Họ và Tên: Nguyễn Thanh Kiên.

MSSV: 22110092.

Bài 1:

Xây dựng lớp Dog và Cat với các thuộc tính Name, Age, Height và Weight và thực hiện các chức năng sau:

. Tạo một danh sách gồm 3 con chó và 2 con mèo ✓

. Thực hiện nhập và xuất thông tin của các con vật ✓

. Bắt lỗi khi người dùng nhập tuổi âm hoặc lớn hơn 20 ✓

. Bắt lỗi khi người dùng nhập chuỗi hay ký tự (không phải số) các thông tin Height và Weight ✓

Bài 2:

. Xây dựng lớp Football, Tenis, và lớp Volleyball với các thông tin số lượng người chơi, thời gian chơi, loại bóng chơi (tự nghĩ thêm thông tin khác) kèm hai phương thức nhập và xuất ✓

. Suy nghĩ lớp cha Sport chứa các thông tin gì để các lớp con có thể kế thừa các thông tin đó ✓

. Viết hàm nhập xuất danh sách các môn thể thao dựa vào mối quan hệ kế thừa trên (Khuyến khích dùng Đa Hình) ✓

Làm thêm:

Bài 1: Tính BMI và chế độ ăn hằng ngày.

Bài 2: Sử dụng Interface để định nghĩa hành vi cụ thể cho môn thể thao

Bài Lab: 01.

1. Xây dựng lớp Dog và Cat với các thuộc tính Name, Age, Height và Weight và thực hiện các chức năng sau:

a. Tạo một danh sách gồm 3 con chó và 2 con mèo

b. Thực hiện nhập và xuất thông tin của các con vật

c. Bắt lỗi khi người dùng nhập tuổi âm hoặc lớn hơn 20

d. Bắt lỗi khi người dùng nhập chuỗi hay ký tự (không phải số) các thông tin Height và Weight

Trước hết, em xây dựng 2 lớp Dog và Cat mẫu với các thông tin điền sẵn trong hàm Main.

Hàm Main:

```
0 references
static void Main(string[] args)
{
    Dog dog = new Dog("kiki", 10, 50, 20);
    dog.DisplayInfo();
    Console.WriteLine(dog.ToString());

    Cat cat = new Cat("Meo", 9, 25, 10);
    cat.DisplayInfo();
    Console.WriteLine(cat.ToString());

    Console.ReadLine();
}
```

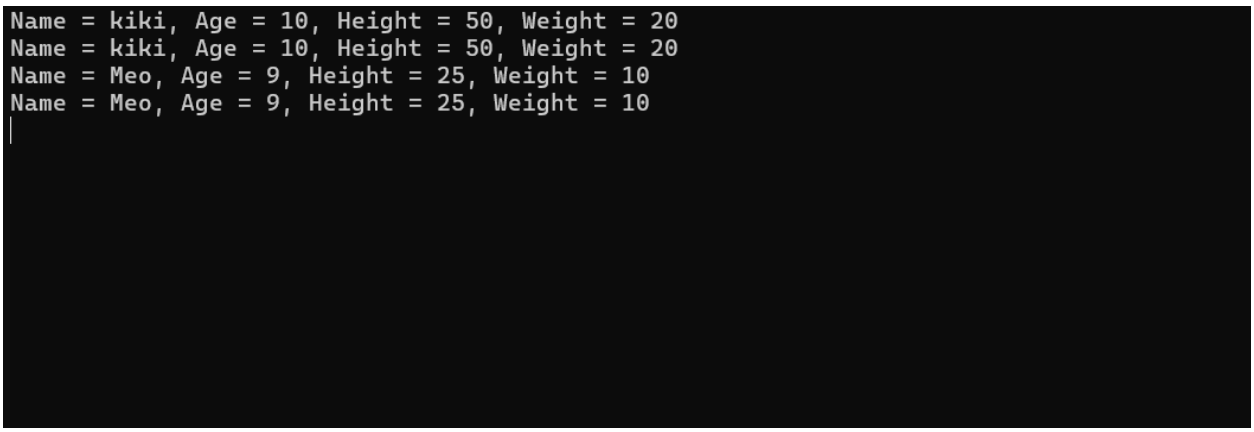
Lớp Dog:

```
public class Dog
{
    private string name = "Not Assgined";
    private int age = 0;
    private int height = 0;
    private int weight = 0;
    1 reference
    public Dog(string name = "", int age = 0, int height = 0, int weight = 0)
    {
        this.name = name;
        this.age = age;
        this.height = height;
        this.weight = weight;
    }
    1 reference
    public void DisplayInfo()
    {
        Console.WriteLine("Name = {0}, Age = {1}, Height = {2}, Weight = {3}", name, age, height, weight);
    }
    1 reference
    public override string ToString()
    {
        return "Name = " + name + ", Age = " + age + ", Height = " + height + ", Weight = " + weight;
    }
}
```

Lớp Cat:

```
public class Cat
{
    private string name = "Not Assgined";
    private int age = 0;
    private int height = 0;
    private int weight = 0;
    1 reference
    public Cat(string name = "", int age = 0, int height = 0, int weight = 0)
    {
        this.name = name;
        this.age = age;
        this.height = height;
        this.weight = weight;
    }
    1 reference
    public void DisplayInfo()
    {
        Console.WriteLine("Name = {0}, Age = {1}, Height = {2}, Weight = {3}", name, age, height, weight);
    }
    1 reference
    public override string ToString()
    {
        return "Name = " + name + ", Age = " + age + ", Height = " + height + ", Weight = " + weight;
    }
}
}
```

Kết quả sau khi chạy:



Ở đây, thông tin được Xuất ra 2 lần đối với mỗi đối tượng (Dog, Cat) bởi vì em đã gọi 2 phương thức DisplayInfo() và ToString() để xuất dữ liệu ra màn hình, tuy vậy chúng ta vẫn cần phải gọi phương thức ToString để ghi đè kiểu dữ liệu theo cách chúng ta muốn.

Tạo một danh sách gồm 3 con chó và 2 con mèo

Em tạo một mảng Object 5 phần tử, nhập Dog trước Cat, dùng cơ chế Boxing và unboxing để xuất đúng thông tin là chó hay là mèo, và dùng (obj[i].GetType() == typeof(Dog)) để kiểm tra object xem là chó hay là mèo.

```
        static void Main(string[] args)
        {
            int nDog = 3;
            int nCat = 2;

            Dog dog;
            Cat cat;

            object[] obj = new object[nDog + nCat];

            for (int i = 0; i < nDog; i++)
            {
                dog = new Dog();
                dog.InputInfo();
                obj[i] = (Dog)dog;
            }

            for (int i = 0; i < nCat; i++)
            {
                cat = new Cat();
                cat.InputInfo();
                obj[nDog + i] = (Cat)cat;
            }
```

```
            for (int i = 0; i < nDog + nCat; i++)
            {
                if (obj[i].GetType() == typeof(Dog))
                {
                    Console.WriteLine("Information of Dog:");
                    dog = (Dog)obj[i];
                    dog.DisplayInfo();
                }
                else
                {
                    Console.WriteLine("Information of Cat:");
                    cat = (Cat)obj[i];
                    cat.DisplayInfo();
                }
            }
            Console.ReadLine();
        }
    }
}
```

Kết quả sau khi chạy:

```
Input Dog Name:kiki1
Input Dog Age:1
Input Dog Height:2
Input Dog Weight:3
Input Dog Name:kiki2
Input Dog Age:1
Input Dog Height:2
Input Dog Weight:3
Input Dog Name:kiki3
Input Dog Age:1
Input Dog Height:2
Input Dog Weight:3
Input Cat Name:meomeo1
Input Cat Age:1
Input Cat Height:2
Input Cat Weight:3
Input Cat Name:meomeo2
Input Cat Age:1
Input Cat Height:2
Input Cat Weight:3
Information of Dog:
Name = kiki1 , Age = 1, Height = 2, Weight = 3
Information of Dog:
Name = kiki2, Age = 1, Height = 2, Weight = 3
Information of Dog:
Name = kiki3, Age = 1, Height = 2, Weight = 3
Information of Cat:
Name = meomeo1, Age = 1, Height = 2, Weight = 3
Information of Cat:
Name = meomeo2, Age = 1, Height = 2, Weight = 3
```

Thực hiện nhập và xuất thông tin của các con vật

Em dùng hàm Console.Readline() để nhập thông tin các con vật,để nhập số thì em phải nhập chuỗi và dùng age = int.Parse(str) để chuyển chuỗi sang số, tuy vậy, em không thể khai báo str thêm 1 lần nào nữa vì sẽ gây ra lỗi trùng tên biến, vì vậy đối với thuộc tính height và weight, em sử dụng

height = int.Parse(Console.ReadLine()); để có thể chuyển chuỗi sang số và xuất luôn.

Lớp Dog:

```
public class Dog
{
    1 reference
    public void InputInfo()
    {
        Console.Write("Input Dog Name:");
        name = Console.ReadLine();
        Console.Write("Input Dog Age:");
        string str = Console.ReadLine();
        age = int.Parse(str);
        Console.Write("Input Dog Height:");
        height = int.Parse(Console.ReadLine());
        Console.Write("Input Dog Weight:");
        weight = int.Parse(Console.ReadLine());
    }
```

Lớp Cat:

```
public class Cat
{
    1 reference
    public void InputInfo()
    {
        Console.Write("Input Cat Name:");
        name = Console.ReadLine();
        Console.Write("Input Cat Age:");
        string str = Console.ReadLine();
        age = int.Parse(str);
        Console.Write("Input Cat Height:");
        height = int.Parse(Console.ReadLine());
        Console.Write("Input Cat Weight:");
        weight = int.Parse(Console.ReadLine());
    }
```

Hàm Main:

```
static void Main(string[] args)
{
    Dog dog = new Dog("kiki", 10, 50, 20);
    dog.InputInfo();
    dog.DisplayInfo();
    Console.WriteLine(dog.ToString());

    Cat cat = new Cat("Meo", 9, 25, 10);
    cat.InputInfo();
    cat.DisplayInfo();
    Console.WriteLine(cat.ToString());

    Console.ReadLine();
}
```

Kết quả sau khi chạy:

```
Input Dog Name:kiki
Input Dog Age:12
Input Dog Height:45
Input Dog Weight:46
Name = kiki, Age = 12, Height = 45, Weight = 46
Name = kiki, Age = 12, Height = 45, Weight = 46
Input Cat Name:meomeo
Input Cat Age:12
Input Cat Height:11
Input Cat Weight:111
Name = meomeo, Age = 12, Height = 11, Weight = 111
Name = meomeo, Age = 12, Height = 11, Weight = 111
```

Bắt lỗi khi người dùng nhập tuổi âm hoặc lớn hơn 20

Ở đây em chỉ nhập 1 Dog và 1 Cat cho đỡ mất thời gian.

Em sử dụng các kiến thức về Exception để bắt lỗi, với điều kiện tuổi <=0 & >20. Em sử dụng khai báo public class NegativeNumException : Exception và if (age <= 0 || age > 20)

throw new NegativeNumException();

Trước hết em khai báo Lớp NegativeNumException trong namespace MainData

```
namespace MainData
{
    2 references
    public class NegativeNumException: Exception
    {
        0 references
        public NegativeNumException() { }
        0 references
        public NegativeNumException(string message): base(message) { }
    }
```

Em viết hàm InputAge() riêng và trong đó đệ quy khi nhập sai, tức là em sẽ nhập sai nhiều lần cho đến đúng.

Lớp Dog:

```
public class Dog
{
    1 reference
    public void InputInfo()
    {
        Console.Write("Input Dog Name:");
        name = Console.ReadLine();
        InputAge();
        Console.Write("Input Dog Height:");
        height = int.Parse(Console.ReadLine());
        Console.Write("Input Dog Weight:");
        weight = int.Parse(Console.ReadLine());
    }
    2 references
    public void InputAge()
    {
        bool isCompleted = false;

        Console.Write("Input Dog Age:");
        string str = Console.ReadLine();
        try
        {
            age = int.Parse(str);
            if (age <= 0 || age > 20)
                throw new NegativeNumException();
            isCompleted = true;
        }
```

```
        catch (NegativeNumException)
        {
            Console.Write("Negative Age is not accepted. Please Reinput a number:");
        }
        finally
        {
            if (!isCompleted)
            {
                InputAge();
            }
        }
    }
```

Lớp Cat:

```
public class Cat
{
    1 reference
    public void InputInfo()
    {
        Console.Write("Input Cat Name:");
        name = Console.ReadLine();
        InputAge();
        Console.Write("Input Cat Height:");
        height = int.Parse(Console.ReadLine());
        Console.Write("Input Cat Weight:");
        weight = int.Parse(Console.ReadLine());
    }
    2 references
    public void InputAge()
    {
        bool isCompleted = false;

        Console.Write("Input Cat Age:");
        string str = Console.ReadLine();
        try
        {
            age = int.Parse(str);
            if (age <= 0 || age > 20)
                throw new NegativeNumException();
            isCompleted = true;
        }
        catch (NegativeNumException)
        {
            Console.Write("Negative Age is not accepted. Please Reinput a number:");
        }
        finally
        {
            if (!isCompleted)
            {
                InputAge();
            }
        }
    }
```

Kết quả sau khi chạy:

```
Input Dog Name:kiki
Input Dog Age:-2
Negative Age is not accepted. Please Reinput a number:Input Dog Age:-3
Negative Age is not accepted. Please Reinput a number:Input Dog Age:4
Input Dog Height:3
Input Dog Weight:2
Name = kiki, Age = 4, Height = 3, Weight = 2
Name = kiki, Age = 4, Height = 3, Weight = 2
Input Cat Name:meomeo
Input Cat Age:32
Negative Age is not accepted. Please Reinput a number:Input Cat Age:234
Negative Age is not accepted. Please Reinput a number:Input Cat Age:24
Negative Age is not accepted. Please Reinput a number:Input Cat Age:19
Input Cat Height:234
Input Cat Weight:24
Name = meomeo, Age = 19, Height = 234, Weight = 24
Name = meomeo, Age = 19, Height = 234, Weight = 24
```

Bắt lỗi khi người dùng nhập chuỗi hay ký tự (không phải số) các thông tin Height và Weight

Em tiếp tục sử dụng kiến thức của Exception và sử dụng FormatException khi chuyển kiểu sai đối với phương thức height và weight, đề nghị nhập lại khi exception xảy ra. Em tạo thêm 2 hàm InputHeight() và InputWeight() để bắt lại FormatException, và sử dụng vòng lặp while (!isHeightValid) để nhập tới khi nào đúng định dạng số thì thôi.

Lớp Dog:

```
public class Dog
{
    1 reference
    public void InputInfo()
    {
        Console.Write("Input Dog Name:");
        name = Console.ReadLine();

        InputAge();

        InputHeight();

        InputWeight();
    }
```

```
    public void InputHeight()
    {
        bool isHeightValid = false;
        while (!isHeightValid)
        {
            Console.Write("Input Dog Height: ");
            string heightInput = Console.ReadLine();
            try
            {
                height = int.Parse(heightInput);
                isHeightValid = true;
            }
            catch (FormatException)
            {
                Console.WriteLine("Invalid input. Please enter a valid number for height.");
            }
        }
    }
```

```
    public void InputWeight()
    {
        bool isWeightValid = false;
        while (!isWeightValid)
        {
            Console.Write("Input Dog Weight: ");
            string weightInput = Console.ReadLine();
            try
            {
                weight = int.Parse(weightInput);
                isWeightValid = true;
            }
            catch (FormatException)
            {
                Console.WriteLine("Invalid input. Please enter a valid number for weight.");
            }
        }
    }
```

Lớp cat:

```
public class Cat
{
    1 reference
    public void InputInfo()
    {
        Console.Write("Input Cat Name:");
        name = Console.ReadLine();

        InputAge();

        InputHeight();

        InputWeight();
    }
```

```
    public void InputAge()
    {
        bool isCompleted = false;

        Console.Write("Input Cat Age:");
        string str = Console.ReadLine();
        try
        {
            age = int.Parse(str);
            if (age <= 0 || age > 20)
                throw new NegativeNumException();
            isCompleted = true;
        }
        catch (NegativeNumException)
        {
            Console.Write("Negative Age is not accepted. Please Reinput a number:");
        }
        finally
        {
            if (!isCompleted)
            {
                InputAge();
            }
        }
    }
```

```
public void InputHeight()
{
    bool isHeightValid = false;
    while (!isHeightValid)
    {
        Console.Write("Input Cat Height: ");
        string heightInput = Console.ReadLine();
        try
        {
            height = int.Parse(heightInput);
            isHeightValid = true;
        }
        catch (FormatException)
        {
            Console.WriteLine("Invalid input. Please enter a valid number for height.");
        }
    }
}
```

Kết quả sau khi chạy:

```
Input Dog Name:kiki
Input Dog Age:20
Input Dog Height: abd
Invalid input. Please enter a valid number for height.
Input Dog Height: abd
Invalid input. Please enter a valid number for height.
Input Dog Height: 20
Input Dog Weight: oal
Invalid input. Please enter a valid number for weight.
Input Dog Weight: an
Invalid input. Please enter a valid number for weight.
Input Dog Weight: 30
Name = kiki, Age = 20, Height = 20, Weight = 30
Name = kiki, Age = 20, Height = 20, Weight = 30
Input Cat Name:meomeo
Input Cat Age:20
Input Cat Height: als
Invalid input. Please enter a valid number for height.
Input Cat Height: asl
Invalid input. Please enter a valid number for height.
Input Cat Height: 2
Input Cat Weight: alsfn
Invalid input. Please enter a valid number for weight.
Input Cat Weight: ashdfoh
Invalid input. Please enter a valid number for weight.
Input Cat Weight: 3
Name = meomeo, Age = 20, Height = 2, Weight = 3
Name = meomeo, Age = 20, Height = 2, Weight = 3
```

Ý tưởng thêm: Tính BMI và chế độ ăn hằng ngày.

```
public double CalculateBMI()
{
    double heightInMeters = height / 100.0;
    return weight / (heightInMeters * heightInMeters);
}
1 reference
public void DisplayBMI()
{
    double bmi = CalculateBMI();
    Console.WriteLine("BMI: {0:F2}", bmi);
    if (bmi < 18.5)
        Console.WriteLine("Underweight");
    else if (bmi < 25)
        Console.WriteLine("Normal weight");
    else
        Console.WriteLine("Overweight");
}
0 references
public void CalculateDailyFood()
{
    double foodAmount;
    if (weight < 10)
        foodAmount = 0.05 * weight;
    else if (weight < 20)
        foodAmount = 0.04 * weight;
    else
        foodAmount = 0.03 * weight;
    Console.WriteLine("{0} should eat {1:F2} kg of food per day.", name, foodAmount);
}
```

Em sử dụng if else cơ bản

Kết quả sau khi chạy:

```
Input Dog Name:kiki
Input Dog Age:10
Input Dog Height: 100
Input Dog Weight: 12
Name = kiki, Age = 10, Height = 100, Weight = 12
BMI: 12.00
Underweight
Name = kiki, Age = 10, Height = 100, Weight = 12
Input Cat Name:20
Input Cat Age:15
Input Cat Height: 130
Input Cat Weight: 6
Name = 20, Age = 15, Height = 130, Weight = 6
BMI: 3.55
Underweight
20 should eat 0.30 kg of food per day.
Name = 20, Age = 15, Height = 130, Weight = 6
```

2. Xây dựng lớp Football, Tenis, và lớp Volleyball với các thông tin số lượng người chơi, thời gian chơi, loại bóng chơi (tự nghĩ thêm thông tin khác) kèm hai phương thức nhập và xuất. Suy nghĩ lớp cha Sport chứa các thông tin gì để các lớp con có thể kế thừa các thông tin đó. Viết hàm nhập xuất danh sách các môn thể thao dựa vào mối quan hệ kế thừa trên (Khuyến khích dùng Đa Hình) (Xem lại lý thuyết để làm)

Xây dựng lớp Football, Tenis, và lớp Volleyball với các thông tin số lượng người chơi, thời gian chơi, loại bóng chơi kèm hai phương thức nhập và xuất.

(em thêm 2 thông tin là số trận đấu (match) và số banh dùng (ball)

Hàm Main:

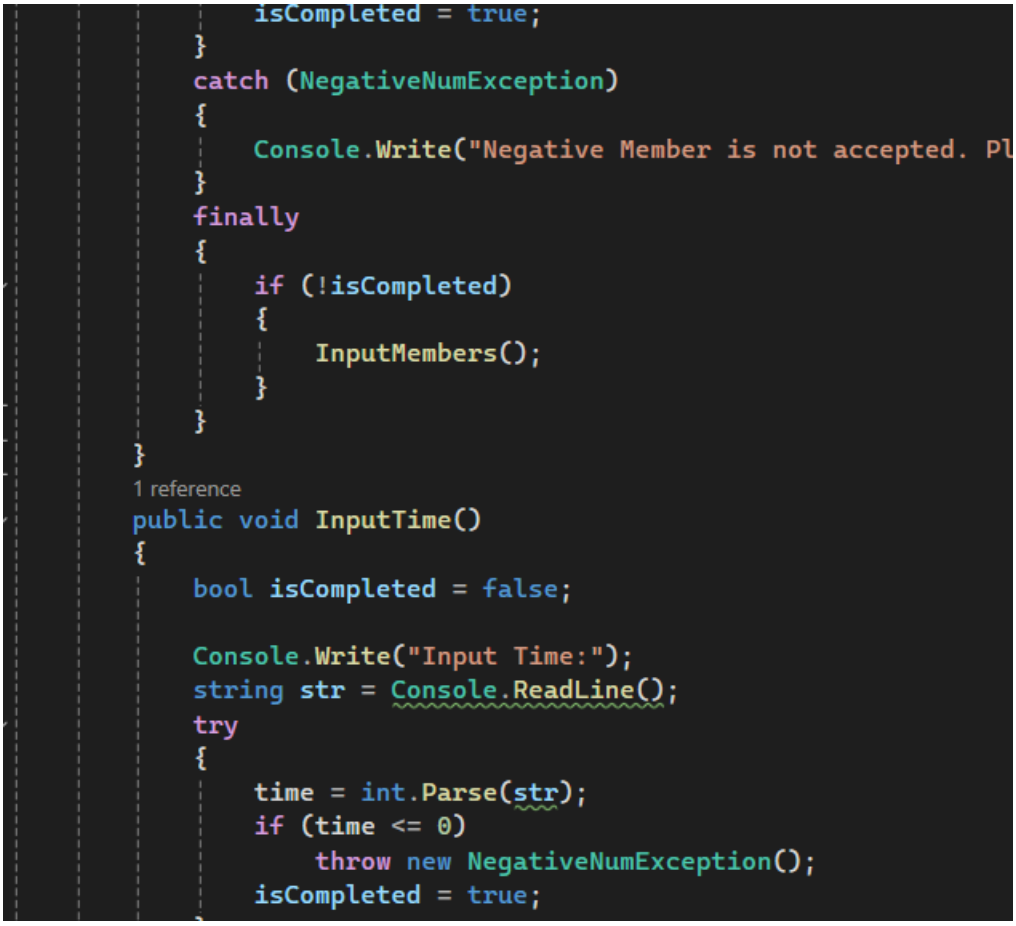
```
static void Main(string[] args)
{
    Football football = new Football(20, 50, "A", 3, 2);
    football.InputInfo();
    football.DisplayInfo();
    Console.WriteLine(football.ToString());

    Tennis tennis = new Tennis(20, 50, "B", 3, 2);
    tennis.InputInfo();
    tennis.DisplayInfo();
    Console.WriteLine(tennis.ToString());

    Volleyball volleyball = new Volleyball(20, 50, "C", 3, 2);
    volleyball.InputInfo();
    volleyball.DisplayInfo();
    Console.WriteLine(volleyball.ToString());
}
```

Bắt Exception:

```
public class NegativeNumException : Exception
{
    12 references
    public NegativeNumException() { }
    0 references
    public NegativeNumException(string message) : base(message) { }
}
```

Lớp Football:

```
public class Football
{
    1 reference
    public void InputInfo()
    {
        InputMembers();

        InputTime();

        Console.Write("Input Type:");
        type = Console.ReadLine();

        InputMatch();

        InputBall();
    }
    4 references
    public void InputMembers()
    {
        bool isCompleted = false;

        Console.Write("Input Members:");
        string str = Console.ReadLine();
        try
        {
            members = int.Parse(str);
            if (members <= 0)
                throw new NegativeNumException();
```



```
            isCompleted = true;
        }
        catch (NegativeNumException)
        {
            Console.Write("Negative Member is not accepted. Pl
        }
        finally
        {
            if (!isCompleted)
            {
                InputMembers();
            }
        }
    }
    1 reference
    public void InputTime()
    {
        bool isCompleted = false;

        Console.Write("Input Time:");
        string str = Console.ReadLine();
        try
        {
            time = int.Parse(str);
            if (time <= 0)
                throw new NegativeNumException();
            isCompleted = true;
```

```
        }
        catch (NegativeNumException)
        {
            Console.Write("Negative Time is not accepted. Please Rein
        }
        finally
        {
            if (!isCompleted)
            {
                InputMembers();
            }
        }
    }

    2 references
    public void InputMatch()
    {
        bool isCompleted = false;

        Console.Write("Input Match:");
        string str = Console.ReadLine();
        try
        {
            match = int.Parse(str);
            if (match <= 0)
                throw new NegativeNumException();
            isCompleted = true;
        }
        catch (NegativeNumException)
```

```
            Console.Write("Negative match is not accepted. Please Reinput a number:");
        }
        finally
        {
            if (!isCompleted)
            {
                InputMatch();
            }
        }
    }
    1 reference
    public void InputBall()
    {
        bool isCompleted = false;

        Console.Write("Input Ball:");
        string str = Console.ReadLine();
        try
        {
            ball = int.Parse(str);
            if (ball <= 0)
                throw new NegativeNumException();
            isCompleted = true;
        }
        catch (NegativeNumException)
```

```
            Console.Write("Negative Ball is not accepted. Please Reinput a number:");
        }
        finally
        {
            if (!isCompleted)
            {
                InputMembers();
            }
        }
    }
    private int members = 0;
    private int time = 0;
    private string type = "Not assgined";
    private int match = 0;
    private int ball = 0;
    1 reference
    public Football(int members = 0, int time = 0, string type = "", int match = 0, int ball = 0)
    {
        this.members = members;
        this.time = time;
        this.type = type;
        this.match = match;
        this.ball = ball;
    }
    1 reference
    public void DisplayInfo()
    {
        Console.WriteLine("Members = {0}, Time = {1}, Type = {2}, Match = {3}, Ball = {4}", members, time, type, match, ball);
    }
```

```
    }
    1 reference
    public override string ToString()
    {
        return "Football: Members = " + members + ", Time = " + time + ", Type = " + type + ", Match = " + match + ", Ball = " + ball;
    }
}
3 references
```

Các lớp Tennis và Volleyball em làm tương tự.

Kết quả sau khi chạy:

```
Input Members:10
Input Time:50
Input Type:a
Input Match:5
Input Ball:3
Members = 10, Time = 50, Type = a, Match = 5, Ball = 3
Football: Members = 10, Time = 50, Type = a, Match = 5, Ball = 3
Input Members:4
Input Time:60
Input Type:b
Input Match:3
Input Ball:10
Members = 4, Time = 60, Type = b, Match = 3, Ball = 10
Tennis: Members = 4, Time = 60, Type = b, Match = 3, Ball =10
Input Members:12
Input Time:30
Input Type:c
Input Match:2
Input Ball:1
Members = 12, Time = 30, Type = c, Match = 2, Ball = 1
Volleyball: Members = 12, Time = 30, Type = c, Match = 2, Ball =1
```

Suy nghĩ lớp cha Sport chứa các thông tin gì để các lớp con có thể kế thừa các thông tin đó: lớp cha Sport sẽ chứa các thông tin mà đề bài yêu cầu (số lượng người chơi members, thời gian chơi time, loại bóng chơi type. Các

lớp con sẽ kế thừa các đặc tính của lớp cha Sport có thể thêm các đặc tính riêng cho các môn thể thao như UsingType(tay, chân, vợt), Ball(số lượng bóng chơi), Match(số lượng trận đấu).

Viết hàm nhập xuất danh sách các môn thể thao dựa vào mối quan hệ kế thừa trên (Khuyến khích dùng Đa Hình) (Xem lại lý thuyết để làm)

Hàm main:

```
0 references
static void Main(string[] args)
{

    List<Sport> sports = new List<Sport>();

    sports.Add(new Football(20, 90, "Football Ball"));
    sports.Add(new Tennis(2, 30, "Tennis Ball"));
    sports.Add(new Volleyball(12, 60, "Volleyball Ball"));


    foreach (var sport in sports)
    {
        sport.InputInfo();
    }

    foreach (var sport in sports)
    {
        sport.DisplayInfo();
    }
}
}
```

Lớp cha Sport:

```
public class Sport
{
    public int members;
    public int time;
    public string ballType;

    3 references
    public Sport(int members, int time, string ballType)
    {
        this.members = members;
        this.time = time;
        this.ballType = ballType;
    }

    4 references
    public virtual void InputInfo()
    {
        Console.WriteLine("Nhập thông tin cho Sport (cơ bản):");
        Console.Write("Members: ");
        members = int.Parse(Console.ReadLine());
        Console.Write("Play Time: ");
        time = int.Parse(Console.ReadLine());
        Console.Write("Ball Type: ");
        ballType = Console.ReadLine();
    }
```

```
    4 references
    public virtual void DisplayInfo()
    {
        Console.WriteLine("Sport - Members: {0}, Time: {1}, Ball Type: {2}", members, time, ballType);
    }
}
```

Lớp con Football

```
public class Football : Sport
{
    1 reference
    public Football(int members, int time, string ballType) : base(members, time, ballType) { }

    2 references
    public override void InputInfo()
    {
        Console.WriteLine("Football:");
        Console.Write("Members: ");
        members = int.Parse(Console.ReadLine());
        Console.Write("Play Time: ");
        time = int.Parse(Console.ReadLine());
        Console.Write("Ball Type: ");
        ballType = Console.ReadLine();
    }

    2 references
    public override void DisplayInfo()
    {
        Console.WriteLine("Football - Members: {0}, Time: {1}, Ball Type: {2}", members, time, ballType);
    }
}
```

Lớp con Tennis:

```
public class Tennis : Sport
{
    1 reference
    public Tennis(int members, int time, string ballType) : base(members, time, ballType) { }

    2 references
    public override void InputInfo()
    {
        Console.WriteLine("Tennis:");
        Console.Write("Members: ");
        members = int.Parse(Console.ReadLine());
        Console.Write("Play Time: ");
        time = int.Parse(Console.ReadLine());
        Console.Write("Ball Type: ");
        ballType = Console.ReadLine();
    }

2 references
    public override void DisplayInfo()
    {
        Console.WriteLine("Tennis - Members: {0}, Time: {1}, Ball Type: {2}", members, time, ballType);
    }
}
```

Lớp con Volleyball:

```
public class Volleyball : Sport
{
    1 reference
    public Volleyball(int members, int time, string ballType) : base(members, time, ballType) { }

    2 references
    public override void InputInfo()
    {
        Console.WriteLine("Volleyball:");
        Console.Write("Members: ");
        members = int.Parse(Console.ReadLine());
        Console.Write("Play Time: ");
        time = int.Parse(Console.ReadLine());
        Console.Write("Ball Type: ");
        ballType = Console.ReadLine();
    }

    2 references
    public override void DisplayInfo()
    {
        Console.WriteLine("Volleyball - Members: {0}, Time: {1}, Ball Type: {2}", members, time, ballType);
    }
}
```

Kết quả sau khi chạy

```
Football:
Members: 24
Play Time: 90
Ball Type: football
Tennis:
Members: 2
Play Time: 30
Ball Type: Tennis ball
Volleyball:
Members: 12
Play Time: 60
Ball Type: Volleyball ball
Football - Members: 24, Time: 90, Ball Type: football
Tennis - Members: 2, Time: 30, Ball Type: Tennis ball
Volleyball - Members: 12, Time: 60, Ball Type: Volleyball ball
```

Giải thích:

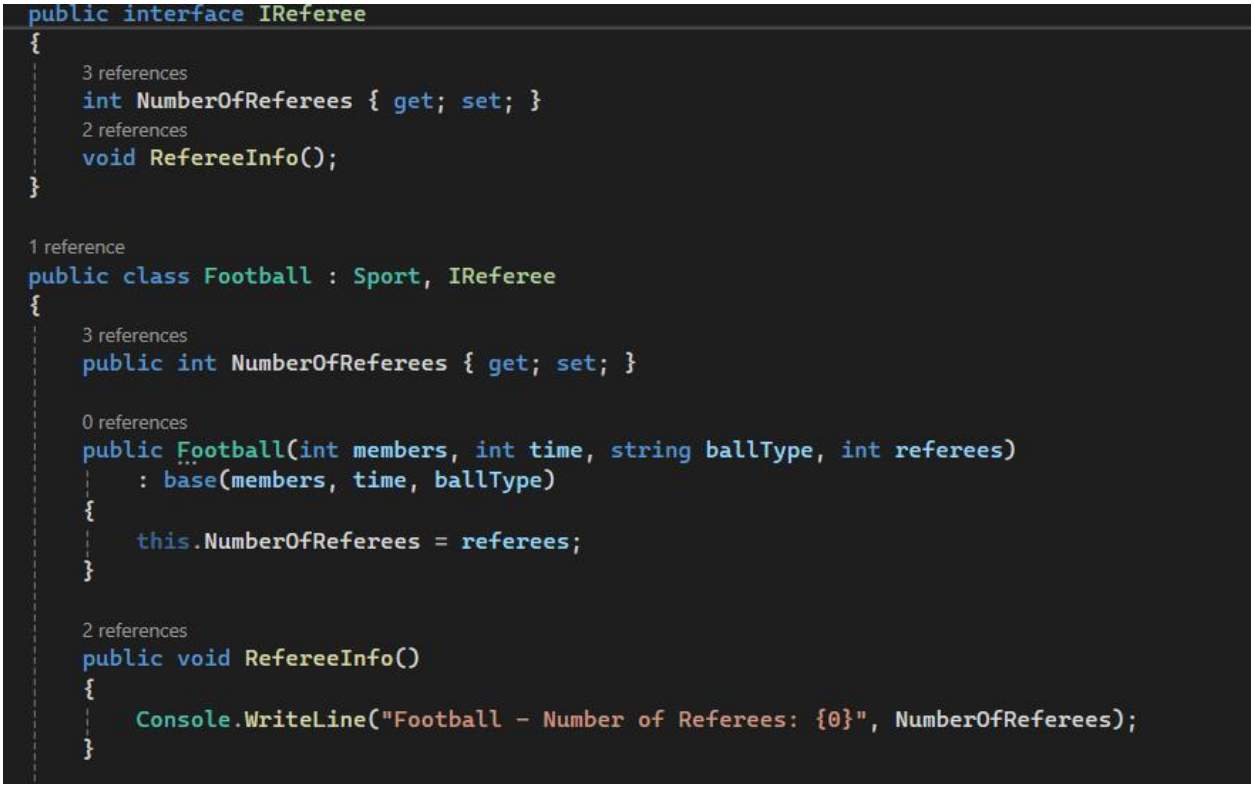
foreach (var sport in sports) { sport.InputInfo(); // Đa hình: phương thức InputInfo của từng lớp con sẽ được gọi }

foreach (var sport in sports) { sport.DisplayInfo(); // Đa hình: phương thức DisplayInfo của từng lớp con sẽ được gọi }

Khi gọi sport.InputInfo() và sport.DisplayInfo() trong vòng lặp foreach, chương trình sẽ gọi đúng phương thức của từng lớp con do tính đa hình, ngay cả khi biến sport có kiểu dữ liệu là Sport.

Cho phép các lớp con triển khai các hành vi riêng biệt thông qua phương thức ảo (virtual) và ghi đè (override).

Làm thêm: Sử dụng Interface để định nghĩa hành vi cụ thể cho môn thể thao



```
public interface IReferee
{
    3 references
    int NumberOfReferees { get; set; }
    2 references
    void RefereeInfo();
}

1 reference
public class Football : Sport, IReferee
{
    3 references
    public int NumberOfReferees { get; set; }

    0 references
    public Football(int members, int time, string ballType, int referees)
        : base(members, time, ballType)
    {
        this.NumberOfReferees = referees;
    }

    2 references
    public void RefereeInfo()
    {
        Console.WriteLine("Football - Number of Referees: {0}", NumberOfReferees);
    }
```

```
    3 references
    public override void DisplayInfo()
    {
        base.DisplayInfo();
        RefereeInfo();
    }
}
```