实验报告

1. 课后题1

实验要求：

分文件改写

实验代码：

头文件声明：

|  |
| --- |
| #ifndef POINT\_H\_H  #define POINT\_H\_H  #include <iostream>  using namespace std;  class point  {  public:  point(float x,float y);  void setpoint(float,float);  float getX() const {return x;}  float getY() const {return y;}  friend ostream & operator<<(ostream &,const point &);  protected:  float x,y;  };  #endif |

|  |
| --- |
| #ifndef CIRCLE\_H\_H  #define CIRCLE\_H\_H  #include <iostream>  #include "point.h"  using namespace std;  class circle:public point  {  public:  circle(float = 0,float = 0,float = 0);  void setradius(float);  float getradius() const;  float aera() const;  friend ostream&operator<<(ostream &,const circle&);  protected:  float radius;  };  #endif |

|  |
| --- |
| #ifndef CYLINDER\_H\_H  #define CYLINDER\_H\_H  #include <iostream>  #include "circle.h"  using namespace std;  class cylinder:public circle  {  public:  cylinder(float =0,float = 0,float =0,float =0);  void setheight(float);  float getheight() const;  float aera() const;  float volume() const;  friend ostream& operator<<(ostream&,const cylinder&);  protected:  float height;  };  #endif |

函数定义

|  |
| --- |
| #include <iostream>  #include "point.h"  using namespace std;  point::point(float a,float b)  {  x = a;  y = b;  }  void point::setpoint(float a,float b)  {  x = a;  y = b;  }  ostream &operator<<(ostream &out,const point &p)  {  out<<"["<<p.x<<","<<p.y<<"]"<<endl;  return out;  } |

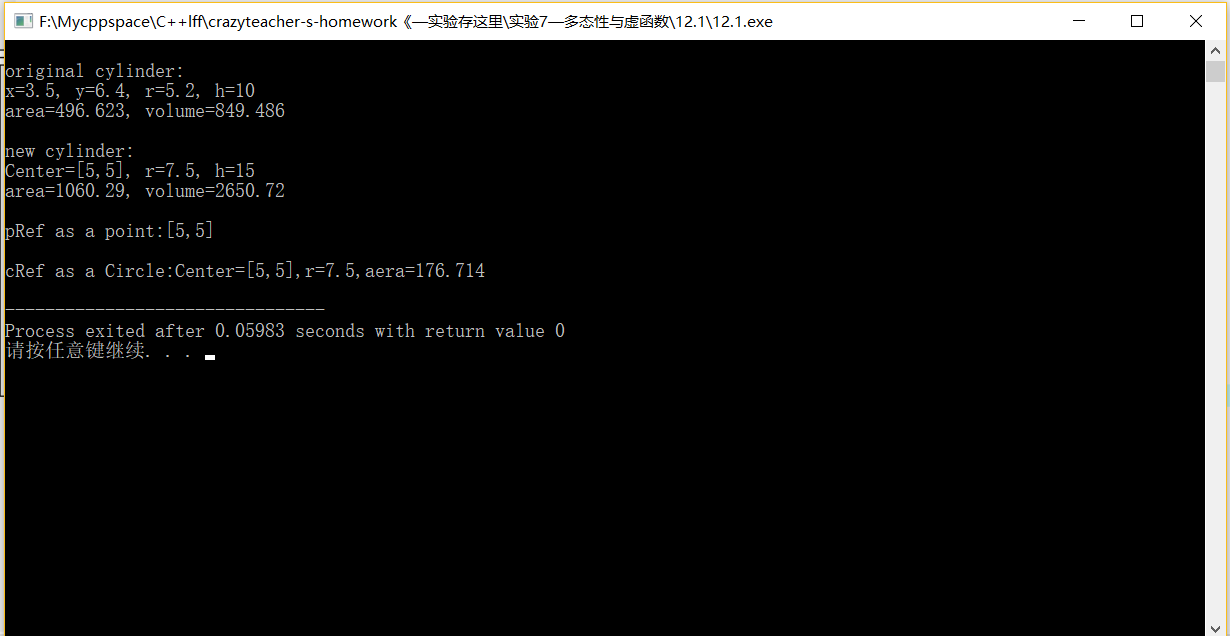
|  |
| --- |
| #include <iostream>  #include "circle.h"  using namespace std;  circle::circle(float a,float b,float c):point(a,b),radius(c){}  void circle::setradius(float r)  {  radius = r;  }  float circle::getradius()const  {  return radius;  }  float circle::aera()const  {  return 3.14159\*radius\*radius;  }  ostream &operator<<(ostream &out,const circle &p)  {  out<<"Center=["<<p.x<<","<<p.y<<"],r="<<p.radius<<",aera="<<p.aera()<<endl;  return out;  } |

|  |
| --- |
| #include <iostream>  #include "cylinder.h"  using namespace std;  cylinder::cylinder(float a,float b,float r,float h):circle(a,b,r),height(h){}  void cylinder::setheight(float h)  {  height = h;  }  float cylinder::getheight()const  {  return height;  }  float cylinder::aera()const  {  return 2\*circle::aera() + 2\*3.14159\*radius\*height;  }  float cylinder::volume()const  {  return circle::aera()\*height;  }  ostream &operator<<(ostream &out,const cylinder&c)  {  out<<"Center=["<<c.x<<","<<c.y<<"], r="  <<c.radius<<", h="<<c.height <<"\narea="  <<c.aera()<<", volume="<<c.volume()<<endl;  return out;  } |

Main函数

|  |
| --- |
| #include <iostream>  #include "cylinder.h"  using namespace std;  int main()  {  cylinder cy1(3.5,6.4,5.2,10);  cout<<"\noriginal cylinder:\nx="<<cy1.getX()  <<", y="<<cy1.getY()<<", r=" <<cy1.getradius()  <<", h="<<cy1.getheight()<<"\narea="<<cy1.aera()  <<", volume="<<cy1.volume()<<endl;  cy1.setheight(15);  cy1.setradius(7.5);  cy1.setpoint(5,5);  cout<<"\nnew cylinder:\n"<<cy1;  point &pRef=cy1;  cout<<"\npRef as a point:"<<pRef;  circle &cRef=cy1;  cout<<"\ncRef as a Circle:"<<cRef;  return 0;  } |

运行结果:



结果分析：

可以通过函数调取类内的数值，也能对数值进行修改，分文件时要注意包含关系防止重定义。

1. 课后题5

实验目的：

设计一个程序，以shape基类派生出五个派生类；运用虚函数计算面积、求和

用基类指针数组。

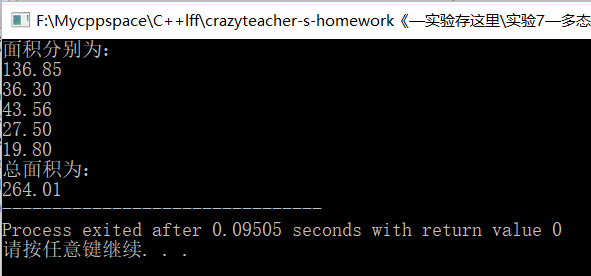
实验代码

|  |
| --- |
| #ifndef SHAPE\_H\_H  #define SHAPE\_H\_H  #include <iostream>  using namespace std;  class shape  {  public:  virtual double area() const =0;  };  class circle:public shape //圆形  {  public:  circle(double r):radius(r){}  virtual double area()const{return 3.14159\*radius\*radius;};  protected:  double radius;  };  class rectangle:public shape //矩形  {  public:  rectangle(double l,double r):c(l),k(r){}  virtual double area() const {return c\*k;};  protected:  double c,k;  };  class square:public shape //正方形  {  protected:  double s;  public:  square(double s):s(s){}  virtual double area() const {return s\*s;};  };  class trapezoid:public shape //梯形  {  protected:  double shang,xia,h;  public:  trapezoid(double s,double x,double h):shang(s),xia(x),h(h){}  virtual double area()const {return (shang+xia)\*h/2.0;};  };  class triangle:public shape //三角形  {  protected:  double di,h;  public:  triangle(double d,double h):di(d),h(h){}  virtual double area()const {return 0.5\*di\*h;};  };  #endif |

Main函数

|  |
| --- |
| #include <iostream>  #include <iomanip>  #include "shape.h"  using namespace std;  int main()  {  circle cir(6.6);  rectangle rec(6.6,5.5);  square squ(6.6);  trapezoid tra(6.6,4.4,5);  triangle tri(6.6,6);  shape \*p[5]={&cir,&rec,&squ,&tra,&tri};  double areas = 0.0;  cout<<"面积分别为：\n";  for(int i=0;i<5;i++)  {  areas += p[i]->area();  cout<<fixed<<setprecision(2)<<p[i]->area()<<endl;  }  cout<<"总面积为：\n";  cout<<areas;  } |

运行结果：



结果分析：

利用同名的虚函数对不同形状的图形进行面积计算，由上到下分别为圆，矩形，正方形、梯形、三角形。利用基类指针指向派生类实现运算。

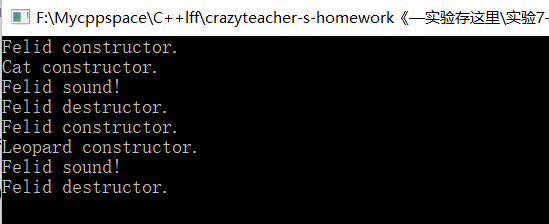
1. 分析代码的运行结果

代码如下（加粗红字为添加内容）

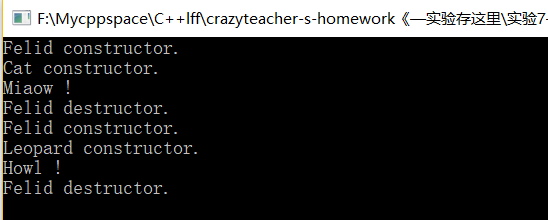
|  |
| --- |
| #include <iostream>  using namespace std;  class Felid  {  public:  Felid()  {  cout<<"Felid constructor."<<endl;  }  ~Felid()  {  cout<<"Felid destructor."<<endl;  }  **virtual** void sound() const  {  cout<<"Felid sound!"<<endl;  }  };  class Cat: public Felid  {  public:  Cat()  {  cout<<"Cat constructor."<<endl;  }  ~Cat()  {  cout<<"Cat destructor."<<endl;  }  void sound() const  {  cout<<"Miaow !"<<endl;  }  };  class Leopard: public Felid  {  public:  Leopard()  {  cout<<"Leopard constructor."<<endl;  }  ~Leopard()  {  cout<<"Leopard destructor."<<endl;  }  void sound() const  {  cout<<"Howl !"<<endl;  }  };  int main()  {  Felid \*animal;  animal = new Cat;  animal->sound();  delete animal;  animal = new Leopard;  animal->sound();  delete animal;  return 0;  } |

运行结果：

原代码结果：



改后结果：



结果分析：

可以看出在原代码中，如果不将基类中的同名函数声明为虚函数，会导致main函数中基类指针不能分辨指向的sound函数是哪个类的函数，直接指向了基类的sound函数，改为虚函数后，指针可以分辨出具体指向的函数，就能输出对应的字符串例如改后代码的Miaow！和Howl！。也可以从结果中看出类的构造顺序由基类到派生类，析构顺序由派生类到基类。

1. 4、有一个交通工具类Vehicle，定义如下：

class Vehicle{

public:

Vehicle() { cout<<"Vehicle constructor."<<endl; }

~Vehicle() { cout<<"Vehicle destructor."<<endl; }

virtual void display() const=0;

};

将Vehicle作为基类派生小车类Car，卡车类Truck和轮船类Boat。试设计并定义上述派生类，并通过display()函数显示出不同类型交通工具的相关信息。题目要求：仿照第3题代码，为对象申请动态空间，并通过构造函数和析构函数的设计，在程序运行结果中反映出动态对象创建与撤销的正确过程。

实验目标：

写出交通工具类及其派生类，在构造和析构函数中加入一定信息了解构造与析构的顺序

实验代码：

Header

|  |
| --- |
| #ifndef VEHICLE\_H\_H  #define VEHICLE\_H\_H  #include <iostream>  using namespace std;  class Vehicle  {  public:  Vehicle() { cout<<"Vehicle constructor."<<endl; }  ~Vehicle() { cout<<"Vehicle destructor."<<endl; }  virtual void display() =0;  };  class Car:public Vehicle  {  public:  virtual void display();  Car(int s):seat(s){ cout<<"Car constructor."<<endl; }  ~Car() { cout<<"Car destructor."<<endl; }  protected:  int seat;//座位数  };  class Truck:public Vehicle  {  public:  virtual void display();  Truck(double l):load(l) { cout<<"Truck constructor."<<endl; }  ~Truck() { cout<<"Truck destructor."<<endl; }  protected:  double load;//载重  };  class Boat:public Vehicle  {  public:  virtual void display();  Boat(double t):tonnes(t) { cout<<"Boat constructor."<<endl; }  ~Boat() { cout<<"Boat destructor."<<endl; }  protected:  double tonnes;//排水量  };  #endif |

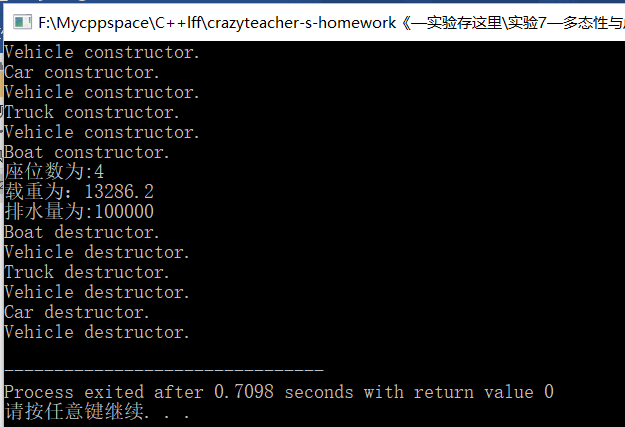
函数定义

|  |
| --- |
| #include <iostream>  #include "Vehicle.h"  using namespace std;  void Car::display()  {  cout<<"座位数为:"<<seat<<endl;  }  void Truck::display()  {  cout<<"载重为："<<load<<endl;  }  void Boat::display()  {  cout<<"排水量为:"<<tonnes<<endl;  } |

Main函数

|  |
| --- |
| #include <iostream>  #include "Vehicle.h"  using namespace std;  int main()  {  Car car(4);  Truck truck(13286.2);  Boat boat(100000);  Vehicle \*p[3]= {&car,&truck,&boat};  for(int i=0;i<3;i++)  {  p[i]->display();  }  delete []p;  } |

运行结果：



结果分析:

前六行分别表现了Car、Truck、Boat类的构造过程。中间三行由虚函数display输出各种不同的交通工具的特性。后六行是析构过程。由基类指针p分别指向三个类的对象。

5、现有一个学校人员管理系统，请设计并实现该系统的人员信息录入及显示的功能。

实验要求：

1. 设计基类Person、教师类Teacher、学生类Student、职工类Employee，使得教师、学生和职工三类均由Person派生；
2. Person类中至少包含三个private成员：编号id、姓名name、工作时间worktime，并包含人员信息输出函数virtual void display(){cout<<"姓名: "<<name<<endl;}；Teacher类人员还包括职称、所属学院、工资等信息；Student类人员将包括所属学院、班级职务等信息；Employee类人员将包括所属部门、工资等信息；
3. 三个派生类中，信息输出函数的名称应均与Person类中的成员函数display()同名，且派生类中的display()需通过调用Person::display()输出各类人员的姓名；
4. 设计一个简单的菜单，方便终端用户对系统的操作；
5. 利用多态性，实现各类人员信息的输出。

实验代码：

Header

|  |
| --- |
| #ifndef PERSON\_H\_H  #define PERSON\_H\_H  #include<iostream>  #include <string.h>  using namespace std;  class person  {  public:  person(string i,string n,double t):id(i),name(n),worktime(t){}  virtual void display();  int time();  private:  string id;  string name;  double worktime;  };  class student:public person  {  public:  student(string i,string n,double t,string m,string c):person(i,n,t),major(m),classpost(c){}  virtual void display();  private:  string major;  string classpost;  };  class teacher:public person  {  public:  teacher(string i,string n,double t,string co,string tit,double s):person(i,n,t),college(co),title(tit),salary(s){}  virtual void display();  private:  string college;  string title;  double salary;  };  class employee:public person  {  public:  employee(string i,string n,double t,string dep,double s):person(i,n,t),department(dep),salary(s){}  virtual void display();  private:  string department;  double salary;  };  #endif |

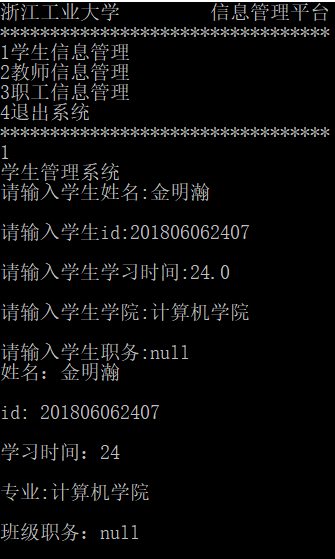
函数定义：

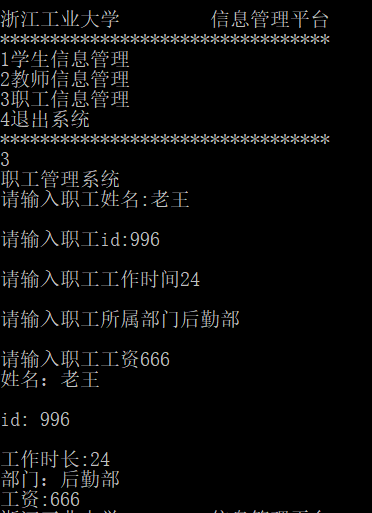
|  |
| --- |
| #include <iostream>  #include "person.h"  #include <string.h>  using namespace std;  void person::display()  {  cout<<"姓名："<<name<<endl<<endl  <<"id: "<<id<<endl<<endl;  }  int person::time()  {  return worktime;  }  void student::display()  {  person::display();  cout<<"学习时间："<<person::time()<<endl<<endl;  cout<<"专业:"<<major<<endl<<endl  <<"班级职务："<<classpost<<endl<<endl;  }  void teacher::display()  {  person::display();  cout<<"教学时长："<<person::time()<<endl<<endl;  cout<<"所属学院："<<college<<endl<<endl  <<"职称："<<title<<endl<<endl  <<"工资："<<salary<<endl<<endl;  }  void employee::display()  {  person::display();  cout<<"工作时长:"<<person::time()<<endl;  cout<<"部门："<<department<<endl;  cout<<"工资:"<<salary<<endl;  } |

Main函数

|  |
| --- |
| #include <iostream>  #include "person.h"  #include <string.h>  using namespace std;  int main()  {  while(1)  {  cout<<"浙江工业大学 信息管理平台"<<endl;  cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;  cout<<"1\学生信息管理\n";  cout<<"2\教师信息管理\n";  cout<<"3\职工信息管理\n";  cout<<"4\退出系统\n";  cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;  int i;  cin>>i;  person \*p;  if(i == 4) break;  else if(i == 1)  {  cout<<"学生管理系统\n";  string name;  string id;  double time;  string college;  string classpos;  cout<<"请输入学生姓名:";  cin>>name;  cout<<"\n请输入学生id:";  cin>>id;  cout<<"\n请输入学生学习时间:";  cin>>time;  cout<<"\n请输入学生学院:";  cin>>college;  cout<<"\n请输入学生职务:";  cin>>classpos;  student stu(id,name,time,college,classpos);  p = &stu;  p->display();  }  else if(i == 2)  {  cout<<"老师管理系统\n";  string name;  string id;  double time;  double s;  string college;  string pos;  string tit;  cout<<"请输入教师姓名:";  cin>>name;  cout<<"\n 请输入教师id:";  cin>>id;  cout<<"\n 请输入教师授课时间:";  cin>>time;  cout<<"\n 请输入教师学院:";  cin>>college;  cout<<"\n 请输入教师职称:";  cin>>tit;  cout<<"\n 请输入教师工资:";  cin>>s;  teacher tea(id,name,time,college,tit,s);  p = &tea;  p->display();  }  else if(i == 3)  {  cout<<"职工管理系统\n";  string name;  string id;  double time;  string dep;  double s;  cout<<"请输入职工姓名:";  cin>>name;  cout<<"\n请输入职工id:" ;  cin>>id;  cout<<"\n请输入职工工作时间";  cin>>time;  cout<<"\n请输入职工所属部门";  cin>>dep;  cout<<"\n请输入职工工资";  cin>>s;  employee emp(id,name,time,dep,s);  p = &emp;  p->display();  }  }    } |

运行结果：



结果分析：

设计了一个简单的菜单，实现了各个派生类的display函数，利用了多态性，用基类指针指向各个派生类，并且录入数据，并输出数据。

实验感受：

相比起上次实验更加简单，上次实验遇到的问题也得到了解决。但就是感觉越来越不会敲代码了（qvq）。