講義メモ
・条件演算子の補足をしてから、p.092「エラーメッセージを読み解こう②」に進みます

【補足】条件演算子（p.039）：再掲載

・if-else構文による判断結果の代入を、3項演算子で表現できる仕掛け
・例：int a = 2, b = 3, max = (a > b) ? a : b; //a > b ならaをでなければbを代入
・書式：（条件式などのbool型の式）？ 式がtrueの時の値 ： 式がfalseの時の値
・これを用いると、p,081 Chap2_5_1のif-elseの5行が1行になる
 → Debug.Log((age < 20) ? "未成年" : "成人");
・なお「:」以降の省略は不可（if文でいうifのみ記述は不可）

提出フォロー：ミニ演習 mini091

・上記の通り、p,081 Chap2_5_1を書き換えてみよう

作成例

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class mini091 : MonoBehaviour {
    public int age; //パブリック変数
    void Start() {
        //ageの値が20未満なら"未成年"を、でなければ"成人"を表示
        Debug.Log((age < 20) ? "未成年" : "成人");
    }
    void Update() {

    }
}
```

ミニ演習 mini091a

・p,085 Chap2_6_1を条件演算子で書き換えてみよう
・構文：（条件文①）？ 値① ：（(条件文②) ？ 値② ： 値③）

作成例

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class mini091a : MonoBehaviour {
    public int age; //パブリック変数
    void Start() {
        //ageの値が20未満なら"未成年"を、でなければ、
```

```
        //ageの値が65未満なら"成人"を、でなければ"高齢者"を表示
        Debug.Log((age < 20) ? "未成年" : (age < 65) ? "成人" : "高齢者");
    }
    void Update() {

    }
}
```

ミニ演習 mini091b

・int型のパブリック変数aとbの差を表示しよう
・差とは大きい方ー小さい方とする
・同値ならどちらから引いてもゼロなので単純に引き算でOK

作成例

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class mini091b : MonoBehaviour {
    public int a, b; //パブリック変数
    void Start() {
        //差を表示
        Debug.Log((a > b) ? a - b : b - a); //大きい方ー小さい方
    }
    void Update() {

    }
}
```

ミニ演習 mini091c

・int型のパブリック変数a,b,cの最大値を表示しよう
・aとbの最大値は「(a > b) ? a : b」で得られる
・この値とcの大きい方が最大値になる

作成例1

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class mini091c : MonoBehaviour {
    public int a, b, c; //パブリック変数
    void Start() {
        int max = (a > b) ? a : b; //a,bの最大を得る
        Debug.Log((max > c) ? max : c); //a,bの最大とcを比較
    }
```

```
    void Update() {

    }
}
```

作成例2

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class mini091c : MonoBehaviour {
    public int a, b, c; //パブリック変数
    void Start() {
        Debug.Log((a > b) ? (a > c) ? a : c : //a>bならaとcで最大決定
                            (b > c) ? b : c); //a<bならbとcで最大決定
    }
    void Update() {

    }
}
```

p.092 エラーメッセージを読み解こう

・ミスの場所とエラーの表示場所がずれるのでわかりづらいのが「カッコの非対応」
・特に、足りないカッコを適当に帳尻合わせすると状況が悪化しやすい
・このタイプのエラーが出たら、インデントをしっかり行い、対応をていねいに確認しよう
・例：mini091cで「public class mini091c : MonoBehaviour {」の直後に「}」がある場合
 → Startメソッド全体がエラーになり、C#側では13個のエラーが表示される
 Unity側では2個のエラーが表示される
 ① Top-level statements must precede namespace and type declarations.
 ② Type or namespace definition, or end-of-file expected
・例：mini091cで「void Start() {」の直後に「}」がある場合
 → Startメソッド以降がエラーになり、C#側では40個のエラーが表示される
 Unity側では19個のエラーが表示される
・テキストに記述の「予期しない'}'」の例は、常にこうなるとは限らない
・例：mini091cで最後の「}」を消してしまった場合
 → C#側でもUnity側でも「} が必要です」「} expected」というエラーになる
・テキストに記述の「予期しないファイルの終端」の例も、常にこうなるとは限らない

p.092 エラーメッセージを読み解こう（つづき）

・また、C#およびUnityのバージョンアップによって、テキストとは異なるメッセージが表示される場合がある
・p.092「else ifの半角空きを忘れた場合」では、C#側で「;が必要です」「現在のコンテキストに'elseif'という名前は存在しません」の2エラー、Unity側で「; expected」の1エラーになる

p.093 Visual Studioでカッコの対応を確認する

・「()」「{}」「[]」「""」において始まりの方の1文字前をクリックすることで対応する相手がハイライト表示される

p.094 復習ドリル 問題1

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class chap2_9_1 : MonoBehaviour {
    public int age;
    void Start() {
        if (age < 6) {
            Debug.Log("幼児");
        }
    }
    void Update() {

    }
}
```

p.094 復習ドリル 問題2

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class chap2_9_2 : MonoBehaviour {
    public int age;
    void Start() {
        if (age <= 5 || age >= 65) { //「&&」ではなく「||」
            Debug.Log("幼児と高齢者");
        }
    }
    void Update()
    {

    }
}
```

テキスト正誤 p.095の枠内のプログラム

【誤】if("") {
【正】if(string型の変数 != "")

ミニ演習 mini095

・p.095の枠内のプログラムには誤りがありエラーになる
・これを修正して、string型のパブリック変数sが空かどうかを判断するプログラムにしよう

作成例

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class mini095 : MonoBehaviour {
    public string s; //パブリック変数
    void Start() {
        if (s != "") {
            Debug.Log("空ではない");
        } else {
            Debug.Log("空です");
        }
    }
    void Update() {

    }
}
```

ミニ演習 mini095a

・p.091 Chap2_7_3を条件演算子で書き直そう
・「9」をint型のパブリック変数にする
・2のべき乗である場合は「2のべき乗である」と表示しよう
・ただし、プログラム中で「2のべき乗で」は1回のみ用いること

作成例

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class mini095a : MonoBehaviour {
    public int n; // パブリック変数
    void Start() {
        string s = "2のべき乗で"; //連結用の前半文字列
        Debug.Log(!Mathf.IsPowerOfTwo(n) ? s + "はない" : s + "ある");
    }
    void Update() {

    }
}
```

追加演習「ジャンケン」その1 janken1

・int型のパブリック変数handを定義する
・インスペクタに説明「手をどうぞ(0=グー,1=チョキ,2=パー)」をHeader機能で表示する

・handの値に応じて「あなたの手＝」に続けて「グー」または「チョキ」または「パー」を表示

作成例

```
using UnityEngine;
public class janken1 : MonoBehaviour {
    [Header("手をどうぞ(0=グー,1=チョキ,2=パー)")]
    public int hand;
    void Start() {
        string s = "あなたの手＝";
        Debug.Log(s + ((hand == 0) ? "グー" : (hand == 1) ? "チョキ" : "パー
"));
    }
    void Update() {}
}
```

追加演習「ジャンケン」その2 janken2

・CPU側を作ろう。毎回異なる手を出して欲しいので乱数を使おう
・C#では、System.Randomクラスのインスタンスメソッドを呼ぶと乱数が提供される
・このとき、Next(種類数)を指定することで、0から種類数-1までの、どれかの値を返してくれる
・利用例：

```
System.Random r = new System.Random(); //乱数クラスのオブジェクトを生成(後述)
Debug.Log(r.Next(10)); //0から9までのどれかが表示される
```

・この機能を用いて、0から2までの乱数を得て、対応する手を表示するようにしよう
・乱数の値に応じて「わたしの手＝」に続けて「グー」または「チョキ」または「パー」を表示

作成例

```
using UnityEngine;
public class janken2 : MonoBehaviour {
    [Header("手をどうぞ(0=グー,1=チョキ,2=パー)")]
    public int hand;
    void Start() {
        System.Random r = new System.Random(); //乱数クラスのオブジェクトを生成
        string s = "あなたの手＝";
        Debug.Log(s + ((hand == 0) ? "グー" : (hand == 1) ? "チョキ" : "パー
"));
        string c = "わたしの手＝";
        int cpuh = r.Next(3); //乱数で0,1,2を得る
        Debug.Log(c + ((cpuh == 0) ? "グー" : (cpuh == 1) ? "チョキ" : "パー
"));
    }
    void Update() { }
}
```

提出：追加演習「ジャンケン」その3 janken3

・判定を表示しよう

・例「あなたの勝ち」「わたしの勝ち」「あいこ」
・ヒント：先に「あいこ」を判断すると楽。判定はORで連結した条件ですると良い
　※ 計算式で判定することもできる：(3 + hand - cpuh) % 3 == 2 ならばhandの勝ち

次回予告：「ジャンケン」その3の補足をしてから、p.098「繰り返し文」に進みます