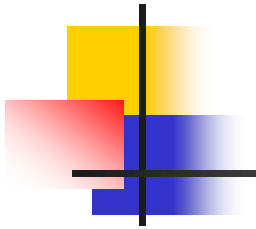
โครงสร้างของโปรแกรมเกมคอมพิวเตอร์

ในการเขียนโปรแกรมเกมคอมพิวเตอร์ ผู้พัฒนาโปรแกรมเกมคอมพิวเตอร์ควรจะต้องพิจารณาถึงข้อจำกัดต่างๆ ของอุปกรณ์ที่นำไปใช้ในการประมวลผลเกมคอมพิวเตอร์ เช่น ขนาดหน่วยความจำ ขนาดเนื้อที่ความจุฮาร์ดิสก์ ขนาดหน่วยความจำการ์ดแสดงผล เป็นต้น ข้อจำกัดดังกล่าว จะมีผลต่อการพัฒนาโปรแกรมเกมคอมพิวเตอร์ การออกแบบโปรแกรมจะพยายามพัฒนาโปรแกรมเกมคอมพิวเตอร์ ให้มีการใช้ทรัพยากรอย่างคุ้มค่าและมีประสิทธิภาพมากที่สุด การเรียนรู้โครงสร้างของโปรแกรมจะช่วยให้ผู้พัฒนาเข้าใจวิธีการ ผลลัพธ์ของโครงสร้างที่ใช้รวมถึงการตรวจสอบความถูกต้องได้



ประกอบด้วย

โครงสร้างการเลือกทำ

โครงสร้างการวนซ้ำ

โครงสร้างการเลือกทำ (Selection)

ในการพัฒนาโปรแกรมเกมคอมพิวเตอร์ โครงสร้างการเลือกทำ(Selection) มีความจำเป็นมากในการพัฒนาโปรแกรม ดังนั้นการศึกษาการทำงาน คำสั่งที่ถูกต้องของโครงสร้างจึงมีความสำคัญเป็นอย่างยิ่ง ในเกมคอมพิวเตอร์จะมีเหตุการณ์ในการเลือกหรือตัดสินใจของการทดสอบเงื่อนไขแล้วเลือกว่าจะทำหรือไม่ทำ ก่อนที่จะไปทำงานอื่นต่อไป ผู้พัฒนาโปรแกรมจึงต้องเข้าใจโครงสร้างของการเลือกทำ ตัวอย่างเช่น โปรแกรมเกมคอมพิวเตอร์ สำหรับเกมขับรถ เกิดเหตุการณ์ที่จะต้องมีการตรวจสอบเงื่อนไขเรื่องของเวลา โดยมีค่าจำกัดของเวลาถ้ารถที่บังคับอยู่ยังไม่เข้าเส้นชัย จะทำให้เวลาหมด Time Over และสถานะจบเกม Game Over เป็นต้น

โครงสร้างการเลือกทำ (Selection)

if (this expression is true)

{

statement (code block)

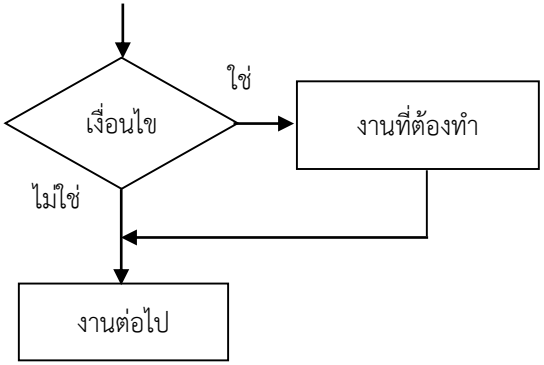
}

elseif

{

another code block

}



โครงสร้างการเลือกทำ (Selection)

```
using UnityEngine;
      using System.Collections;
      public class  IfStatements:MonoBehaviour
      {
         float coffeeTemperature        = 85.0f;
         float hotLimitTemperature      = 70.0f;
         float coldLimitTemperature     = 40.0f;

         void Update ()
         {
             if(Input.GetKeyDown(KeyCode.Space))
                 TemperatureTest();
                 coffeeTemperature-=Time.deltaTime *5f;
         }

         voidTemperatureTest()
         {
             // If the coffee's temperature is greater than the hottest drinking temperature...
             if(coffeeTemperature>hotLimitTemperature)
             {
                 // ... do this.
                 print("Coffee is too hot.");
             }
             // If it isn't, but the coffee temperature is less than the coldest drinking temperature...
```

โครงสร้างการเลือกทำ (Selection)

```
        else if(coffeeTemperature<coldLimitTemperature)
        {
            //…do this.
            print("Coffee is too cold.");
        }
        //If it is neither of those then…
        else
        {
            //…do this.
            print("Coffee is just right.");
        }
    }
}
```

โครงสร้างการเลือกทำ (Selection)

จากตัวอย่างโปรแกรมการใช้งานคำสั่ง if statement เริ่มต้นจะทำการเปรียบเทียบตัวแปรอุณหภูมิ coffeeTemperatureกับตัวแปร hotLimitTemperature ถ้าค่ามากกว่าให้แสดงคำว่า "Coffee is too hot." แต่ถ้าไม่มากกว่าให้ตรวจสอบกับตัวแปร coldLimitTemperature ถ้าค่า coffeeTemperature น้อยกว่าให้แสดงคำว่า "Coffee is too cold." แต่ถ้าไม่น้อยกว่าให้แสดงคำว่า “Coffee is just right.”

โครงสร้างการเลือกทำ (Selection)

คำสั่ง if statement จะทำงานเมื่อการทดสอบเงื่อนไขเป็นจริง และจะไม่ทำงานถ้าการทดสอบเงื่อนไขออกมาเป็นเท็จ หรือจะไปทำในส่วนของที่อยู่ภายใต้คำสั่ง else และจะทำงานในต่อไป โครงสร้างแบบเลือกทำในโปรแกรมเกมคอมพิวเตอร์ หรือเรียกแบบมีเงื่อนไขยังมีคำสั่ง switch statements ที่สามารถเป็นการทำงานแบบโครงสร้างแบบเลือกทำเช่นกัน ซึ่งจะมี case ให้เลือกทำอาจจะหลายทางเลือก คำที่ต้องใช้ประกอบด้วย switch, case, break และ default โดย คำสั่ง break มีไว้สำหรับหยุดการทำงาน คำสั่ง default มีไว้สำหรับสร้างทางเลือกในกรณีที่ไม่ตรงเงื่อนไขใดๆ ใน case

โครงสร้างการเลือกทำ (Selection)

```
using UnityEngine;
using System.Collections;

public class ConversationScript : MonoBehaviour
{
    public int intelligence = 5;


    void Greet()
    {
        switch (intelligence)
        {
        case 5:
            print ("Why hello there good sir! Let me teach you about
Trigonometry!");
         break;
        case 4:
            print ("Hello and good day!");
            break;
```

โครงสร้างการเลือกทำ (Selection)

```
case 3:
            print("Whadya want?");
          break;
        case 2:
            print ("Grog SMASH!");
            break;
        case 1:
        print ("Ulg, glib, Pblblblblb");
            break;
        default:
            print ("Incorrect intelligence level.");
            break;
      }
    }
}
```

โครงสร้างการเลือกทำ (Selection)

จากตัวอย่างเป็นการทำงานโดยใช้คำสั่ง switch โดยมีการรับค่าตัวแปรที่ชื่อว่า intelligence มี 5 ทางเลือกดังต่อไปนี้

กรณีที่ 1 พิมพ์คำว่า "Ulg, glib, Pblblblblb"หยุดการทำงาน

กรณีที่ 2 พิมพ์คำว่า "Grog SMASH!"หยุดการทำงาน

กรณีที่ 3 พิมพ์คำว่า "Whadya want?"หยุดการทำงาน

กรณีที่ 4 พิมพ์คำว่า "Hello and good day!"หยุดการทำงาน

กรณีที่ 5 พิมพ์คำว่า "Why hello there good sir! Let me teach you about Trigonometry!"

หยุดการทำงาน

กรณีที่ไม่เข้าเงื่อนไขใดเลย พิมพ์คำว่า "Incorrect intelligence level."หยุดการทำงาน

โครงสร้างการวนซ้ำ (Loop)

โปรแกรมเกมคอมพิวเตอร์ โครงสร้างการวนซ้ำ คือ การทำงานที่ทราบจำนวนรอบหรือมีการกำหนดจำนวนรอบ การทำงานซ้ำๆ กัน การใช้คำสั่งวนซ้ำจะเป็นการช่วยให้ตรวจสอบการทำงานของโปรแกรมได้ง่ายขึ้น มีการเขียนโปรแกรมที่สั้นลงในกรณีที่มีการวนรอบเป็นจำนวนมาก

โปรแกรมเกมคอมพิวเตอร์มีคำสั่งที่ใช้สำหรับโครงสร้างการวนซ้ำ คือ คำสั่ง for, while และ do-while

โครงสร้างการวนซ้ำ (Loop)

1) คำสั่ง for ใช้สำหรับการวนรอบที่รู้จำนวนรอบที่แน่นอนต้องมีตัวแปรนับรอบ เงื่อนไข และการเพิ่มจำนวนรอบ

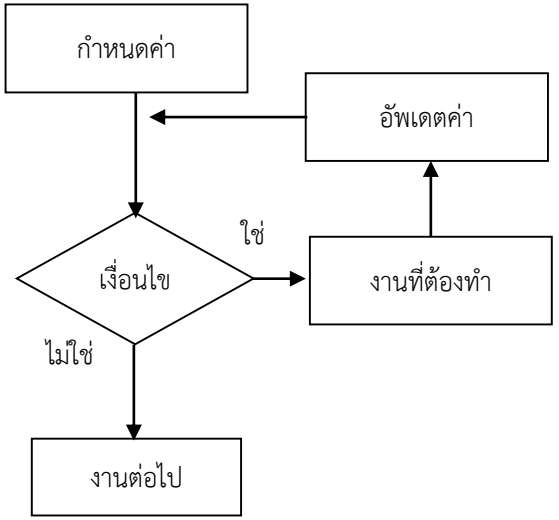
การใช้คำสั่ง For Loop

แนวคิดในการใช้คำสั่ง for เมื่อโปรแกรมมีการวนรอบที่รู้จำนวนรอบที่แน่นอน จะทำงานตามจำนวนตัวแปรนับรอบ และตรวจสอบตามเงื่อนไข ถ้าเป็นเท็จจะหยุดทำงานลำดับต่อไป โดยมีโครงสร้างและผังการทำงานแสดงดังภาพ

โครงสร้างการวนซ้ำ (Loop)

โครงสร้างการวนซ้ำและผังงานของคำสั่ง for loop

```
for (init; condition; update)
{
statement (code block)
}
```



โครงสร้างการวนซ้ำ (Loop)

```
using UnityEngine;
using System.Collections;
public class ForLoop:MonoBehaviour
{
int numEnemies=3;
void Start ()
    {
for(int i=0; i<numEnemies; i++)
        {
Debug.Log("Creating enemy number:"+i);
        }
    }
}
```

โครงสร้างการวนซ้ำ (Loop)

2) คำสั่ง Do While โครงสร้างการทำซ้ำที่ต้องประมวลผลในลูปอย่างน้อย 1 รอบก่อนทดสอบเงื่อนไขออกจากลูป ถ้าเงื่อนไขเป็นจริงจะทำงานและตรวจสอบในรอบต่อไป จนกระทั่งการตรวจสอบเงื่อนไขเป็นเท็จจะหยุดการทำงาน

แนวคิดในการใช้คำสั่ง Do While ทำการประมวลผลอย่างน้อย 1 รอบก่อน ตรวจสอบตามเงื่อนไข ถ้าเป็นเท็จจะหยุดทำงาน และทำงานในลำดับต่อไป โดยมีโครงสร้างและผังการทำงานแสดงดังภาพ

โครงสร้างการวนซ้ำ (Loop)

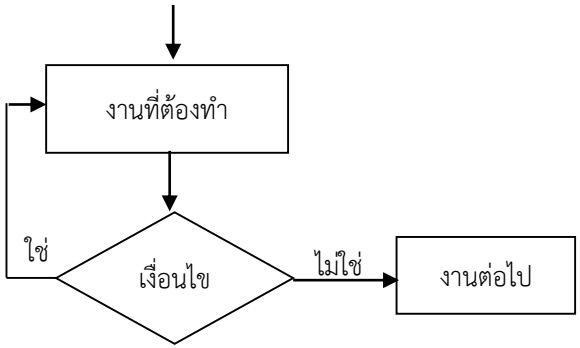
โครงสร้างการวนซ้ำและผังงานของคำสั่ง Do While

{

statement (code block)

}

while (this expression is true)



โครงสร้างการวนซ้ำ (Loop)

```
using UnityEngine;
using System.Collections;
public class DoWhileLoop :MonoBehaviour
{
    void Start()
    {
        bool shouldContinue =false;
        do
        {
            print ("Hello World");
        }while(shouldContinue ==true);
    }
}
```

โครงสร้างการวนซ้ำ (Loop)

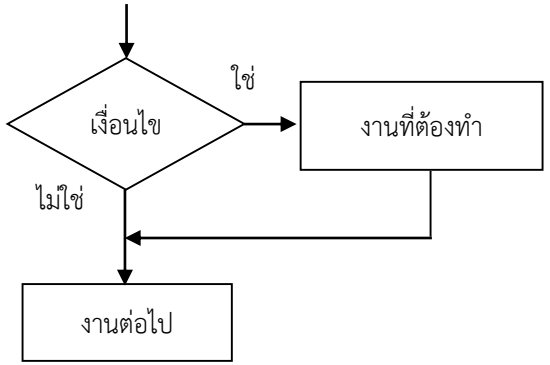
3) คำสั่ง While โครงสร้างที่ต้องตรวจสอบเงื่อนไขก่อนทำในลูป ถ้าเงื่อนไขเป็นจริงจะทำงานและถ้าเงื่อนไขเป็นเท็จจะหยุดการทำงาน

แนวคิดในการใช้คำสั่ง While จะตรวจสอบตามเงื่อนไข ก่อนประมวลผล ถ้าเป็นเท็จจะหยุดทำงาน และทำงานในลำดับต่อไป โดยมีโครงสร้างและผังการทำงานแสดงดังภาพ

โครงสร้างการวนซ้ำ (Loop)

โครงสร้างการวนซ้ำและผังงานของคำสั่ง While loop

```
while (this expression is true)
{
statement (code block)
}
```



โครงสร้างการวนซ้ำ (Loop)

```
using UnityEngine;
using System.Collections;
public class WhileLoop :MonoBehaviour
{
intcupsInTheSink =4;
    void Start ()
    {
        while(cupsInTheSink> 0)
        {
Debug.Log ("I've washed a cup!");
cupsInTheSink--;
        }
    }
}
```