

HW9 Thursday Combi

HW9 Thursday Combi - โปรแกรมแสดงจำนวนวิธีและรูปแบบทั้งหมดที่เป็นไปได้ของปัญหาการเรียงสับเปลี่ยน
รายงานฉบับนี้จะขอแนะนำเสนอการเขียนโปรแกรมแก้โจทย์ปัญหาการเรียงสับเปลี่ยนที่น่าสนใจ 2 ข้อด้วยกัน

Form1

Combi

1. วิธีจัดที่นั่งหน้ากระดาน ให้กับ มช 5 คน, มภ 5 คน โดย มภ ต้องมี มภ นั่งซ้ายๆ
2. เรียงลูกแก้ว 7 ลูก (แดง 2, เขียว 2, ขาว 3) โดย 2 ลูกที่ติดกันต้องสีต่างกัน

Gen Test1

Gen Test2

แนวคิด: หลักการในการวิธีเรียงสับเปลี่ยนในลักษณะเส้นตรง (หน้ากระดาน)
ทั้งหมดคือการใส่ทุกๆตำแหน่งที่เป็นไปได้ ตามเงื่อนไขของโจทย์ ซึ่ง
สามารถนำแนวคิดและการเขียนโปรแกรมของฟังก์ชันเวียนเกิด หรือ Recursive
Function มาช่วยในการคำนวณได้

โจทย์ข้อที่ 1:

หาวิธีในการจัดที่นั่งหน้ากระดานให้กับผู้ชาย 5 คน และผู้หญิง 5 คน โดยผู้หญิงทุกคนจะต้องมีผู้หญิงนั่งข้างๆ

จำนวนวิธีทั้งหมด = 36 วิธี

รูปแบบทั้งหมดที่เป็นไปได้:

Form1

Combi

1. วิธีจัดที่นั่งหน้ากระดาน ให้กับ มช 5 คน, มภ 5 คน โดย มภ ต้องมี มภ นั่งซ้ายๆ
2. เรียงลูกแก้ว 7 ลูก (แดง 2, เขียว 2, ขาว 3) โดย 2 ลูกที่ติดกันต้องสีต่างกัน

Gen Test1

Gen Test2

จำนวนวิธี = 36 วิธี

```
{  
{ G, G, G, G, G, B, B, B, B, B },  
{ G, G, G, B, G, G, B, B, B, B },  
{ G, G, G, B, B, G, G, B, B, B },  
{ G, G, G, B, B, B, G, G, B, B },  
{ G, G, G, B, B, B, B, G, G, B },  
{ G, G, G, B, B, B, B, B, G, G },  
{ G, G, B, G, G, G, B, B, B, B },  
{ G, G, B, B, G, G, G, B, B, B },  
{ G, G, B, B, B, G, G, G, B, B },  
{ G, G, B, B, B, B, G, G, G, B },  
{ G, G, B, B, B, B, B, G, G, G },  
{ B, G, G, G, G, G, B, B, B, B },  
{ B, G, G, G, B, G, G, B, B, B },  
{ B, G, G, G, B, B, G, G, B, B },  
{ B, G, G, G, B, B, B, G, G, B }  
}
```

หลักการทำงานของโปรแกรม:

Method:

<pre>private int findAndCount(List<string> s, string c) { int count = 0; foreach (string e in s) { if (e == c) count++; } return count; }</pre>	<ul style="list-style-type: none">- private int findAndCount(List<string> s, string c) นับและ return ว่าภายใน List<string> s มี string c ปะกุกอยู่กี่ตัว
<pre>private List<string> copy(List<string> from, List<string> to) { foreach(string e in from) { to.Add(e); } return to; }</pre>	<ul style="list-style-type: none">- private List<string> copy(List<string> from, List<string> to) จะทำการ copy ทุกๆ element ใน List<string> from มาเก็บไว้ใน List<string> to
<pre>90 String show_list(List<String> l) 91 { 92 String output = "{ "; 93 for (int i = 0; i < l.Count; i++) 94 { 95 output += l[i].ToString(); 96 if(i!=l.Count-1) output += ", "; 97 } 98 output += " }"; 99 return output; 100 101 102 103 104 } 105 String show_result(List<List<String>> l) 106 { 107 String output = "จำนวนวิธี = " + l.Count.ToString() + " วิธี \r\n"; 108 output += " {\r\n"; 109 for (int i = 0; i < l.Count; i++) 110 { 111 output += " " + show_list(l[i]) + " ,\r\n"; 112 } 113 output += " }\r\n"; 114 return output; 115 116 117 118 } 119</pre>	<ul style="list-style-type: none">- สอง method นี้ จะทำการแสดงผลลัพธ์ผ่านทาง testBox โดยวน loop แสดงผลลัพธ์ทุกรูปแบบที่เกิดขึ้น รวมถึงจำนวนวิธีทั้งหมดที่เป็นไปได้

```

41     private void sitting(List<string> s)
42     {
43         if (s.Count() == 10)
44         {
45             test1Ans.Add(s);
46             return;
47         }
48
49         if (s.Count() > 1 && s.ElementAt(s.Count() - 1) == "G" && s.ElementAt(s.Count() - 2) != "G")
50         {
51             if (findAndCount(s, "G") < 5)
52             {
53                 List<string> tmp = new List<string>();
54                 tmp = copy(s, tmp);
55                 tmp.Add("G");
56                 sitting(tmp);
57             }
58         }
59         else if (s.Count() == 1 && s.ElementAt(s.Count() - 1) == "G")
60         {
61             if (findAndCount(s, "G") < 5)
62             {
63                 List<string> tmp = new List<string>();
64                 tmp = copy(s, tmp);
65                 tmp.Add("G");
66                 sitting(tmp);
67             }
68         }
69
70         else if (s.Count() == 9 && s.ElementAt(s.Count() - 1) != "G")
71         {
72             if (findAndCount(s, "B") < 5)
73             {
74                 List<string> tmp = new List<string>();
75                 tmp = copy(s, tmp);
76                 tmp.Add("B");
77                 sitting(tmp);
78             }
79         }
80         else
81         {
82             if (findAndCount(s, "G") < 5)
83             {
84                 List<string> tmp = new List<string>();
85                 tmp = copy(s, tmp);
86                 tmp.Add("G");
87                 sitting(tmp);
88             }
89             if (findAndCount(s, "B") < 5)
90             {
91                 List<string> tmp = new List<string>();
92                 tmp = copy(s, tmp);
93                 tmp.Add("B");
94                 sitting(tmp);
95             }
96         }
97     }

```

private void sitting(List<string> s) จะเป็น method หลักในการทำงานเพื่อหาจำนวนวิธีจัดที่นั่งโจทยข้อที่ 1

การทำงานจะเป็นรูปแบบ recursive โดยจะรับ parameter เป็น List<string> s จะเวียนเกิดทุกรูปแบบที่เป็นไปได้ โดย หากจำนวนผู้ที่มานั่งยังไม่ครบ 10 คน และผู้ชายและผู้หญิงที่มานั่งไม่ครบตามจำนวน ก็จะทยอยใส่ผู้ที่มานั่งใหม่ (จาก ชาย 5 คน หญิง 5 คน) โดยได้แยกการทำงานของโปรแกรมเป็นกรณีย่อย ๆ เพิ่มเติมดังนี้

หากผู้ที่มานั่งคนแรกเป็นผู้หญิง คนถัดไปจะต้องเป็นผู้หญิงด้วย

หากคนรองสุดท้าย (คนที่ 9) เป็นผู้หญิง แต่คนที่ 8 ไม่ใช่ผู้หญิง คนสุดท้ายแล้วจะต้องเป็นผู้หญิง

เมื่อครบ 10 คน ตามเงื่อนไขแล้ว จะทำการ Add ไปยัง List<List<string>> test1Ans

และจบการทำงานของ method นั้นๆ

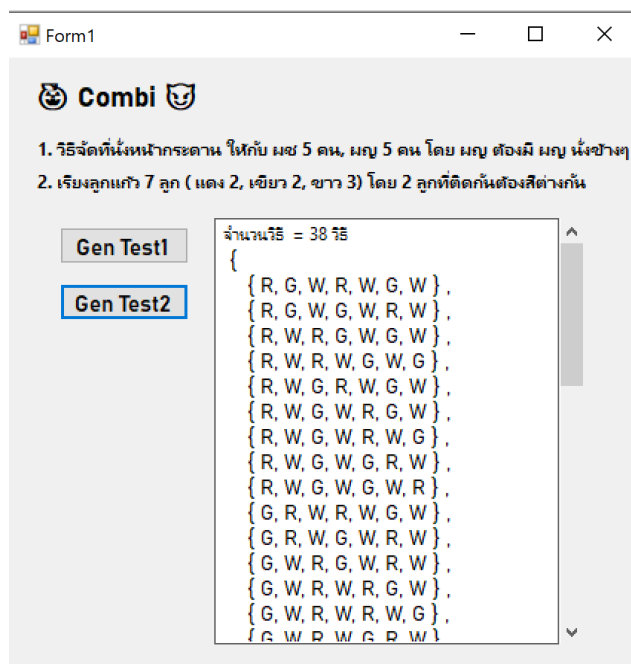
โจทย์ข้อที่ 2:

เรียงลูกแก้ว 7 ลูก (ประกอบไปด้วย สีแดง 2 ลูก สีเขียว 2 ลูก และสีขาว 3 ลูก) โดยทุกๆ 2 ลูกที่อยู่ติดกันต้องมีสีแตกต่างกัน



จำนวนวิธีทั้งหมด 38 วิธี

รูปแบบทั้งหมดที่เป็นไปได้:



`private void lookkaew(List<string> s)` จะเป็น method หลักในการทำงานเพื่อหาจำนวนวิธีจัดที่นั่งโจทย์ข้อที่ 2 ซึ่งจะทำงานในลักษณะเดียวกับ `private void sitting(List<string> s)`

การทำงานจะเป็นรูปแบบ recursive โดยจะรับ parameter เป็น `List<string> s` จะเวียนเกิดทุกๆรูปแบบที่เป็นไปได้ โดย หากจำนวนผลลัพธ์ใน `List<string>` ยังไม่ครบ 7 ลูก และจำนวนลูกแก้วสีต่างๆที่ใส่ก็ยังไม่ครบตามจำนวน ก็จะทยอยใส่ลูกแก้วสีต่างๆ (จาก สีแดง 2 ลูก สีเขียว 2 ลูก และสีขาว 3 ลูก) โดยได้แยกการทำงานของโปรแกรมเป็นกรณีย่อย ๆ เพิ่มเติมดังนี้

หากลูกแก้วลูกสุดท้ายของ `List<string>` มีสีหนึ่ง ก็จะใส่ลูกแก้วสีที่ไม่ซ้ำเข้าไป โดยทำงานเวียนเกิดไปจนครบ

เมื่อครบ 7 ลูก ตามเงื่อนไขแล้ว จะทำการ Add ไปยัง `List<List<string>> test2Ans`

และจบการทำงานของ method นั้นๆ

(สามารถดูการทำงานของโปรแกรม `private void lookkaew(List<string> s)` ได้ในหน้าถัดไป)

```

20     private void button1_Click(object sender, EventArgs e)
21     {
22         textBox1.Text = "";
23         test1Ans.Clear();
24         List<string> es = new List<string>();
25         sitting(es);
26         textBox1.Text = show_result(test1Ans);
27     }
28
29     private void button2_Click(object sender, EventArgs e)
30     {
31         textBox1.Text = "";
32         test2Ans.Clear();
33         List<string> es = new List<string>();
34         lookkaew(es);
35         textBox1.Text = show_result(test2Ans);
36     }

```

- private void button1_Click(object sender, EventArgs e)
 - private void button2_Click(object sender, EventArgs e)
- จะทำการเรียก main method และแสดงผลลัพธ์ที่คำนวณได้จากข้อที่ 1 หรือ ข้อที่ 2

```

99     private void lookkaew(List<string> s)
100     {
101         if (s.Count() == 7)
102         {
103             test2Ans.Add(s);
104             return;
105         }
106
107         // R R , G G , W W W
108
109         if (s.Count() > 0 && s.ElementAt(s.Count() - 1) == "R")
110         {
111             if (findAndCount(s, "G") < 2)
112             {
113                 List<string> tmp = new List<string>();
114                 tmp = copy(s, tmp);
115                 tmp.Add("G");
116                 lookkaew(tmp);
117             }
118             if (findAndCount(s, "W") < 3)
119             {
120                 List<string> tmp = new List<string>();
121                 tmp = copy(s, tmp);
122                 tmp.Add("W");
123                 lookkaew(tmp);
124             }
125         }
126         else if (s.Count() > 0 && s.ElementAt(s.Count() - 1) == "G")
127         {
128             if (findAndCount(s, "R") < 2)
129             {
130                 List<string> tmp = new List<string>();
131                 tmp = copy(s, tmp);
132                 tmp.Add("R");
133                 lookkaew(tmp);
134             }
135             if (findAndCount(s, "W") < 3)
136             {
137                 List<string> tmp = new List<string>();
138                 tmp = copy(s, tmp);
139                 tmp.Add("W");
140                 lookkaew(tmp);
141             }
142         }
143
144         else if (s.Count() > 0 && s.ElementAt(s.Count() - 1) == "W")
145         {
146             if (findAndCount(s, "R") < 2)
147             {
148                 List<string> tmp = new List<string>();
149                 tmp = copy(s, tmp);
150                 tmp.Add("R");
151                 lookkaew(tmp);
152             }
153             if (findAndCount(s, "G") < 2)
154             {
155                 List<string> tmp = new List<string>();
156                 tmp = copy(s, tmp);
157                 tmp.Add("G");
158                 lookkaew(tmp);
159             }
160         }
161
162         else
163         {
164             if (findAndCount(s, "R") < 2)
165             {
166                 List<string> tmp = new List<string>();
167                 tmp = copy(s, tmp);
168                 tmp.Add("R");
169                 lookkaew(tmp);
170             }
171             if (findAndCount(s, "G") < 2)
172             {
173                 List<string> tmp = new List<string>();
174                 tmp = copy(s, tmp);
175                 tmp.Add("G");
176                 lookkaew(tmp);
177             }
178             if (findAndCount(s, "W") < 3)
179             {
180                 List<string> tmp = new List<string>();
181                 tmp = copy(s, tmp);
182                 tmp.Add("W");
183                 lookkaew(tmp);
184             }
185         }
186     }

```