**第6章习题参考答案**

1、什么叫做多态性？在C++中是如何实现多态性的？

|  |
| --- |
| （1）什么叫做多态性  多态性是指当不同的对象收到相同的消息时产生不同的动作。多态性使软件中的对象行为更加符合软件所模拟的客观世界中对象的行为多样性，为软件设计中的功能实现和任务调度提供了灵活性。  （2）C++中如何实现多态性  C++在基类中声明相应的函数为 virtual 型，然后在派生类中实现该函数，这样就可以通过基类指针调用派生类对象的函数，实现了运行时动态绑定，即多态的功能。 |

2、简述虚函数与纯虚函数的作用和区别。

|  |
| --- |
| （1）作用  虚函数的作用是为了实现多态性。虚函数允许指向派生类对象的基类指针可以调用覆盖了基类虚方法的对应子类方法。运用抽象手法，把不同的子类对象的共同方法抽取出来，形成虚函数，可以屏蔽不同子类对象之间的差异，写出通用的代码，做出通用的编程，以适应需求的不断变化。赋值之后，父类指针就可以根据当前赋值给它的子对象的特性以不同的方式运作，从而形成动态多态性。  纯虚函数是一种特殊的虚函数，在许多情况下，在基类中不能对虚函数给出有意义的实现，而把它声明为纯虚函数，它的实现留给该基类的派生类去做。这就是纯虚函数的作用。通俗来说，使用纯虚函数的类我们称为抽象类，该类不提供方法只提供接口，也可以很形象的理解为就是一个API。  （2）区别  类里声明为虚函数的话,这个函数是实现的，哪怕是空实现，它的作用就是为了能让这个函数在它的子类里面可以被重载，这样编译器就可以使用后期绑定来达到多态性。纯虚函数只是一个接口，是个函数的声明而已，它要留到子类里去实现，纯虚函数是实现事物多态性的又一种形式。  虚函数在子类里面也可以不重载；但纯虚函数必须在子类去实现。通常我们把很多函数加上virtual，是一个好的习惯，虽然牺牲了一些性能，但是增加了面向对象的多态性，因为它允许父类里面的这个函数在子类里面被修改它的实现。  带纯虚函数的类叫虚基类，这种基类不能直接生成对象，而只有被继承，并重写其虚函数后，才能使用。这样的类也叫抽象类。虚函数是为了继承接口和默认行为，纯虚函数只是继承接口，行为必须重新定义。 |

3、什么叫做抽象类？抽象类有何作用？抽象类的派生类是否一定要给出纯虚函数的实现？

|  |
| --- |
| 带有纯虚函数的类是抽象类。其主要作用是通过它为一个类族建立一个公共的接口，使他们能够更有效地发挥多态特性。抽象类声明了一个类族派生类的共同接口，而接口的完整实现，即纯虚函数的函数体，要由派生类自己定义。抽象类派生出新的类之后，如果派生类给出所有纯虚函数的函数实现，这个派生类就可以定义自己的对象，因而不再是抽象类；反之，如果派生类没有给出全部纯虚函数的实现，此时的派生类仍然是一个抽象类。 |

4、在C++中，能否声明虚构造函数？为什么？能否声明虚析构函数？为什么？有何用途？

|  |
| --- |
| （1）能否声明虚构造函数？为什么？  在C++中，不能声明虚构造函数。多态性是指不同的对象对同一消息有不同的行为特性。虚函数作为运行时多态性的基础，主要是针对对象的，而构造函数是在对象产生之前运行的，因此虚构造函数是没有意义的。  （2）能否声明虚析构函数？为什么？有何用途？  可以声明虚析构函数，析构函数的功能是在该类对象消亡之前进行一些必要的清理工作，如果一个类的析构函数是虚函数，那么，由它派生而来的所有子类的析构函数也是虚函数。 析构函数设置为虚函数之后，在使用指针引用时可以动态联编，实现运行时的多态，保证使用基类的指针就能够调用适当的析构函数针对不同的对象进行清理工作。 |

5、编写一个抽象类Shape，在此基础上派生出类Rectangle和Circle，二者都有计算对象面积的函数getArea（），计算对象周长的函数getPerim（）。

|  |
| --- |
| #include<iostream>  #define PI 3.1415926;  using namespace std;  class Shape //抽象类的 定义  {  public:  virtual double GetArea() = 0; //纯虚函数  virtual double GetPerim() = 0; //纯虚函数  };  class Rectangle : public Shape //矩形类，公有继承  {  public: Rectangle(double aa, double bb) //带参数的 构造函数  {  a = aa;  b = bb;  cout << "长" << a << "宽" << b << endl;  }  virtual double GetArea()  {  return a \* b;  }  virtual double GetPerim()  {  return 2 \* (a + b);  }  private:  double a;  double b;  };  class Circle : public Shape //圆类，公有继承  {  public: Circle(double rr) //带参数的 构造函数  {  r = rr;  cout << "半径" << r << endl;  }  virtual double GetArea()  {  return r \* r \* PI;  }  virtual double GetPerim()  {  return 2 \* r \* PI;  }  private:  double r;  };  void main()  {  double length, width;  cout << "输入长和宽: ";  cin >> length >> width;  Rectangle rect(length, width);  cout << "面积是：" << rect.GetArea() << endl << "周长是：" << rect.GetPerim() << endl;  double rr;  cout << "输入半径: ";  cin >> rr;  Circle cir(rr);  cout << "面积是：" << cir.GetArea() << endl << "周长是：" << cir.GetPerim() << endl;  }  执行结果：  C:\Users\Administrator\AppData\Roaming\Tencent\Users\473025436\TIM\WinTemp\RichOle\EJE_N[1KDW@XJK_5QJB)2HH.png |

6、定义一个基类BaseClass，从它派生出类DerivedClass。BaseClass有成员函数fn1(),fn2(),其中fn1()是虚函数；DerivedClass也有成员函数fn1(),fn2()。在主函数中声明一个DerivedClass的对象，分别用BaseClass和DerivedClass的指针指向DerivedClass的对象，并通过指针调用fn1(),fn2(),观察运行结果。

|  |
| --- |
| #include <iostream>  using namespace std;  class BaseClass  {  public:  virtual void fn1();  void fn2();  };  void BaseClass::fn1()  {  cout << "调用基类的虚函数 fn1()" << endl;  }  void BaseClass::fn2()  {  cout << "调用基类的非虚函数 fn2()" << endl;  }  class DerivedClass : public BaseClass  {  public: void fn1();  void fn2();  };  void DerivedClass::fn1()  {  cout << "调用派生类的函数 fn1()" << endl;  }  void DerivedClass::fn2()  {  cout << "调用派生类的函数 fn2()" << endl;  }  void main()  {  DerivedClass aDerivedClass;  DerivedClass \*pDerivedClass = &aDerivedClass;  BaseClass \*pBaseClass = &aDerivedClass;  pBaseClass->fn1();  pBaseClass->fn2();  pDerivedClass->fn1();  pDerivedClass->fn2();  }  执行结果：  C:\Users\Administrator\AppData\Roaming\Tencent\Users\473025436\TIM\WinTemp\RichOle\3E[JU_83@ZD((O1A1PULRCH.png |

7、编写程序定义一个基类BaseClass，从它派生出类DerivedClass。在BaseClass中声明虚析构函数，在主函数中将一个动态分配的DerivedClass的对象地址赋给一个BaseClass的指针，然后通过指针释放对象空间。观察程序运行过程。

|  |
| --- |
| #include <iostream>  using namespace std;  class BaseClass {  public:  virtual ~BaseClass() {  cout << "~BaseClass()" << endl;  }  };  class DerivedClass : public BaseClass {  public:  ~DerivedClass() {  cout << "~DerivedClass()" << endl;  }  };  void main()  {  BaseClass\* bp = new DerivedClass;  delete bp;  }  执行结果：  C:\Users\Administrator\AppData\Roaming\Tencent\Users\473025436\TIM\WinTemp\RichOle\R[_2CPD(3S0%{OA$G@MQ`QQ.png |

|  |
| --- |
|  |

8、编写程序定义类Point，有数据成员x、y，为其定义友元函数实现重载“+”。

|  |  |  |
| --- | --- | --- |
| |  | | --- | | #include <iostream.h>  class Point  {  private:  int x, y;  public:  Point(int a = 0, int b = 0) :x(a), y(b) {}  void printInfo();  friend Point operator + (Point const & a, Point const & b);  };  Point operator + (Point const & a, Point const & b)  {  return Point(a.x + b.x, a.y + b.y);  }  void Point::printInfo()  {  cout << "(" <<x<<","<<y<<")" << endl;  }  int main()  {  Point p(1, 2), q(2, 3);  Point k;  k= p + q;  k.printInfo();  return 0;  }  执行结果：  C:\Users\Administrator\AppData\Roaming\Tencent\Users\473025436\TIM\WinTemp\RichOle\~5G98[VPW)KNLVGRHR[~[87.png | |  | |

9、编写一个程序实现图书和杂志销售管理。当输入一系列图书和杂志销售记录后，将销售良好（图书销售：每月500本以上，杂志销售：每月2500本以上）的图书和杂志名称显示出来。

|  |
| --- |
| #include <iostream>  #include <string>  using namespace std;  int main() {  string bookName;  string magazineName;  string a[100], b[100];  int bookNumber;  int magazineNumber;  int i = 0, j = 0, c, d;  do {  cout << "请输入图书的名称:" << endl;  cin >> bookName;  cout << "请输入图书销售的数量:" << endl;  cin >> bookNumber;  if (bookNumber > 500)  {  a[i] = bookName;  i++;  }  cout << "请问还有图书需要输入吗？有请按1，没有请按0" << endl;  cin >> c;  } while (c == 1);  do {  cout << "请输入杂志的名称:" << endl;  cin >> magazineName;  cout << "请输入杂志销售的数量:" << endl;  cin >> magazineNumber;  if (magazineNumber > 2500)  {  b[j] = magazineName;  j++;  }  cout << "请问还有杂志需要输入吗？有请按1，没有请按0" << endl;  cin >> d;  } while (d == 1);  cout << "销售良好的图书有：" << endl;  for (int k = 0; k != i; k++)  cout << a[k] << '\n';  cout << "销售良好的杂志有：" << endl;  for (int x = 0; x != j; x++)  cout << b[x] << '\n';  return 0;  system("pause");  }  执行结果：  C:\Users\Administrator\AppData\Roaming\Tencent\Users\473025436\TIM\WinTemp\RichOle\2IS@]SNGK]%}I0ZTRE{63YR.png |

10、应用抽象类，求圆、圆内接正方形和圆外切正方形的面积和周长。

|  |
| --- |
| #include <iostream>  #include <math.h>  #define PI 3.1415  using namespace std;  class base  {  protected:  double r;  public:  virtual double Area() = 0;  virtual double Length() = 0;  };  class Circle :public base  {  public:  Circle(double R)  {  r = R;  }  double Area()  {  return PI \* r\*r;  }  double Length()  {  return 2 \* PI\*r;  }  };  class Square\_in :public base  {  public:  Square\_in(double R)  {  r = R;  }  double Area()  {  return 2 \* r\*r;  }  double Length()  {  return 4 \* sqrt(2)\*r;  }  };  class Square\_out :public base  {  public:  Square\_out(double R)  {  r = R;  }  double Area()  {  return 4 \* r\*r;  }  double Length()  {  return 8 \* r;  }  };  int main()  {  double radius;  cout << "请输入圆的半径: ";  cin >> radius;  Circle c(radius);  cout << endl;  cout << "圆的面积是: " << c.Area() << endl;  cout << "圆的周长是: " << c.Length() << endl;  cout << endl;  Square\_in si(radius);  cout << "内接正方形面积: " << si.Area() << endl;  cout << "内接正方形周长: " << si.Length() << endl;  cout << endl;  Square\_out so(radius);  cout << "外切正方形面积: " << so.Area() << endl;  cout << "外切正方形周长: " << so.Length() << endl;  cout << endl;  return 0;  }  执行结果：  C:\Users\Administrator\AppData\Roaming\Tencent\Users\473025436\TIM\WinTemp\RichOle\OXE1~W}W$LG%6`NELM{F)XL.png |