浙江理工大学 2017—2018 学年第 2 学期

《C#程序设计》期末试卷（B）卷标准答案和评分标准

一、单选题（本大题共 32 分，每小题 2 分）

1	2	3	4	5
B	B	D	A	D
6	7	8	9	10
D	C	A	D	D
11	12	13	14	15
C	D	D	B	B
16				
D				

二、程序设计题（共 68 分）

1、

```
//数据中的值两两操作，得到最终结果
private static int MapCalculation(NumberCalculation cal, params int[] intArray)
{
    int temp = intArray[0];
    for (int i = 1; i < intArray.Length; i++)
        temp = cal(temp, intArray[i]);
    return temp;
}

//两个整数相加
private static int Add(int number1, int number2)
{
    return number1 + number2;
}
```

2

```
public struct Point2d
{
    public double x;
    public double y;
    public static readonly Point2d Origin = new Point2d();

    public Point2d(double x, double y)
    {
        this.x = x;
        this.y = y;
    }
```

```
        public override string ToString()
        {
            return String.Format("Point(x: {0}, y:{1})", x, y);
        }
    }

    abstract class Shape
    {
        public abstract double Area { get; }

        public abstract double Perimeter { get; }

        public abstract bool Contains(double x, double y);
    }

    class Rect : Shape
    {
        public Point2d TopLeft { get; set; }
        public double Width { get; set; }
        public double Length { get; set; }
        public override double Area
        {
            get
            {
                return Width * Length;
            }
        }

        public override double Perimeter
        {
            get
            {
                return 2 * ( Width + Length);
            }
        }

        public override bool Contains(double x, double y)
        {
            double subx = x - TopLeft.x;
            double suby = y - TopLeft.y;
            if (subx >= 0 && subx <= Length && suby >= 0 && suby <= Width)
                return true;
            else
                return false;
        }

        public override string ToString()
        {
            return string.Format("Rect [ TopLeft: {0}, Length: {1}, Width: {2}, Area: {3}]",
TopLeft, Length, Width, Area);
        }

    }
```

```
    class Circle : Shape
    {
        public Point2d Center { get; set; }
        public double Radius { get; set; }

        public Circle()
            : this(Point2d.Origin, 1)
        {
        }

        public Circle(Point2d center, double radius)
        {
            Center = center;
            Radius = radius;
        }
        public override double Area
        {
            get { return Math.PI * Radius * Radius; }
        }
        public override double Perimeter
        {
            get { return 2 * Math.PI * Radius; }
        }

        public override bool Contains(double x, double y)
        {
            double distance = Math.Sqrt((Center.x - x) * (Center.x - x) + (Center.y - y) *
(Center.y - y));
            return distance <= Radius;
        }

        public override string ToString()
        {
            return string.Format("Circle [ Center: {0}, Radius: {1}, Area: {2}]", Center,
Radius, Area);
        }
    }

    class GraphicsTest
    {
        public static void Main(string[] argv)
        {
            Point2d p1 = new Point2d();
            Console.WriteLine(p1);

            Point2d p2 = new Point2d(1, 1);
            Console.WriteLine(p2);

            Rect rect = new Rect { TopLeft = p1, Length = 2, Width = 3 };
            Console.WriteLine(rect);

            Circle circle = new Circle(p2, 2);
            Console.WriteLine(circle);
        }
    }
```

3

```
IList <Racer> racers = Racer.GetChampions();
var query = from r in racers
            where r.Country == "UK"
            orderby r.Wins descending
            select r;
foreach (Racer r in query)
{
    Console.WriteLine("{0:A}", r);
}
Console.WriteLine();
query = racers.Where(r => r.Country == "Brazil")
    .OrderByDescending(r => r.Wins);
foreach (Racer r in query)
{
    Console.WriteLine("{0:A}", r);
}
Console.WriteLine();

var query2 = from r in racers
        where r.Wins > 25
        orderby r.Wins descending
        select r;
foreach (Racer r in query2)
{
    Console.WriteLine("{0:A}", r);
}
Console.WriteLine();

foreach (Racer r in racers)
{
    if (r.Country == "UK")
        r.Country = "United Kingdom";
}
foreach (Racer r in query2)
{
    Console.WriteLine("{0:A}", r);
}
Console.WriteLine();

var query3 = from r in racers
        where r.Starts >= 100 && r.Wins >= 20
        select r;
foreach (Racer r in query3)
{
    Console.WriteLine("{0:A}", r);
}
Console.WriteLine();

var query4 = from r in racers
             where r.Country == "Germany"
             select r.Wins;
Console.WriteLine(query4.Sum());
Console.WriteLine();
```

```
var query5 = from r in racers
             select new
             {
                 Name = r.FirstName + " " + r.LastName,
                 Ratio = r.Wins / r.Starts
             };
foreach (var value in query5)
{
    Console.WriteLine(value);
}
```