ra màn hình tổng, hiệu 2 phân số đó[2].

**Hướng dẫn:**

Xây dựng lớp phân số:

|  |
| --- |
| **class phanso**  **{**  **private**:  float **tuso**;  float **mauso**;  **public**:  **phanso(float ts=0, float ms=1);**  **~phanso()**{}  void **nhap();**  void **in();**  phanso **tong**(phanso p1);  float **Gettuso**();  float **Getmauso**();  //…  **};** |

Trong đó phương thức nhập lưu ý kiểm tra để nhập mẫu khác 0 như sau:

|  |
| --- |
| **void nhap()**  **{**  cout<<"-Nhap tu so = ";  **cin>>tuso;**  **do**  **{**  cout<<"-Nhap mau so = ";  **cin>>mauso;**  **if(mauso==0)**  cout<<"mau so sai dinh dang !!! :) \n";  **}while(mauso==0);**  **}** |

Phương thức tính tổng lưu ý khi rút gọn phân số: Do tuso và mauso là 2 thuộc tính **private** nên hàm tìm ước số chung lớn nhất của 2 số là 1 hàm tự do ngoài lớp không thể truy cập đến. Để đảm bảo truy cập hoặc là khai báo tuso, mauso là public (nhưng như vậy làm mất đi tính an toàn cho dữ liệu hệ thống) hoặc là định nghĩa phương thức gettuso(), getmauso() trong lớp để thông qua 2 phương thức này truy cập đến tuso và maso như sau:

|  |
| --- |
| **phanso tong(phanso p1)**  **{**  phanso **p**;  float k;  p.tuso= (tuso\*p1.mauso)+(mauso\*p1.tuso);  p.mauso=mauso\*p1.mauso;  **k=UCLN(p.Gettuso(),p.Getmauso());**  **p.tuso=p.tuso/k;**  **p.mauso=p.mauso/k;**  **return p;**  **}**  **float Gettuso()**  {  return **tuso**;  }  **float Getmauso()**  {  return **mauso**;  } |

Hàm tự do tìm ước chung lớn nhất của 2 số thực cần khai báo trước định nghĩa lớp, và định nghĩa hàm tìm ước chung lớn nhất bằng thuật toán Euclid như sau::

|  |
| --- |
| **float UCLN(float a, float b)**  {  a = abs(a); b = abs(b); //Cần khai báo thư viện <math.h>  **while** (**a != b**)  {  **if** (a > b)  { **a = a-b;** }  **else**  { **b = b-a;** }  }  **return a;**  } |

Chương trình chính có thể tạo 2 đối tượng từ hàm khởi tạo, hoặc nhập 2 phân số trực tiếp từ bàn phím, ví dụ tạo 2 phân số từ hàm khởi tạo như sau:

|  |
| --- |
| **int main()**  **{**  **phanso p1(1,3);**  **phanso p2(2,5);**  phanso tong;  **tong=p1.tong(p1)**;  cout<<"Tong 2 phan so la: ";  **tong.in();**  return 0;  **}**  *Tương tự viết phương thức tính hiệu, tích, thương cho 2 phân số.* |

Nhập 2 phân số trực tiếp từ bàn phím như sau:

|  |
| --- |
| **int main()**  **{**  phanso p1, p2, tong;  **p1.nhap();**  **p2.nhap();**  **tong=p1.tong(p1);**  cout<<"Tong 2 phan so da nhap la: ";  **tong.in();**  return 0;  **}** |

**Mở rộng:** Tương tự viết phương thức tính hiệu, tích, thương cho 2 phân số.

**Bài 6:**

Xây dựng lớp công nhân CN gồm các thuộc tính họ tên ht (kiểu char), tuổi t (kiểu số nguyên), quê quán (kiểu char), bậc lương bl (kiểu float). Trong lớp có định nghĩa các hàm sau:

* Hàm khởi tạo các thông tin trên cho 1 công nhân
* Hàm xuất các thông tin trên của công nhân ra màn hình
* Hàm tính tiền lương cho công nhân biết rằng lương = bậc lương\*2500 + phụ cấp (tiền phụ cấp công nhân là 20% lương )

Viết chương trình nhập vào thông tin 1 công nhân. In ra màn hình tiền lương công nhân đó.

**Hướng dẫn:**

Các thuộc tính được khai báo như sau:

|  |
| --- |
| **private:**  char **\*ht;**  int **tuoi;**  char **\*qq;**  float **bl;** |

Phương thức khởi tạo: trong trường hợp đặt trùng tên các tham số hàm khởi tạo trùng tên thuộc tính bắt buộc phải sử dụng con trỏ this tường minh như sau:

|  |
| --- |
| **CN(**char **\*ht,**int **tuoi,**char **\*qq,**float **bl)**  **{**  **this->ht =** ht**;**  **this->tuoi =** tuoi**;**  **this->qq =** qq**;**  **this->bl =** bl**;**  **}** |

Nếu không sử dụng con trỏ this phải đặt tên các tham số khác tên thuộc tính, nên cũng có thể viết như sau

|  |
| --- |
| **CN(**char **\*hten,**int **t,**char **\*quequan,**float **bluong)**  **{**  **ht =** hten**;**  **tuoi =** t**;**  **qq =** quequan**;**  **bl =** bluong**;**  **}** |

Phương thức in sử dụng con trỏ this tường minh, và trong phương thức này có sử dụng phương thức ***this->tinhtl()***

|  |
| --- |
| **void CN::in()**  **{**  cout<<**this->ht**<<"\t"<<**this->tuoi**<<"\t"<<**this->qq**  <<"\t"<<**this->bl**<<"\t"<<**this->tinhtl()**;  cout<<endl;  **}** |

**Bài 7:**

Xây dựng lớp giáo viên GV gồm các thuộc tính họ tên ht (kiểu char[30]), tuổi t (kiểu số nguyên), bằng cấp cao nhất được đào tạo bc (kiểu char[15]), chuyên ngành (kiểu char[20]), bậc lương bl (kiểu float). Trong lớp có định nghĩa các hàm sau:

* Hàm nhập dữ liệu cho các thông tin trên của 1 giáo viên
* Hàm xuất các thông tin trên của giáo viên ra màn hình
* Hàm tính tiền lương biết rằng lương = bl \*1490

Viết chương trình nhập vào thông tin 1giáo viên. In ra màn hình tiền lương của giáo viên đó.

**Hướng dẫn:**

Xây dựng lớp GV như sau:

|  |
| --- |
| **class GV**  **{**  **private:**  char **ht**[30];  int **tuoi**;  char **bc**[15];  char **cn**[15];  float **bl**;  **public:**  void **nhap();**  void **in();**  float **tinhtl();**  **};** |

Định nghĩa phương thức nhập cần lưu ý nhập chuỗi có khoảng cách bằng lệnh cin.getline và luồng fflush(stdin);

|  |
| --- |
| **void GV:: nhap()**  **{**  **fflush(stdin);**  cout<<"\n-Nhap vao ho ten giao vien : ";  **cin.getline(ht,30);**  cout<<"\n-Nhap vao tuoi : ";  **cin>>tuoi;**  **fflush(stdin);**  cout<<"\n-Nhap vao bang cap giao vien : ";  **cin.getline(bc,15);**  cout<<"\n-Nhap vao chuyen nganh giao vien : ";  **cin.getline(cn,15);**  cout<<"\n-Nhap vao bac luong : ";  **cin>>bl;**  **}** |

Phương thức in() định nghĩa cần lưu ý căn chỉnh khoảng cách giữa các thuộc tính cho hợp lý và đẹp.

Phương thức tinhtl() định nghĩa bình thường

Chương trình chính viết như sau:

|  |
| --- |
| int **main()**  **{**  **GV gv;**  **gv.nhap();**  cout<<"Thong tin cua giao vien nhap la:"<<endl;  **gv.in();**  return 0;  **}** |

**Mở rộng:** Từ chương trình này, hãy mở rộng định nghĩa hàm khởi tạo và hàm hủy phù hợp…

**Bài 8:**

Xây dựng lớp khách hàng KH gồm các thuộc tính họ tên ht (kiểu char[30]), ngày sinh ns (kiểu struct NS(ngày, tháng, năm)), số chứng minh thư cmt (kiểu char[10]), hộ khẩu k (kiểu char[50]). Trong lớp có định nghĩa các hàm sau:

* Hàm khởi tạo không có tham số
* Hàm khởi tạo có tham số
* Hàm hủy
* Hàm xuất các thông tin trên của khách hàng ra màn hình

Viết chương trình tạo thông tin cho 1 khách hàng. In ra màn hình thông tin của khách hàng đó.

**Hướng dẫn:**

Xây dựng lớp KH với các thuộc tính như sau:

|  |
| --- |
| **private:**  char **ht**[30];  **NGAY ns;**  char **cmt**[10];  char **hk**[50]; |

Trong đó thuộc tính ns có kiểu cấu trúc NGAY phải được định nghĩa trước định nghĩa lớp như sau:

|  |
| --- |
| **struct NGAY**  **{**  **int** ngay,thang,nam;  **};** |

Các phương thức trong đó lưu ý phương thức khởi tạo và phương thức hủy như sau:

Phương thức khởi tạo không có tham số:

|  |
| --- |
| **KH()**  **{**  **strcpy**(ht," ");  **ns.ngay**=0;  **ns.thang**=0;  **ns.nam**=0;  **strcpy**(cmt," ");  **strcpy**(hk," ");  **}** |

Phương thức khởi tạo có tham số:

|  |
| --- |
| **KH(char \*hoten,int ng,int th,int n,char \*scmt,char \*hokhau)**  **{**  **strcpy(**ht,hoten**);**  **ns.ngay=**ng**;**  **ns.thang=**th**;**  **ns.nam=**n**;**  **strcpy(**cmt,scmt**);**  **strcpy(**hk,hokhau**);**  **}** |

Có thể xây dựng phương thức khởi tạo với các tham số có giá trị ngầm định

Chương trình chính:

|  |
| --- |
| int **main**()  {  **KH kh1("Tran Anh",15,10,2000,"16231333","Ha Noi - Viet Nam");**  **kh1.in();**  cout<<endl;  return 0;  } |

Lưu ý phương thức nhập thuộc tính ngày sinh phải nhập như sau:

|  |
| --- |
| cin>>ns.ngay;  cin>>ns.thang;  cin>>ns.nam; |

**Mở rộng:** Từ chương trình này, hãy mở rộng định nghĩa phương thức nhập, xuất, tính toán…