**新建项目**

**计算机学院（软件学院）实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **学生姓名** | 董扬 | **学号** | 171530422 | **实验成绩** |  |
| **专业** | 软件工程 | **班级** | 软工四班 | **实验日期** | 2019年11月4日 |
| **课程名称** | C#高级程序设计 | | | **任课教师** | 彭伟国 |
| **实验名称** | **C#编程进阶** | | | **实验序号** | 4 |
| **实验地点** | S409 | **实验台号** | 48 | **指导教师** | 彭伟国 |
| **一、实验目的及要求**  1．会使用类的继承机制派生子类。  2．会使用虚函数、抽象类、接口实现多态。  3．会使用List和Dictionary泛型集合类组织数据。  4．会使用ArryList和Hashtable非泛型集合类组织数据。  5．掌握委托的定义和使用方法。  6．会使用泛型知识设计程序。  7．会使用C#反射机制获取类信息。 | | | | | |
| 二、实验内容（或实验原理、实验拓扑）  第一次：  1.写一个Student类和一个Teacher类,他们都有一个打招呼的方法,不同的是Studetn打招呼是说"大家好,我叫XX,我今年XX岁了,我的爱好是XXX",Teacher的打招呼的方法是说"大家好,我叫XX,我今年XX岁了,我已经工作XX年了"?  2．自己定义一个父类Person,两个子类Teacher和Student类.练习类内部构造器的调用，练习子类调用父类的构造器.  3．自己试试,子类可以自动转父类,父类转子类要通过强转.为了不出错,可以先通过is判断或用as转换.  第二次：  1．定义一个动物类,在动物类中有一个叫的抽象方法. 写两个子类,一个猫一个狗,继承自动物类,并实现相应的抽象方法.（抽象类）  2．在一个程序中，既想实现按年龄排序，又想实现按姓名排序，还要实现按人气值排序，怎么办？  第三次  1．创建一个集合，里面添加一些数字，求平均值与和，最大值，最小值，并在控制台显示输出结果。  2．写一个长度为10的集合，要求在里面随机地存放10个数字，并在控制台上显示输出这10个数字。（提示：用到Random随机数类中的Next方法，首先创建随机数对象，然后调用Next方法）。  3.定义一个集合类（学生类），不仅可以通过下标查找学生，还可以通过姓名查找（设姓名没有重复)  第四次  1．将一个字符串数组中每个元素都转换成大写、小写，每个元素两边都加上双引号。  第五次  1．一个简单的类，包含一个有参数的构造器，一个GetValue的方法，一个Value属性，运用反射知识通过方法的名称来得到方法并且调用之。 | | | | | |
| **三、实验设备与环境**  **Windows7 PC机** | | | | | |
| **四、实验设计方案（包括实验步骤、设计思想、算法描述或开发流程等）**  第一次   1. 分别定义Student类和Teacher类，写出不同的打招呼函数即可 2. 定义一个父类Person,两个子类Teacher和Student类，使用base关键字即可调用父类构造器 3. 使用is关键字可判断对象类型，as关键字可实现转换   第二次   1. 使用absract关键字构建动物抽象类，再构建Dog和Cat类继承自动物类   第三次  1、使用Array中的函数即可实现  2、使用Random类的Next函数即可实现  3、  第四次  使用委托可将三个方法代理，减少代码书写  第五次  引入using System.Reflection头文件，运用反射知识，使用Type和其中函数进行实现 | | | | | |

|  |
| --- |
| **五、实验结果（包括设计效果、测试数据、运行结果等）**  第一次  1、  C:\Users\Administrator\Desktop\捕获.PNG  2、  C:\Users\Administrator\Desktop\捕获.PNG  3、    第二次  1、    2、    第三次  1、    2、    第四次    第五次 |
| **六、实验小结（包括收获、心得体会、注意事项、存在问题及解决办法、建议等）**  1．学会了使用类的继承机制派生子类。  2．学会了使用虚函数、抽象类、接口实现多态。  3．学会了使用List和Dictionary泛型集合类组织数据。  4．学会了使用ArryList和Hashtable非泛型集合类组织数据。 |
| **七、附录（包括作品、流程图、源程序及命令清单等）**  **实验代码**  第一次  1、  using System;  using System.Collections;  using System.Collections.Generic;  using System.Linq;  using System.Text;  namespace ConsoleApplication1  {  class Teacher  {  private string name;  private int years;  private int ages;  public Teacher(string n, int a, int y)  {  name = n;  ages = a;  years = y;  Console.WriteLine("大家好,我叫{0},我今年{1}岁了,我已经工作{2}年了", name, ages, years);  }  }  class Student  {  private string name;  private string hobby;  private int ages;  public Student(string n, int a, string h)  {  name = n;  ages = a;  hobby = h;  Console.WriteLine("大家好,我叫{0},我今年{1}岁了,我已经工作{2}年了", name, ages, hobby);  }  }  class Program  {  static void Main(string[] args)  {  Teacher t = new Teacher("1", 35, 10);  Student S = new Student("小明", 18, "5");  }  }  }  2、  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace \_4\_2  {  class Program  {  static void Main(string[] args)  {  Student student = new Student(1, "a", 5, 80);  Teacher teacher = new Teacher(2, "b", "c", "d");  }  }  public class Person  {  public int id;  public string name;  public Person(int pid, string pname)  {  id = pid;  name = pname;  }  }  public class Student : Person  {  public int class\_id { get; set; }  public int score;  public Student(int pid, string pname, int sclass\_id, int sscore)  : base(1, "a")  {  id = pid;  name = pname;  class\_id = sclass\_id;  Console.WriteLine("4");  }  }  public class Teacher : Person  {  public string position;  public string department;  public Teacher(int pid, string pname, string tposition, string tdepartment) : base(2, "b")  {  id = pid;  name = pname;  position = tposition;  department = tdepartment;  Console.WriteLine("5");  }  }  }  3、  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace \_1  {  class Person  {  private string name;  public string Name  {  get { return name; }  set { name = value; }  }  private int age;  public int Age  {  get { return age; }  set { age = value; }  }  public Person() { }  public Person(string na, int ag)  {  this.Name = na;  this.Age = ag;  }  public virtual void print()  {  Console.WriteLine("My name is " + Name);  Console.WriteLine("I am " + Age + " years old");  }  }  class Student : Person  {  private string xingke;  public string Xingke  {  get { return xingke; }  set { xingke = value; }  }  private string nianji;  public string Nianji  {  get { return nianji; }  set { nianji = value; }  }  public Student(string na, int ag, string xk, string nj)  {  this.Name = na;  this.Age = ag;  this.Nianji = nj;  this.Xingke = xk;  }  public new void print()  {  Console.WriteLine("My name is " + Name);  Console.WriteLine("I am " + Age + " years old");  Console.WriteLine(Nianji);  Console.WriteLine(Xingke);  }  }  class Problem  {  static void Main(string[] args)  {  Person p = new Person("as",20);  Student s = new Student("张三",15,"高一","理科");  Person p1 = new Student("张三", 18, "高一", "理科");  Student s1 = p1 as Student;  if (p1 is Student)  {  Console.WriteLine("转换成功");  s.print();  }  else  {  Console.WriteLine("转换失败");  }  }  }  }  第二次  1、  using System;  using System.Collections;  using System.Collections.Generic;  using System.Linq;  using System.Text;  namespace ConsoleApplication1  {  abstract class Animal  {  public string n;  public string s;  public abstract string talk();  }  class Cat : Animal  {  public Cat(String n, String s)  {  this.n = n;  this.s = s;  }  public override string talk()  {  return "动物 " + this.n + " " + this.s;  }  }  class Dog : Animal  {  public Dog(String n, String s)  {  this.n = n;  this.s = s;  }  public override string talk()  {  return "动物 " + this.n + " " + this.s;  }  }  class Program  {  static void Main(string[] args)  {  Animal c = new Cat("猫", "喵~喵!");  Animal d = new Dog("狗", "汪~汪!");  Console.WriteLine(c.talk());  Console.WriteLine(d.talk());  }  }  }  2、  using System;  using System.Collections.Generic;  namespace \_2\_2  {  class Program  {  static void Main(string[] args)  {  List<Teacher> list = new List<Teacher>();  list.Add(new Teacher("张三", 10000, 100));  list.Add(new Teacher("张三", 10000, 101));  list.Add(new Teacher("宋江", 10000, 101));  list.Add(new Teacher("李四", 5000, 102));  list.Add(new Teacher("王五", 1000, 103));  Console.WriteLine("排序后：");  list.Sort();  foreach (Teacher t in list)  Console.WriteLine(t);  }  }  public class Teacher : IComparable<Teacher>  {  public string Name;  public decimal Salary;  public int PopularityValues;  public Teacher(string name, decimal salary, int popularityValues)  {  this.Name = name;  this.Salary = salary;  this.PopularityValues = popularityValues;  }  public override string ToString()  {  return "name:" + this.Name + "\tsalary:" + this.Salary + "\tpopularityValues:" + this.PopularityValues;  }  public int CompareTo(Teacher other)  {  if (this.Salary == other.Salary)  {  if (this.Name == other.Name)  return (int)(this.PopularityValues - other.PopularityValues);  return this.Name.CompareTo(other.Name);  }  return (int)(this.Salary - other.Salary);  }  }  }  第三次  1、  using System;  using System.Collections;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace \_1  {  class Problem  {  static void Main(string[] args)  {  int[] arr = new int[10];  Console.WriteLine("请输入十个数");  for (int i = 0; i <= 9; i++)  {  arr[i] = int.Parse(Console.ReadLine());  }  Console.WriteLine("最大的数为" + arr.Max());  Console.WriteLine("最小的数为:" + arr.Min());  Console.WriteLine("平均数为" + arr.Average());  }  }  }  2、  using System;  using System.Collections;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace \_1  {  class Problem  {  static void Main(string[] args)  {  int[] arr = new int[10];  Random random = new Random();  for(int i=0;i<10;i++)  {  arr[i] = random.Next() %100;  }  for (int i = 0; i < 10; i++)  {  Console.WriteLine(arr[i]);  }  }  }  }  3、  第四次  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace \_4\_4\_1  {  public delegate string DelProStr(string name);  class Program  {  static void Main(string[] args)  {  string[] names = { "ABCdef", "hiJK1", "MnOpQ", "rsT", "UvwXYz" };  ProStr(names, delegate (string name)  {  return "\"" + name + "\"";  });  for (int i = 0; i < names.Length; i++)  {  Console.WriteLine(names[i]);  }  }  public static void ProStr(string[] names, DelProStr del)  {  for (int i = 0; i < names.Length; i++)  {  names[i] = del(names[i]);  }  }  public static string ProStrToUpper(string name)  {  return name.ToUpper();  }  public static string ProStrToLower(string name)  {  return name.ToLower();  }  public static string ProStrToSYH(string name)  {  return "";  }  //char[] split = { ' ' };  //string str = Console.ReadLine();  //string[] wordstring = str.Split(split);  //foreach (string s in wordstring)  //{  // Console.WriteLine("\"{0}\"", s);  //}  //Console.ReadKey();  }  }  第五次  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Reflection;  namespace TestSpace  {  public class Project  {  private string \_value;  public Project()  {  Console.WriteLine("调用无参构造函数");  }  public Project(string value)  {  \_value = value;  }  public string GetValue(string p)  {  if (\_value == null)  return "NULL";  else  return p + " : " + \_value;  }  public string Value  {  set  {  \_value = value;  }  get  {  if (\_value == null)  return "NULL";  else  return \_value;  }  }  public void print()  {  Console.WriteLine(Value);  }  }  }  class Program  {  public static void Main(string[] arge)  {  Type t = typeof(TestSpace.Project);  MethodInfo[] x = t.GetMethods();  foreach (MemberInfo xtemp in x)  {  Console.WriteLine(xtemp.ToString());  }  Console.WriteLine();  MemberInfo[] x2 = t.GetMembers();  foreach (MemberInfo xtemp in x2)  {  Console.WriteLine(xtemp.ToString());  }  Object o1 = "5";  object o = Activator.CreateInstance(t, o1);  MethodInfo mi = t.GetMethod("print");  mi.Invoke(o, null);  }  } |