단국대학교 멀티미디어공학 HCI 프로그래밍 II 중간고사 (2010년 가을학기) 2010년 10월 22일

학과 _____ 학번 _____ 이름 _____

중간고사

담당교수: 단국대학교 멀티미디어공학전공 박경신

- 답은 반드시 답안지에 기술할 것. 공간이 부족할 경우 반드시 답안지 몇 쪽의 뒤에 있다고 명기한 후 기술할 것. 그 외의 경우의 답안지 뒤쪽이나 연습지에 기술한 내용은 답안으로 인정 안 함. 답에는 반드시 네모를 쳐서 확실히 표시할 것.
- 답안지에 학과, 학번, 이름 외에 본인의 암호를 기입하면 성적공고시 학번대신 암호를 사용할 것임.

1. 다음 프로그램의 실행 결과를 써라. 지역 변수와 전역 변수에 주의할 것. (10점)

```
class MethodTest
{
        int i = 5;
        void Add1(int a, int b, ref int c)
        {
            int i = 2;
            ++i;
            Console.WriteLine("Add1 : i=" + i);
            Console.WriteLine("Add1 : a={0} b={1} c={2}", a, b, c);
            Add2();
            c = Add3(a, b);
        }

        void Add2()
        {
            ++i;
            Console.WriteLine("Add2 : i=" + i);
        }

        int Add3(int a, int b)
        {
            ++i;
            Console.WriteLine("Add3 : i=" + i);
            int c = a + b;
            Console.WriteLine("Add3 : a={0} b={1} c={2}", a, b, c);
            return c;
        }
```

학과 _____ 학번 _____ 이름 _____

```
        static void Main()
        {
            MethodTest p = new MethodTest ();
            Console.WriteLine("Main1: i=" + p.i);
            p.i++;
            Console.WriteLine("Main2: i=" + p.i);
            int a = 3, b = 4, c = 6;
            p.Add1(a, b, ref c);
            Console.WriteLine("Main3: i=" + p.i);
    }
}
```

출력결과:

```
Main1 : i=5
Main2 : i=6
Add1 : i=3
Add1 : a=3 b=4 c=6
Add2 : i=7
Add3 : i=8
Add3 : a=3 b=4 c=7
Main3 : i=8
```

2. 다음은 피보나치 수열을 재귀함수로 구현한 프로그램이다. 이 프로그램이 실행되는 결과를 써라. (10점)

```
class RecursiveTest
{
        static int Fibonacci(int n)
        {
            if (n == 0 || n == 1)
                return n;
            else
                return Fibonacci(n-1) + Fibonacci(n-2);
        }

        static void Main()
        {
            int f = Fibonacci(1);
            Console.WriteLine(f);
            f = Fibonacci(3);
            Console.WriteLine(f);
```

학과 _____ 학번 _____ 이름 _____

```
            f = Fibonacci(4);
            Console.WriteLine(f);
            f = Fibonacci(5);
            Console.WriteLine(f);
            f = Fibonacci(7);
            Console.WriteLine(f);
    }
}
```

출력결과:

```
1
2
3
5
13
```

3. 다음은 2차원 배열을 이용한 3목 게임보드 클래스이다. 보드 클래스의 IsFullBoard()와 IsEmpty(int i, int j) 메소드를 구현하라. (10점)

```
class Board
{
        public char[,] matrix;                    // board (3x3 2차원 배열)

        public Board()
        {
            InitBoard();
        }

        // 보드 초기화
        public void InitBoard()
        {
            matrix = new char[3, 3];              // 3x3 2차원 배열
            for (int i = 0; i < matrix.GetLength(0); i++)
                for (int j = 0; j < matrix.GetLength(1); j++)
                    matrix[i, j] = ' ';
        }
```

학과 _____ 학번 _____ 이름 _____

```
        // 보드 전체가 다 채워졌는지?
        public bool IsFullBoard()
        {
            foreach (char c in matrix)
            {
                if (c == ' ')
                {
                    return false;
                }
            }
            return true;
        }

        // 보드에 해당 위치가 비었는지?
        public bool IsEmpty(int i, int j)
        {
            if (matrix[i, j] == ' ')
                return true;
            return false;
        }
}
```

4. C#의 boxing과 unboxing이 무엇인지 간단히 설명하라. (5점)

- **박싱 (boxing)**
  - Value 형식의 자료형을 Reference 형으로 바꾸는 것
- **언박싱 (unboxing)**
  - Reference 형식의 자료형을 Value 형으로 바꾸는 것
  - Boxing 할 때 명시적인 변환은 필요하지 않지만 Unboxing을 할 때에는 필요함

```
// boxing
int a = 137;
object o1 = a;
object o2 = o1;
// unboxing
int b = (int)o2;
```

학과 _____ 학번 _____ 이름 _____

5. C#의 CTS(Common Type System)는 값 형식 (Value Type)과 참조 형식 (Reference Type)으로 나뉜다. C#의 기본형 (enum, int, float, class, char, interface, object, struct, string 등)을 예로 들어서, 값 형식과 참조 형식을 간단히 설명하라. (5점)

- **값 형식 (value type)**
  - 값 형식은 직접 값을 메모리에 갖고 있으며 동일한 객체를 가리킬 수 없기 때문에 한 값의 변경이 다른 것에 영향을 줄 수 없음
  - Bool, byte, char, decimal, float, int, struct, 등
- **참조 형식 (reference type)**
  - 참조 형식은 값을 갖지 않고 메모리에 있는 어떤 값을 가리킴 즉 동일한 객체를 가리키는 것이 가능하며 변경된 값이 다른 참조 형 값에 영향을 줄 수 있음
  - Class, delegate, interface, object, string, array

6. 다음은 값 형식과 참조 형식을 비교하는 프로그램이다. 이 프로그램이 실행되는 출력결과를 써라. (10점)

```
class DoubleClass
{
        public double value;
        public DoubleClass (double value)
        {
            this.value = value;
        }
}

class TypeTest
{
    static void Main()
    {
            double val1 = 3.5;
            double val2 = 3.5;
            if (val1 == val2)
                Console.WriteLine("val1 == val2");
            else
                Console.WriteLine("val1 != val2");

            double val3 = val1;
            if (val1 == val3)
                Console.WriteLine("val1 == val3");
            else
                Console.WriteLine("val1 != val3");

            object val4 = val1;
            object val5 = val2;
            if (val4 == val5)
                Console.WriteLine("val4 == val5");
            else
                Console.WriteLine("val4 != val5");

            double val6 = (double)val4;
            if (val1 == val6)
                Console.WriteLine("val1 == val6");
            else
                Console.WriteLine("val1 != val6");
```

학과 _____ 학번 _____ 이름 _____

```
            if (val2 == val6)
                Console.WriteLine("val2 == val6");
            else
                Console.WriteLine("val2 != val6");

            DoubleClass d1 = new DoubleClass(3.5);
            DoubleClass d2 = new DoubleClass(3.5);
            if (d1 == d2)
                Console.WriteLine("d1 == d2");
            else
                Console.WriteLine("d1 != d2");

            DoubleClass d3 = d1;
            if (d3 == d1)
                Console.WriteLine("d1 == d3");
            else
                Console.WriteLine("d1 != d3");

            object d4 = d1;
            object d5 = d2;
            if (d4 == d5)
                Console.WriteLine("d4 == d5");
            else
                Console.WriteLine("d4 != d5");

            d3.value = 4.5;
            if (d1.value == d2.value)
                Console.WriteLine("d1.value == d2.value");
            else
                Console.WriteLine("d1.value != d2.value");

            if (d1.value == d3.value)
                Console.WriteLine("d1.value == d3.value");
            else
                Console.WriteLine("d1.value != d3.value");

            Console.WriteLine();
        }
    }
```

출력결과:

```
val1 == val2
val1 == val3
val4 != val5
val1 == val6
val2 == val6
ref1 != ref2
ref1 == ref3
ref4 != ref5
ref1.value != ref2.value
ref1.value == ref3.value
```

학과 _____ 학번 _____ 이름 _____

7. 값 형식과 참조 형식에 대한 Pass-by-value와 Pass-by-reference 메소드 매개 변수 전달 방식을 본인이 직접 예를 들어 간단히 설명하라. (10점)

- **Pass value type by value (값 형식을 값에 의한 전달 방식)**
  - 값 형식(value type)은 값을 copy하여 매개변수에 전달
  - static void Square1(int x) {
    x*= x; Console.WriteLine("The value inside: {0}", x); // 함수내부에선 변경된 값이 출력 }
    static void Main() {
    int i =5; Square1(i);
    Console.WriteLine("i={0}", i); // 그러나, 함수에서 변경된 값이 외부에 반영되지 않음
    }
- **Pass reference type by value (참조 형식을 값에 의한 전달 방식)**
  - 참조형식(reference type)은 참조를 copy하여 매개변수에 전달
  - static void ChangeArray1(int[] arr) {
    arr[0] = 888; // 메인의 myArray를 같이 참조하고 있으므로 myArray[0]=888로 변경
    arr = new int[5] {-3, -1, -2, -3, -4}; // 함수내부에서 arr는 새로 지정
    Console.WriteLine("The value inside: arr[0]={0}", arr[0]); // 새로 지정된 배열로 출력 }
    static void Main() {
    int[] myArray ={1, 4, 5}; ChangeArray1(myArray);
    Console.WriteLine("myArray[0]={0}", myArray[0]); // myArry[0]=888로 출력
    }
- **Pass value type by reference (값 형식을 참조에 의한 전달 방식)**
  - ref 키워드를 사용하여, 값 형식(value type)을 참조에 의한 전달방식을 사용
  - static void Square2(ref int x) {
    x*= x; Console.WriteLine("The value inside: {0}", x); // 메인 i를 참조하므로 값이 변경 }
    static void Main() {
    int i =5; Square2(i);
    Console.WriteLine("i={0}", i); // 함수에서 변경된 값이 외부로 반영됨
    }
- **Pass reference type by reference (참조 형식을 참조에 의한 전달 방식)**
  - ref 키워드를 사용하여, 참조 형식(reference type)을 참조에 의한 전달방식을 사용
  - static void ChangeArray2(ref int[] arr) {
    arr[0] = 888; // 메인의 myArray를 같이 참조하고 있으므로 myArray[0]=888로 변경
    arr = new int[5] {-3, -1, -2, -3, -4}; // 그러나 원본 배열이 다시 변경
    Console.WriteLine("The value inside: arr[0]={0}", arr[0]); // 새로 지정된 배열로 출력 }
    static void Main() {
    int[] myArray ={1, 4, 5}; ChangeArray2(myArray);
    Console.WriteLine("myArray[0]={0}", myArray[0]); // myArry[0]=-3로 출력
    }

학과 _____ 학번 _____ 이름 _____

8. 메소드 오버로딩 (method overloading)과 메소드 오버라이딩 (method overriding)을 간단히 설명하고, 다음 PointTest 프로그램에서 오버로딩과 오버라이딩의 예를 모두 찾아서 적어라. (10점)

```
class Point
{
        protected static int count = 0;
        protected int x, y;
        public Point() : this(0, 0) { }
        public Point(int x, int y) { this.x = x; this.y = y; count++; }
        ~Point() { count--; }
        public void SetPosition(int x, int y) { this.x = x; this.y = y; }
        public void Move(int x, int y) { this.x += x; this.y += y; }
        public virtual void Print() { Console.WriteLine("X={0} Y={1}", x, y); }
        public static int GetCount() { return count; }
        public override string ToString() { return (String.Format("({0}, {1})", x, y)); }
}
class Point3D : Point
{
        protected int z;
        public Point3D() : this(0, 0, 0) { }
        public Point3D(int x, int y, int z) : base(x, y) { this.z = z; }
        ~Point3D() { }
        public void SetPosition(int x, int y, int z) { base.SetPosition(x, y); this.z = z; }
        public void Move(int x, int y, int z) { base.Move(x, y); this.z += z; }
        public override void Print() { Console.WriteLine("X={0} Y={1} Z={2}", x, y, z); }
        public override string ToString() { return (String.Format("({0}, {1}, {2})", x, y, z)); }
}
class PointTest
{
        static void Swap(ref Point p1, ref Point p2)
        {
            Point p;
            p = p1;
            p1 = p2;
            p2 = p;
        }
        static void Swap(ref Point3D p1, ref Point3D p2)
        {
            Point3D p;
            p = p1;
            p1 = p2;
            p2 = p;
        }
        static void Main(string[] args)
        {
            Point p1 = new Point();
            Point p2 = new Point();
            p1.SetPosition(10, 10);
            p1.Move(20, 50);
            p1.Print();
            p2.SetPosition(20, 30);
            p2.Print();
            Swap(ref p1, ref p2);
            Console.WriteLine(p1);
            Console.WriteLine(p2);
            Console.WriteLine(Point.GetCount());
```

학과 _____ 학번 _____ 이름 _____

```
            Point3D p3 = new Point3D();
            Point3D p4 = new Point3D();
            p3.SetPosition(10, 20);
            p3.Print();
            p4.SetPosition(30, 40, 50);
            p4.Print();
            Swap(ref p3, ref p4);
            Console.WriteLine(p3);
            Console.WriteLine(p4);
            Console.WriteLine(Point.GetCount());

            Point p5 = p3;
            p5.Print();
            Point p6 = new Point3D(10,20,30);
            p6.Print();
            Point3D p7 = (Point3D) p5;
            Console.WriteLine(p7);
        }
}
```

- **메소드 오버로딩 (method overloading)**
  - 동일한 함수명에 매개변수가 다른 함수를 둘 이상 정의하는 것으로, 동일한 함수 기능을 수행하지만 다른 매개변수의 경우를 처리할 때 사용
  - PointTest 클래스에 Swap(ref Point p1, ref Point p2)와 Swap(ref Point3D p1, ref Point3D p2)
  - Point 클래스에서 생성자 오버로딩 Point()와 Point(int x, int y)
  - Point3D 클래스에서 생성자 오버로딩 Point3D()와 Point3D(int x, int y, int z)
  - Point와 Point3D 클래스에서 SetPosition(int x, int y)와 SetPosition(int x, int y, int z)
  - Point와 Point3D 클래스에서 Move(int x, int y)와 Move(int x, int y, int z)

- **메소드 오버라이딩 (method overriding)**
  - 상속받은 파생 클래스에서 동일한 함수명에 동일한 매개변수로 정의하여 함수를 재정의하는 것으로, 상속되어진 함수의 기능을 변경해서 재사용하고 싶을 때 사용
  - Virtual/override 메소드는 이름, 접근 제한자, 반환 값, 및 매개변수 리스트가 동일해야 함
  - Point와 Point3D 클래스에서 virtual void Print() 와 override void Print()
  - Point 클래스에서 override string ToString()
  - Point3D 클래스에서 override string ToString()
  - PointTest 클래스에서 p5.Print()경우 p5가 실제 Point3D이므로, Point3D의 Print()가 호출됨

9. static field/method를 간단히 설명하고, 8번의 PointTest 프로그램에서 static의 예를 모두 찾아서 적어라. (5점)

- **Static (정적) field**
  - Static field 는 전역 데이터로 클래스당 하나만 할당됨
  - 클래스 외부에서는 클래스명.정적필드명 형태로 사용
  - Point 클래스에서 static int count

- **Static (정적) method**
  - Static method 는 클래스 메소드나 정적 필드 조작을 위한 메소드
  - 클래스 외부에서는 클래스명.정적메소드명 형태로 사용
  - Static method는 instance field는 접근 불가능
  - Point 클래스에서 static int GetCount()
  - PointTest 클래스에서 static void Swap(ref Point p1, ref Point p2)와 static void Swap(ref Point3D p1, ref Point3D p2)

학과 _____ 학번 _____ 이름 _____

10. 다형성 (Polymorphism)을 간단히 설명하고, 8번의 PointTest 프로그램에서 그 예를 모두 찾아서 적어라. (5점)

- **다형성**

  - 상위클래스에서 선언된 메소드를 하위클래스에서 재구현 (override)하고 실행시간에 새로 정의된 override된 멤버의 내용으로 처리(late binding)

```
Point p5 = p3;
p5.Print();          // p5는 p3를 참조, p3가 Point3D이므로 Point3D의 Print 호출 –late binding
Point p6 = new Point3D(10, 20);
p6.Print();          // p6는 Point3D객체를 생성한 것이므로 Point3D의 Print 호출 –late binding
Point3D p7 = (Point3D) p5;
Console.WriteLine(p7);     // p7는 p5를 downcasting한 것으로 Point3D의 ToString 호출
```

11. 8번의 PointTest 프로그램에서 Point클래스와 Point3D 클래스에 IEquatable<T> 인터페이스 (interface)를 상속(inherit)받아 구현(implement)하라. (10점)

```
interface IEquatable<T>
{
    bool Equals(T obj); // 모든 멤버필드가 일치하면 true 아니면 false
}

class Point : IEquatable<Point>
{

// 중간 생략
        public bool Equals(Point other)
        {
            if (this.x == other.x && this.y == other.y)
                return true;
            return false;
        }


}

class Point3D : Point, IEquatable<Point3D>
{

// 중간 생략
        public bool Equals(Point3D other)
        {
            if (this.x == other.x && this.y == other.y && this.z == other.z)
                return true;
            return false;
        }


}
```

학과 _____ 학번 _____ 이름 _____

12. 다음은 대리자(Delegate)를 사용하여 원하는 온도변환 방법을 지정하여 사용할 수 있는 프로그램이다. 빈 칸을 채워라. (10점)

```
public delegate double TemperatureConversion(double from);

class DelegateTest
{
        static ___double___ FromFahrenheitToCelsius(___double F___)
        {
            return (F - 32.0) * (5.0 / 9.0);
        }
        static ___double___ FromCelsiusToFahrenheit(___double C___)
        {
            return ((9.0 / 5.0) * C + 32.0);
        }
        // factory method for returning delegate
        static public TemperatureConversion GetConversionMethod(string conversionType)
        {
            TemperatureConversion conversion;
            switch (conversionType)
            {
                case "1":
                    conversion = new TemperatureConversion(FromCelsiusToFahrenheit);
                    break;
                case "2":
                    conversion = ___new TemperatureConversion(FromFahrenheitToCelsius);___
                    break;
                default:
                    throw new ArgumentException("알수없는  conversion type: "+conversionType);
            }
            return conversion;
        }
        // user input to temperature and type of conversion
        static public void GetUserInput(out double temperature, out string conversionType)
        {
            Console.WriteLine("Temperature Converter");
            Console.WriteLine("1: Celsius => Fahrenheit");
            Console.WriteLine("2: Fahrenheit => Celsius");
            Console.Write("원하는  온도변환을  입력하시오  (1 or 2):   ");
            conversionType = ___Console.ReadLine();___
            Console.Write("변환하고자하는  온도를  입력하시오:   ");
            temperature = ___Double.Parse(Console.ReadLine());___
        }
        static public void Main()
        {
            double temperature;
            string conversionType;
            GetUserInput(out temperature, out conversionType);
            TemperatureConversion doConversion = GetConversionMethod(conversionType);
            double result = doConversion(temperature);
            Console.WriteLine("\n conversionType={0} Temperature={1} Result={2}\n",
                                    conversionType, temperature, result);
        }
}
```

-끝-