**课程实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| 实验名称 | 实验三 继承程序设计 | | | | |
| 实  验  目  的  及  要  求 | 1. 理解继承的含义，掌握派生类的定义方法和实现。 2. 理解公有继承下基类成员对派生类成员和派生类对象的可见性，能正确地访问继承层次中的各种类成员。 3. 理解保护成员在继承中的作用，能够在适当的时候选择使用保护成员，以便派生类成员可以访问基类的部分非公开的成员。 4. 理解多继承的特点以及多继承下派生类的创建。 5. 掌握派生类构造函数的书写及执行特点。理解多继承及其二义性的产生以及解决方法。 6. 理解组合和多继承的区别。 7. 理解赋值兼容，掌握继承机制下赋值兼容的使用方法。 | | | | |
| 实验  环境 | 操作系统：Windows 10 loT企业版LTSC 版本21H2(操作系统内部版本19044.1645)  开发环境：Clion 2021.3.4 | | | | |
| 实  验  内  容 | 1.，并能够计算矩形的面积，并给出测试程序。  #include <iostream>  using namespace std;  class Point  {  public:  Point(int a,int b){ x=a; y=b;}  Point(){x=0;y=0;}  int GetY(){return y;}  int GetX(){return x;}  void SetY(int b){ y=b;}  void SetX(int a){ x=a;}  private:  int x;  int y;  };  class Rectangle : public Point  {  public:  Rectangle(int x1,int x2,int y1,int y2)  {  dot1.SetX(x1);  dot1.SetY(y1);  dot2.SetX(x2);  dot2.SetY(y2);  area=((dot1.GetX()-dot2.GetX())\*(dot1.GetY()-dot2.GetY()));  if(area<0)area=-area;//面积为正数  }  int Getarea( )  {  return area;  }  private:  Point dot1;  Point dot2;  int area;  };  int main() {  Rectangle sample1(1,3,1,3);  cout<<sample1.Getarea()<<endl;  return 0;  }  2.设计一个圆类Circle和一个桌子类Table，另设计一个圆桌类Roundtable，它是前面两个类的派生类，要求编写测试程序给出输出一个圆桌的高度、面积和颜色等数据。  #include <iostream>  using namespace std;  class Circle{  protected:  double r;  public:  void setr(int \_r)  {  r=\_r;  }  };  class Table{  protected:  double h;  char color;  public:  void seth(int \_h)  {  h=\_h;  }  void setcolor(char \_c)  {  color=\_c;  }  };  class Rouldtable: public Circle,public Table{  public :  double Getarea()  {  return r\*r\*3.1416;  }  int Gethight(){  return h;  }  char Getcolor()  {  return color;  }  };  int main(){  Rouldtable c1;  c1.setr(3);  c1.seth(2);  c1.setcolor('a');  cout<<"area:"<<c1.Getarea()<<endl;  cout<<"hight:"<<c1.Gethight()<<endl;  cout<<"color:"<<c1.Getcolor()<<endl;  return 0;  }  3.下面的程序可以输出ASCII字符与所对应的数字的对照表。修改下列程序,使其可以输出字母a 到z与所对应的数字的对照表。  #include<iostream>  #include<cstring>  #include <iomanip>  using namespace std;  class Table  {  public:  Table(int p)  {  i=p;  }  void ascii( );  protected :  int i;  };  void Table::ascii( )  {  int k=1;  for (; i< 123; i++)  {  cout<<setw(4)<<i<<" "<<(char)i;  if ((k)%12==0)  cout<<"\n";  k++;  }  cout<<"\n";  }  class Der\_table:public Table  {  public:  Der\_table(char start,char end,char \*m):Table(start)  {  if(\*m<=end){  c = new char[strlen(m) + 1];  strcpy(c, m);}  }  void print( );  protected:  char \*c;  };  void Der\_table::print( )  {  cout<<c<<"\n";  Table::ascii();  }  /\*int main()  {  Der\_table ob1(32,"ASCII value---char");  ob1.print();  return 0;  }\*/  int main()  {  Der\_table ob('a','z',"ASCII value---char");  ob.print();  return 0;  }  4.编写一个求出租车收费的程序，输入起始站、终止站和路程。  计费标准为：3 公里以内10元，3 公里以后每1公里加 2 元，超过 15 公里，每公里加 3元。  要求：设计一个站类Station(用于设置起始站、终止站)和路程类Mile(用于设置路程)，由这两个类派生出收费类Price(用于计费)。  已知Station类和Mile类以及测试main函数如下，请编写Price类。  （可以使用string类型代替给出代码中的字符数组。）  #include <iostream>  #include<cstring>  using namespace std;  #define Max 50  class Station  {  public:  Station()  { strcpy(from," "); strcpy(to," "); }  Station(char f[],char t[])  { strcpy(from,f); strcpy(to,t); }  void getdata()  {  cout<<"pls input start and destinition:";  cin>>from>>to;  }  void disp()  {  cout<<"from"<<from<<"to"<<to<<"station";  }  protected:  char from[Max];  char to[Max];  };  class Mile  {  public:  Mile(){mile=0;}  Mile(double m){mile=m;}  void getdata()  {  cout<<"pls input distance：";  cin>>mile;  }  void disp()  {  cout<<"is"<<mile<<"km";  }  protected:  double mile;  };  class Price: public Mile, public Station{  public:  void disp()  {  cout<<"from: "<<from<<endl;  cout<<"to: "<<to<<endl;  int ans;  if(mile<3)ans=10;  else {  if (mile < 15) {  mile -= 3;  ans += 10;  ans += mile \* 2;  }  else{  mile-=15;  ans+=mile\*3+34;  }  }  cout<<"price:"<<ans<<endl;  }  Price()  {  mile=0;  };  Price(char const \*f,char const \*t,double m){  strcpy(from,f);  strcpy(to,t);  mile=m;  };  };  int main()  {  Price A; //乘车，  A.Station::getdata(); //输入起始站和终点站  Price B("QUT","CentralPark",20); //乘车，有起始站和终点站  //A和B是Price的两种初始化方式。20是路程数。  cout<<"result:"<<endl;  A.disp(); //输出此次乘车价格  B.disp(); //输出此次乘车价格  return 0;  }  5.设计一个小猫钓鱼的游戏程序。基本需求如下：  （1）每个小猫有自己的等级(level)和经验分(exp)，每累计获得500经验分，就升一级，同时经验分清0；  （2）小猫每次只能钓一条鱼，如果钓上一条章鱼(Octopus)，经验分的增加值为 2\*章鱼的重量；如果钓上一条鲸鱼(Whale)，经验分增加200；如果钓上一个金龟(Turtle)，则等级直接升一级；如果钓上来一条鲨鱼(Shark),则在等级不变的前提下减少经验分(至多减至0)，减少值为5\*鲨鱼的重量。考虑到游戏的趣味性，将来可能还要增加其他类型的鱼以及相应的奖励或惩罚方法。  请根据上面的模型描述，制定合理的设计方案，请完整定义并实现小猫类，其中类的成员至少要有一个成员函数CatchFish，用来体现小猫钓鱼的行为过程。同时设计Fish类，使用继承机制从其派生出具体的鱼类。  #include <iostream>  #include<cstring>  #include<string>  using namespace std;  class Fish{  public:  Fish() {}  Fish(string fishname)  {  species=fishname;  }  string Getspecies()  {  return species;  }  int Getvalue()  {  return score;  }  protected:  string species;  int randnum=rand()%100;  int rands=rand()%100;  int score=randnum;  };  class Octopus:public Fish  {  public:  Octopus()  {  species="Octopus";  score=2\*score;  }  };  class Turtle:public Fish{  public:  Turtle()  {  species="Turtle";  score=500;  }  };  class Shark:public Fish  {  public:  Shark()  {  species="Shark";  score=0;  }  };  class Cat:public Fish  {  public:  void CatchFish()  {  rands=rand()%100;  if(rands<50)  {  exp=exp+Getvalue()\*2;  cout<<"A Octopus is catched"<<endl;  }  if(rands>=50&&rands<=90)  {  exp+=500;  cout<<"A Turtle is catched"<<endl;  }  if(rands>90)  {  exp=0;  cout<<"A Shark is catched"<<endl;  }  if(exp>=500)  {  level++;  exp-=500;  }  cout<<"Exp is:"<<exp<<" Level is:"<<level<<endl;  }  protected:  int level=0;  int exp=0;  };  int main() {  srand((unsigned)time(NULL));  int randnum,round;  Cat cat1;  cout<<"pls input howmany rounds for the cat:"<<endl;  cin>>round;  for(int i=1;i<=round;i++)  {  cat1.CatchFish();  }  return 0;  } | | | | |
| 调试  过程  及  实验  结果 | 1、遇到中文UTF-8编码转换ANSI导致的乱码问题，最好的解决方法是不用中文。  2、小猫钓鱼模拟抽卡游戏 用时间种子随机数  3、多继承 使用作用域分辨符：：加以消除。 | | | | |