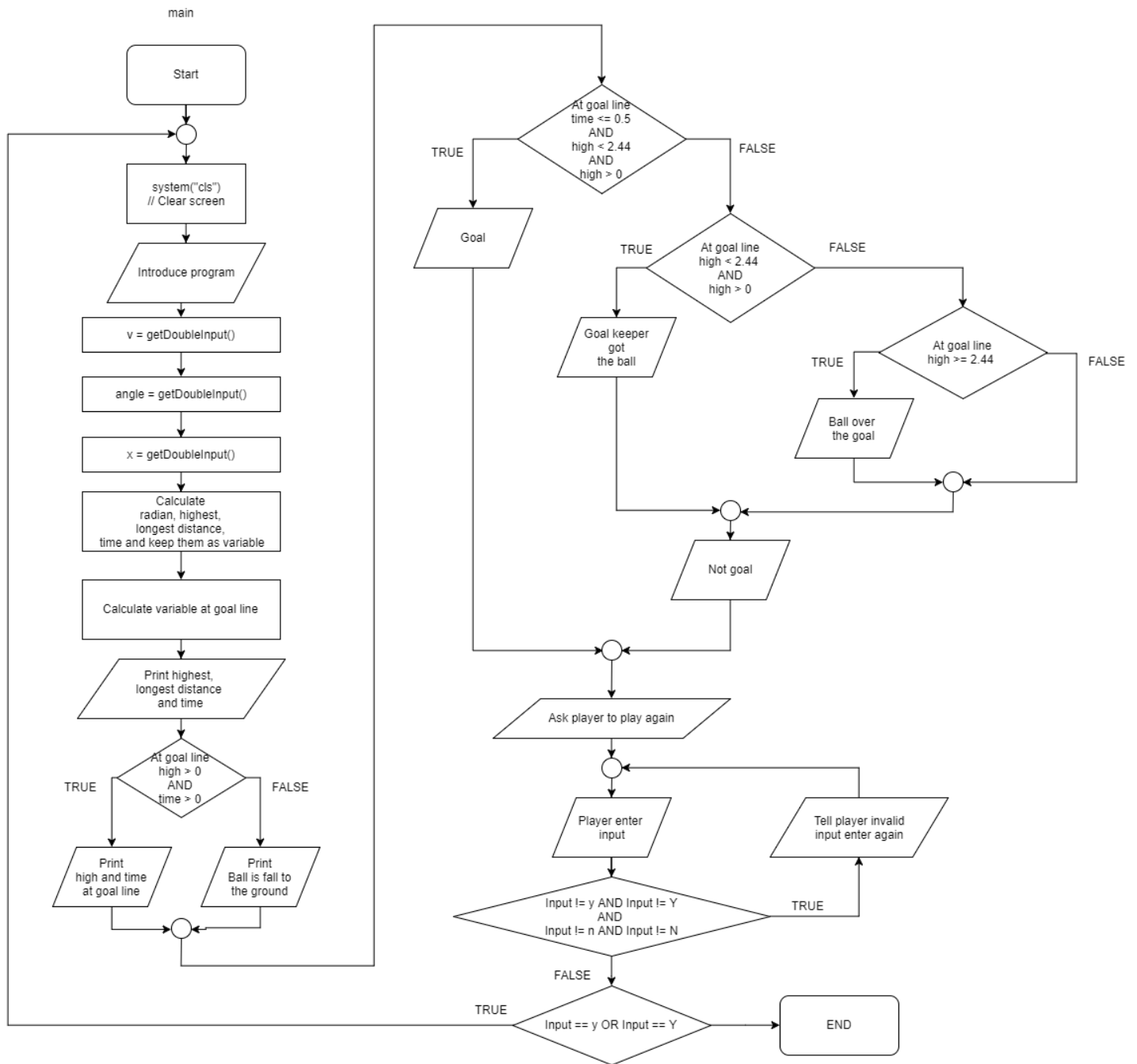
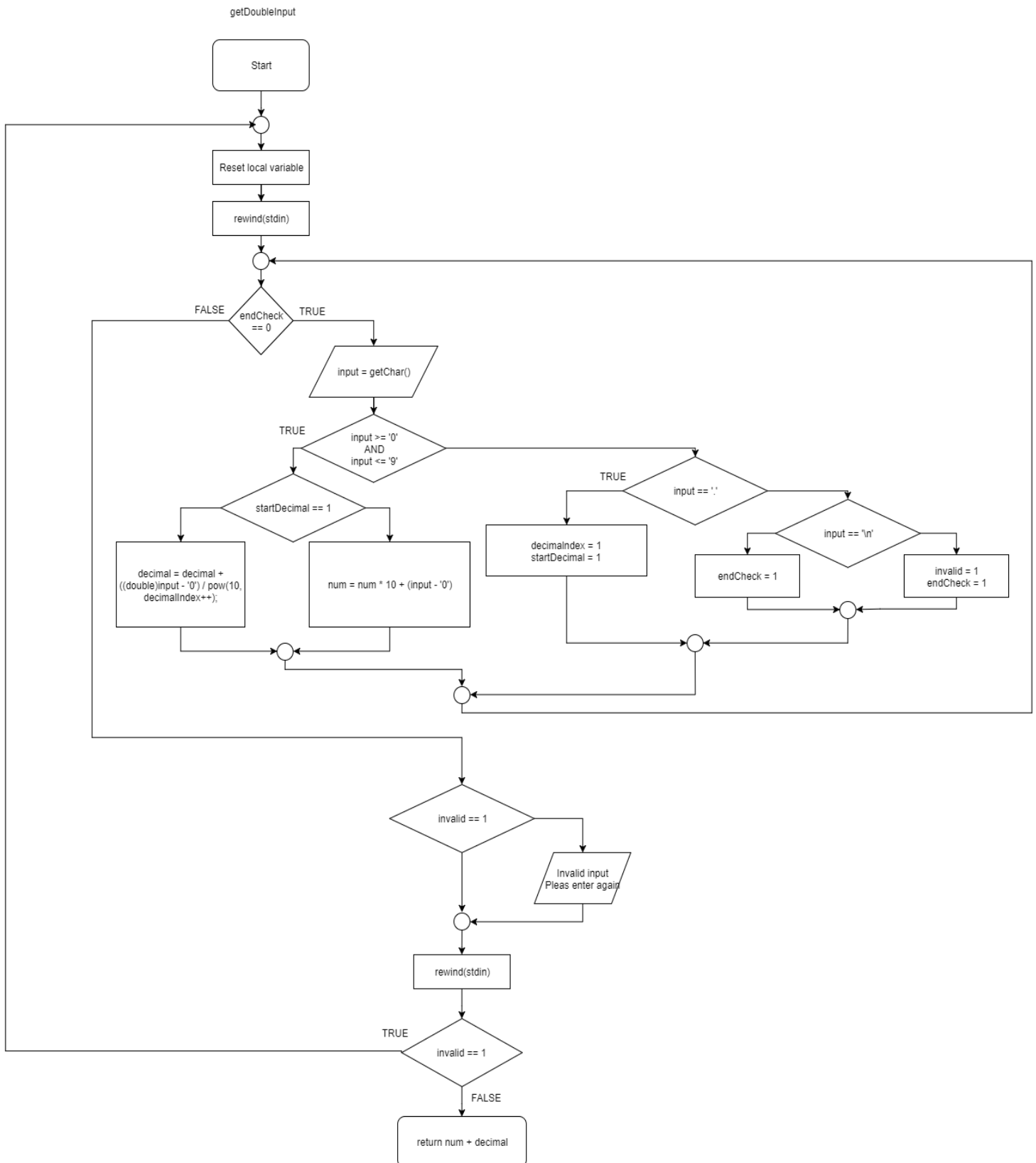


Assignment 4: Project tile

Flow chart: Main function



Flowchart: getDoubleInput function



อธิบายการทำงานของโค้ด

```
1  #include <stdio.h>
2  #include <math.h>
3  #include <stdlib.h>
```

บรรทัดที่ 1 – 3: include library ที่จำเป็นต้องใช้

```
5  double getDoubleInput ()
6  {
7      int endCheck, invalid, startDecimal, decimalIndex;
8      char input;
9      double output, num, decimal;
10     do
11     {
12         endCheck = 0;
13         invalid = 0;
14         startDecimal = 0;
15         decimalIndex = 0;
16         output = 0;
17         num = 0;
18         decimal = 0.0;
19         rewind(stdin);
```

บรรทัดที่ 5: ประกาศฟังก์ชัน getDoubleInput สำหรับเช็คความถูกต้องของ input

บรรทัดที่ 7 – 9: ประกาศตัวแปรเตรียมไว้สำหรับเช็ค input

บรรทัดที่ 10: เริ่มลูป do while

บรรทัดที่ 12 - 18: รีเซ็ตค่าตัวแปรสำหรับลูป

บรรทัดที่ 19: ลบค่าที่ค้างอยู่ใน buffer

```
20     while (endCheck == 0)
21     {
22         input = getchar();
23         if (input >= '0' && input <= '9')
24         {
25             if (startDecimal == 1)
26             {
27                 decimal = decimal + ((double)input - '0') / pow(10, decimalIndex);
28                 decimalIndex++;
29             }
30             else num = num * 10 + (input - '0'); // Regular number
31         }
32         else if (input == '.')
33         {
34             decimalIndex = 1;
35             startDecimal = 1;
36         }
37         else if (input == '\n')
38         {
39             endCheck = 1;
40         } else {
41             invalid = 1;
42             endCheck = 1;
43         }
44     }
```

บรรทัดที่ 20: เริ่มลูป do while ทำงานเมื่อ endCheck == 0 หรือก็คือยังเช็คค่า input ไม่เสร็จ

บรรทัดที่ 22: รับค่า input

บรรทัดที่ 23: เช็ครหัส ASCII ของ input ว่าอยู่ระหว่าง 0 – 9 หรือไม่

บรรทัดที่ 25 – 30: เช็คว่าเริ่มต้นเก็บค่าในตำแหน่งทศนิยมหรือยัง

ถ้าเริ่มแล้ว ให้แปลงค่า input แล้วเก็บเป็นทศนิยมตามตำแหน่งไว้ที่ decimal

ถ้ายังไม่ได้เริ่ม ให้แปลงค่า input แล้วเพิ่มหน่วย num และเพิ่ม input ไปที่ num

บรรทัดที่ 32 - 36: ถ้า input เท่ากับ . ให้เริ่มเก็บค่าในตำแหน่งทศนิยม

บรรทัดที่ 37 - 39: ถ้าว่า input เท่ากับขึ้นบรรทัดใหม่ ให้ endCheck = 1 เพื่อจบการตรวจสอบ Input

บรรทัดที่ 40 – 43: เมื่อไม่เข้าเงื่อนไขใดๆ เลย แสดงว่าได้ input มาไม่ถูกต้อง กำหนด invalid = 1, endCheck = 1

เพื่อให้งบการตรวจสอบค่า input และจำไว้ว่าค่า input ไม่ถูกต้อง

```
46         if (invalid == 1)
47             printf("\nInvalid input ! Please enter again: ");
48
49             rewind(stdin);
50         } while (invalid == 1);
51         return num + decimal;
```

บรรทัดที่ 46: ถ้าค่า input ไม่ถูกต้อง ให้ออกผู้เล่นว่าค่า input ไม่ถูกต้องให้กรอกใหม่

บรรทัดที่ 49: ลบค่าที่ค้างใน buffer ออก

บรรทัดที่ 50: ถ้าค่า input ไม่ถูกต้องทำงานลูปใหม่

บรรทัดที่ 51: return ค่าตัวเลขที่ได้จากการตรวจสอบ input กลับไปแบบ double

```

55     int main ()
56     {
57         double angle, u, x, t;
58         double ux, uy, ty, airTime;
59         double gx, gy, gt;
60         double r, h, sy, sx;
61         double g = 9.81;
62         char playAgain;

```

บรรทัดที่ 55: เริ่มฟังก์ชัน main

บรรทัดที่ 57 – 62: ประกาศตัวแปร

```

64     do
65     {
66         // Clear screen
67         system("cls");
68
69         // Introduce
70         printf("*****\n");
71         printf("\n*");
72         printf("\n*   Freekick calculator   *");
73         printf("\n*   v.1 by Nitipoom Unxrom *");
74         printf("\n*");
75         printf("\n*****\n\n");

```

บรรทัดที่ 64: เริ่มลูป do while สำหรับเริ่มเล่นรอบใหม่

บรรทัดที่ 67: clear screen ล้างหน้าจอจากลูปครั้งที่แล้วออก เพื่อความสวยงามไม่รกหน้าจอ

บรรทัดที่ 70 – 75: โปรแกรมแนะนำตนเอง เนื่องจากการล้างหน้าจอออกใหม่ในรอบที่เริ่มเล่น จึงต้องแนะนำโปรแกรมใหม่ทุกครั้งเพื่อความสวยงาม

```

77         // Enter input
78         printf("Velocity (m/s): ");
79         u = getDoubleInput();
80         printf("Angle (degree): ");
81         angle = getDoubleInput();
82         printf("Distance (m): ");
83         x = getDoubleInput();

```

บรรทัดที่ 78 – 79: ถามความเร็ว และให้ฟังก์ชัน getDoubleInput รับค่า Input จากผู้เล่นไปตรวจสอบและส่งค่ากลับมาเป็น double เก็บไว้ที่ตัวแปร u

บรรทัดที่ 78 – 79: ถามมุม และให้ฟังก์ชัน getDoubleInput รับค่า Input จากผู้เล่นไปตรวจสอบและส่งค่ากลับมาเป็น double เก็บไว้ที่ตัวแปร angle

บรรทัดที่ 78 – 79: ถามระยะทาง และให้ฟังก์ชัน getDoubleInput รับค่า Input จากผู้เล่นไปตรวจสอบและส่งค่ากลับมาเป็น double เก็บไว้ที่ตัวแปร x

```

85 // Calculation
86 r = angle * M_PI / 180.0;
87 uy = u * sin(r);
88 ux = u * cos(r);
89 ty = uy / g;
90 airTime = 2 * (ty);
91 sy = pow(uy, 2) / (2 * g);
92 sx = ux * airTime;
93 gt = x/ux; // Time at goal line
94 gy = uy * gt + 0.5 * (-g) * pow(gt, 2); // Height at goal line

```

บรรทัดที่ 86: คำนวณแปลงมุมองศาเป็นเรเดียน

บรรทัดที่ 87 – 88: คำนวณแรงในแนวนอนและแนวระดับ

บรรทัดที่ 89: คำนวณหาเวลาที่ตำแหน่งสูงสุด

บรรทัดที่ 90: คำนวณหาเวลาที่ลอยอยู่ในอากาศ

บรรทัดที่ 91: คำนวณหาค่าความสูงจากพื้นของลูกบอลที่ลอยขึ้นไปได้สูงที่สุด

บรรทัดที่ 92: คำนวณหาระยะที่ไกลที่สุดที่ลูกบอลไปได้

บรรทัดที่ 93: คำนวณหาเวลาที่ลูกบอลลอยไปถึงตำแหน่งเส้นประตู

บรรทัดที่ 94: คำนวณหาค่าความสูงจากพื้นของลูกบอลที่ลอยอยู่ที่ตำแหน่งเส้นประตู

```

96 // Print result
97 printf("\n===== RESULT =====");
98 printf("\n   Highest: %gm", sy);
99 printf("\n   Longest distance: %gm", sx);
100 printf("\n   Air Time: %gs", airTime);

```

บรรทัดที่ 97 – 100: บอกค่าที่ได้จากการคำนวณกับผู้เล่น ว่าลูกลอยได้สูงสุดเท่าไร ไปได้ไกลเท่าใด และเวลาที่ลอยอยู่ในอากาศนานเท่าไร

```

102 if (gy > 0 && gt > 0)
103     printf("\n   Height at goal line %gm in %gs", gy, gt);
104 else printf("\n   * Ball fall to the ground before reach goal");

```

บรรทัดที่ 102 – 104: ตรวจสอบว่าบอลลอยเหนือพื้นที่ตำแหน่งเส้นประตูหรือไม่

ถ้าใช่ ให้แสดงค่าความสูงที่ตำแหน่งเส้นประตูและเวลาที่ลูกบอลไปถึง

ถ้าไม่ใช่ ให้บอกผู้เล่นว่าบอลตกถึงพื้นก่อนถึงเส้นประตู

```

106         if (gt <= 0.5 && gy < 2.44 && gy > 0)
107         {
108             printf("\n >>>>> GOAL ! ");
109         } else {
110             if (gy < 2.44 && gy > 0)
111                 printf("\n * Goal keeper get the ball");
112             else if (gy >= 2.44)
113                 printf("\n * Ball over the goal");
114             printf("\n >>>>> NOT GOAL");
115         }

```

บรรทัดที่ 106 - 109: ตรวจสอบว่าตำแหน่งเส้นประตูบอลใช้เวลาเดินทางมาถึงน้อยกว่าหรือเท่ากับ 0.5 วินาที รวมทั้งยังไม่ตกถึงพื้น และลอยต่ำกว่า 2.44 เมตร

ถ้าใช่ แสดงว่าบอลเข้าประตู ให้บอกผู้เล่นว่าลูกบอลเข้าประตู

ถ้าไม่ใช่ แสดงว่าบอลไม่เข้าประตูให้ทำคำสั่งในบรรทัดที่ 110 - 114 เพื่อหาสาเหตุว่าทำไมบอลถึงไม่เข้าประตู

บรรทัดที่ 110 - 111: ถ้าบอลลอยอยู่ต่ำกว่า 2.44 เมตร และลอยเหนือพื้น แสดงว่าเวลาที่บอลเดินทางมาถึงโกลด์เกิน 0.5 บอกผู้เล่นว่า ผู้รักษาประตูสามารถรักษาประตูเอาไว้ได้

บรรทัดที่ 112 - 113: ถ้าบอลสูงกว่าหรือเท่ากับ 2.44 เมตร ให้บอกผู้เล่นว่า บอลลอยข้ามประตู

บรรทัดที่ 114: บอกผู้เล่นว่าบอลไม่เข้าประตู

```

116         rewind(stdin);
117         printf("\n\nDo you want to play again ? [Y/n]: ");
118         scanf("%c", &playAgain);
119         while(playAgain != 'y' && playAgain != 'Y' && playAgain != 'n' && playAgain != 'N')
120         {
121             printf("\nInvalid input ! Please enter Y or N");
122             scanf("%c", &playAgain);
123             rewind(stdin);
124         }
125
126     } while(playAgain == 'y' || playAgain == 'Y');
127
128     return 0;
129 }

```

บรรทัดที่ 116: ลบค่า input ที่ค้างอยู่ใน buffer เพื่อรอรับค่า input ใหม่จากผู้เล่น

บรรทัดที่ 117: ถามผู้เล่นว่าต้องการเล่นใหม่หรือไม่

บรรทัดที่ 118: รับ input จากผู้เล่น

บรรทัดที่ 119: เริ่มลูป while ทำงานเมื่อค่า input ที่ได้รับมาไม่ใช่ y, Y, n, N

บรรทัดที่ 121: บอกผู้เล่นว่าใส่ค่า input ไม่ถูกต้องให้ใส่มาใหม่

บรรทัดที่ 122: รอรับค่าที่ผู้เล่นจะใส่ input มาใหม่

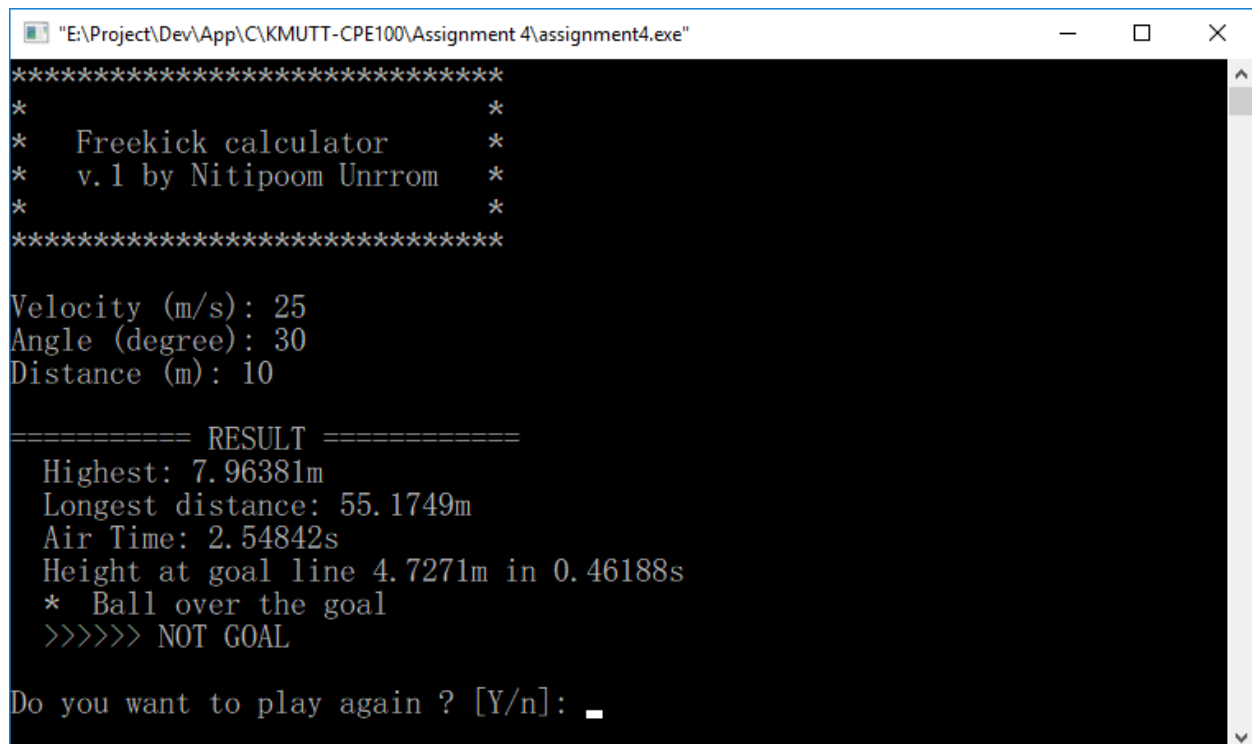
บรรทัดที่ 123: ลบค่า input ที่ค้างอยู่ใน buffer

บรรทัดที่ 126: ถ้าค่า input เท่ากับ y หรือ Y ให้วนกลับไปทำงานใน do เพื่อเล่นใหม่อีกครั้ง

ผลการทดสอบการทำงานของโค้ด

1. ทดสอบที่ ความเร็ว 25 มุม 30 ระยะทาง 10

ผลลัพธ์ บอลลอยข้ามประตู บอลไม่เข้าประตู



```
"E:\Project\Dev\App\C\KMUTT-CPE100\Assignment 4\assignment4.exe"
*****
*                               *
*   Freekick calculator         *
*   v.1 by Nitipoom Unrrom     *
*                               *
*****

Velocity (m/s): 25
Angle (degree): 30
Distance (m): 10

===== RESULT =====
Highest: 7.96381m
Longest distance: 55.1749m
Air Time: 2.54842s
Height at goal line 4.7271m in 0.46188s
* Ball over the goal
>>>>> NOT GOAL

Do you want to play again ? [Y/n]: _
```


2. ทดสอบที่ ความเร็ว 25 มุม 10 ระยะทาง 10

ผลลัพธ์ บอลเข้าประตู

```
"E:\Project\Dev\App\C\KMUTT-CPE100\Assignment 4\assignment4.exe"
*****
*                               *
*   Freekick calculator         *
*   v.1 by Nitipoom Unrrom     *
*                               *
*****

Velocity (m/s): 25
Angle (degree): 10
Distance (m): 10

===== RESULT =====
Highest: 0.960553m
Longest distance: 21.7903m
Air Time: 0.885057s
Height at goal line 0.954069m in 0.406171s
>>>>> GOAL !

Do you want to play again ? [Y/n]: _
```

3. ทดสอบที่ ความเร็ว 20 มุม 20 ระยะทาง 20

ผลลัพธ์ ผู้รักษาประตูรักษาประตูเอาไว้ได้ บอลไม่เข้าประตู

```
"E:\Project\Dev\App\C\KMUTT-CPE100\Assignment 4\assignment4.exe"
*****
*                               *
*   Freekick calculator         *
*   v.1 by Nitipoom Unrrom     *
*                               *
*****

Velocity (m/s): 20
Angle (degree): 20
Distance (m): 20

===== RESULT =====
Highest: 2.38487m
Longest distance: 26.2095m
Air Time: 1.39458s
Height at goal line 1.72462m in 1.06418s
* Goal keeper get the ball
>>>>> NOT GOAL

Do you want to play again ? [Y/n]:
```

4. ทดสอบที่ ความเร็ว 10 มุม 10 ระยะทาง 10

ผลลัพธ์ บอลตกถึงพื้นก่อนถึงเส้นประตู บอลไม่เข้าประตู

```
"E:\Project\Dev\App\C\KMUTT-CPE100\Assignment 4\assignment4.exe"

*****
*                                     *
*   Freekick calculator               *
*   v.1 by Nitipoom Unrrom           *
*                                     *
*****

Velocity (m/s): 10
Angle (degree): 10
Distance (m): 10

===== RESULT =====
Highest: 0.153689m
Longest distance: 3.48644m
Air Time: 0.354023s
* Ball fall to the ground before reach goal
>>>>> NOT GOAL

Do you want to play again ? [Y/n]: _
```

5. ทดสอบการป้องกันการหยุดทำงาน เมื่อป้อนค่าผิด

```
"E:\Project\Dev\App\C\KMUTT-CPE100\Assignment 4\assignment4.exe"

*****
*                                     *
*   Freekick calculator               *
*   v.1 by Nitipoom Unrrom           *
*                                     *
*****

Velocity (m/s): 2x

Invalid input ! Please enter again: _
```

6. คำถามเพื่อถามผู้เล่นว่าต้องการเล่นใหม่หรือไม่

```

E:\Project\Dev\App\C\KMUTT-CPE100\Assignment 4\assignment4.exe
*****
*                                     *
*   Freekick calculator               *
*   v.1 by Nitipoom Unrrom          *
*                                     *
*****

Velocity (m/s): 10
Angle (degree): 10
Distance (m): 10

===== RESULT =====
Highest: 0.153689m
Longest distance: 3.48644m
Air Time: 0.354023s
* Ball fall to the ground before reach goal
>>>>> NOT GOAL

Do you want to play again ? [Y/n]: _

```

หน้าจอเมื่อเริ่มใหม่

[illegible]