รายวิชาคอมพิวเตอร์

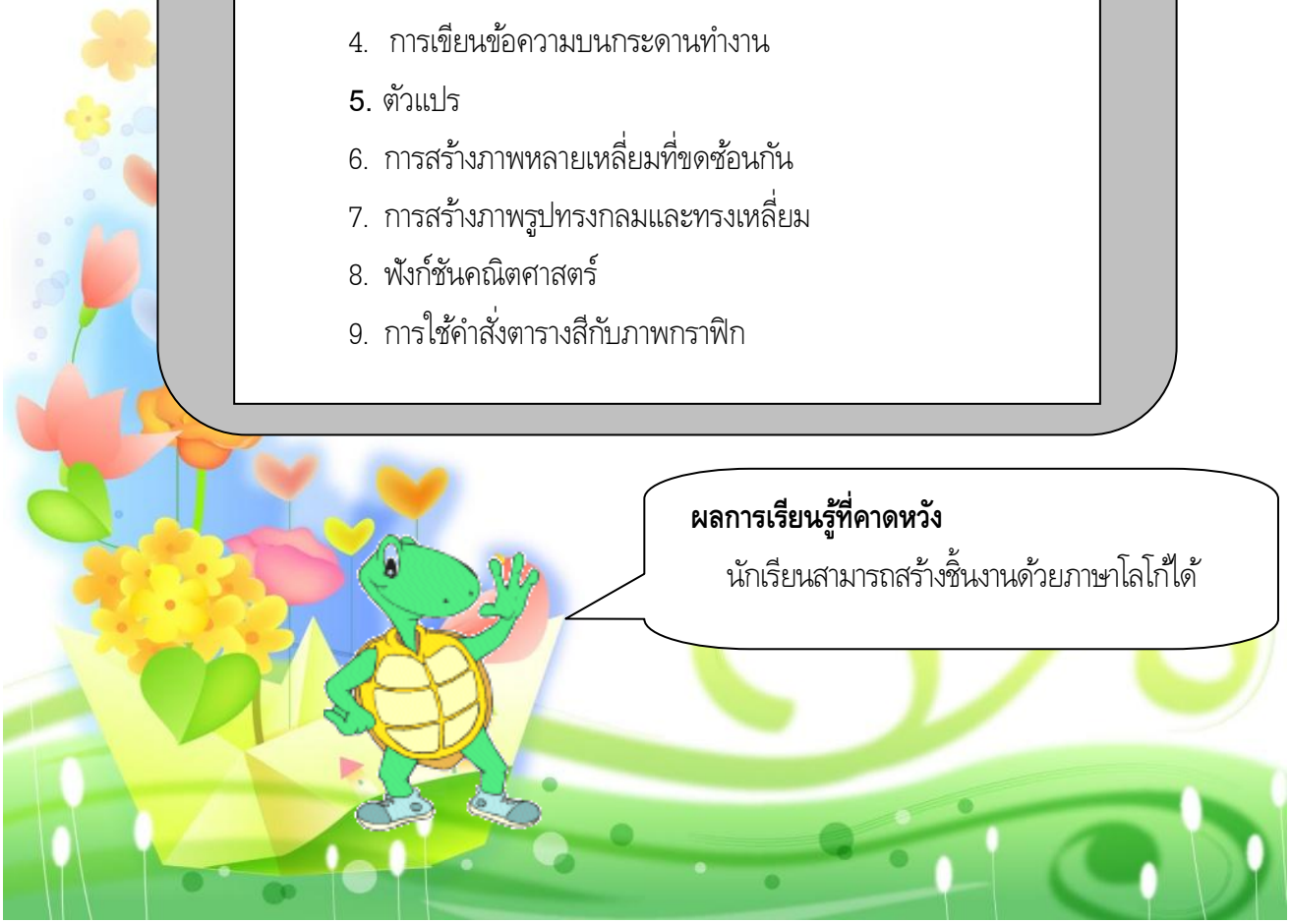
เอกสารประกอบการเรียน

การพัฒนาโปรแกรม LOGO

หน่วยการเรียนรู้ที่ 6 การสร้างชิ้นงานด้วยภาษาโลโก้

สาระการเรียนรู้

1. การวาดภาพวงกลมและส่วนโค้งอย่างง่าย
2. การสร้างภาพแบบสุ่มข้อมูล
3. การสร้างภาพหลายเหลี่ยม
4. การเขียนข้อความบนกระดานทำงาน
5. ตัวแปร
6. การสร้างภาพหลายเหลี่ยมที่ขดซ้อนกัน
7. การสร้างภาพรูปทรงกลมและทรงเหลี่ยม
8. ฟังก์ชันคณิตศาสตร์
9. การใช้คำสั่งตารางสีกับภาพกราฟิก



ผลการเรียนรู้ที่คาดหวัง

นักเรียนสามารถสร้างชิ้นงานด้วยภาษาโลโก้ได้

ใบความรู้ที่ 5 เรื่อง การสร้างชิ้นงานด้วยภาษาโลโก้

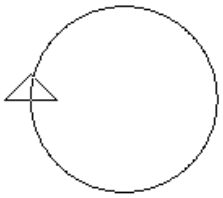
1. การวาดภาพวงกลมและส่วนโค้ง

1) การใช้คำสั่ง Repeat

ในโปรแกรม MSWLogo มีคำสั่งใช้วาดวงกลมอย่างง่าย คือ การใช้คำสั่งดังนี้

```
repeat 360 [ fd  1  rt  1 ]
```

ผลลัพธ์ที่ได้คือ



รูปที่ 5.1 แสดงผลจากการใช้คำสั่ง repeat สร้างรูปวงกลม

แล้วถ้าต้องการสร้างรูปส่วนโค้ง ลักษณะครึ่งวงกลม จะใช้คำสั่ง repeat ในการสร้างรูปได้ยังไง ?

เราสามารถสร้างรูปครึ่งวงกลมด้วยคำสั่ง repeat ได้ดังนี้

เช่น

```
repeat 180 [ fd  1  rt  1 ]
```

จะได้ผลลัพธ์ดังนี้ คือ

รูปที่ 5.2 แสดงผลจากการใช้คำสั่ง repeat สร้างรูปครึ่งวงกลม

ข้อสังเกต ของการสร้างรูปภาพวงกลม และส่วนโค้งวงกลม ด้วยคำสั่ง repeat ก็คือ ไม่สามารถกำหนดขนาดของวงกลมตามความต้องการได้

2) คำสั่ง arc

นอกจากนั้นสามารถใช้คำสั่ง arc ในการสร้างวงกลมและส่วนโค้งได้ โดยมีรูปแบบการใช้คำสั่งดังนี้

รูปแบบคำสั่ง

arc องศา รัศมี

ตัวอย่างเช่น

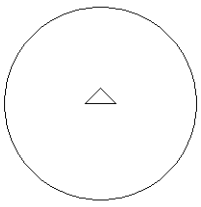
arc 360 100

ความหมายของคำสั่ง คือ การวาดวงกลมที่มีขนาด รัศมีเท่ากับ 100 หน่วย

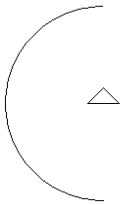
arc 180 100

ความหมายของคำสั่ง คือ การวาดส่วนโค้งของวงกลมที่มีขนาด 180 องศา และมีรัศมีเท่ากับ 100 หน่วย

จะได้ผลลัพธ์ดังนี้ คือ



arc 360 100



arc 180 100

รูปที่ 5.3 แสดงผลจากการใช้คำสั่ง arc สร้างรูปวงกลมและส่วนโค้งวงกลม

3) คำสั่ง circle

การสร้างรูปกลมที่สามารถกำหนดขนาดของวงกลมด้วยขนาดของรัศมี ในโปรแกรม MSWLogo สามารถเขียนวงกลมตามความยาวของรัศมี สามารถทำได้ 2 รูปแบบ คือ

3.1 การใช้คำสั่ง circle เป็นการสร้างรูปวง โดยที่เต่าโลโก้จะอยู่ตำแหน่งที่จุดศูนย์กลางของรูป

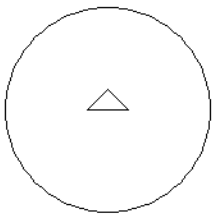
รูปแบบคำสั่ง

```
circle  ความยาวของรัศมี
```

ตัวอย่างการใช้คำสั่ง

```
circle  80
```

ผลลัพธ์ที่ได้



รูปที่ 5.4 แสดงผลจากการใช้คำสั่ง circle สร้างรูปวงกลมและส่วนโค้งวงกลม

3.2 การใช้คำสั่ง circle2 เป็นการสร้างรูปวง โดยที่เต่าโลโก้จะอยู่ในตำแหน่งที่เส้นรอบวง

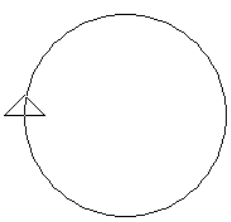
รูปแบบคำสั่ง

```
circle2  ความยาวของรัศมี
```

ตัวอย่างการใช้คำสั่ง

```
circle2  80
```

ผลลัพธ์ที่ได้



รูปที่ 5.5 แสดงผลจากการใช้คำสั่ง circle2 สร้างรูปวงกลมและส่วนโค้งวงกลม

4) คำสั่ง ellipse

การสร้างภาพกราฟิกในรูปแบบของวงรี สามารถใช้คำสั่ง ellipse ในการวาดภาพวงรีได้ โดยมีรูปแบบการใช้คำสั่งดังนี้

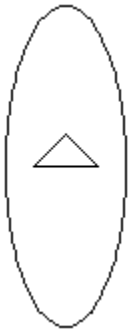
รูปแบบคำสั่ง

ellipse ความกว้าง ความสูง

ตัวอย่างการใช้คำสั่ง

ellipse 30 80

ผลลัพธ์ที่ได้



รูปที่ 5.6 แสดงผลจากการใช้คำสั่ง ellipse สร้างรูปวงกลมและส่วนโค้งวงกลม

2. การสร้างภาพแบบสุ่มข้อมูล

การสร้างภาพโดยไม่มีข้อกำหนดที่ชัดเจน (Random) เป็นวิธีการหนึ่งที่จะทำให้เราได้ภาพที่ปรากฏบนจอแสดงผลไม่สามารถจินตนาการล่วงหน้าว่าผลที่เกิดจะเป็นอย่างไร เนื่องจากเป็นการสุ่มค่าข้อมูลโดยคอมพิวเตอร์ คำสั่งเหมือนกันอาจจะได้ภาพที่แตกต่างกัน

รูปแบบคำสั่ง

random <ค่าตัวเลข>

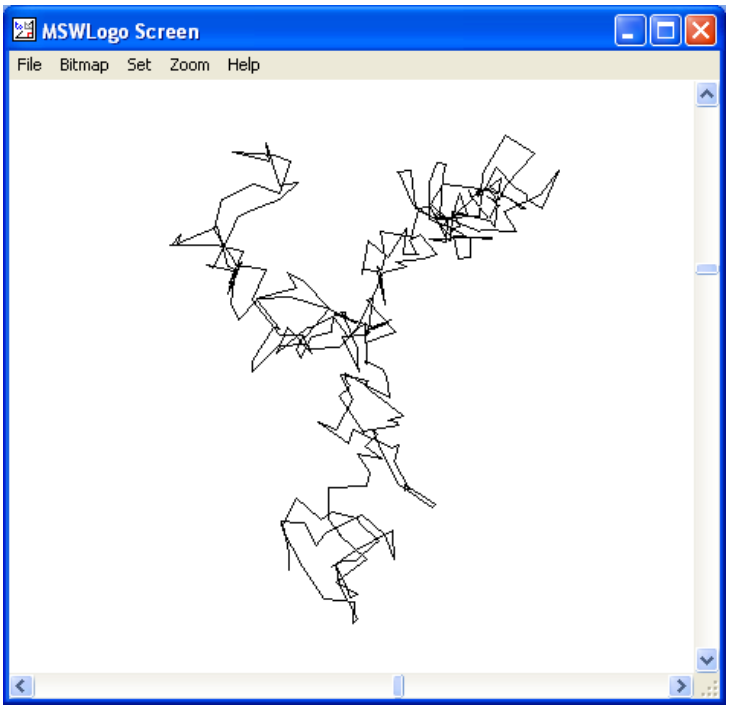
ตัวอย่างการใช้คำสั่ง

random 30 (หมายถึง การเลือกสุ่มค่าตัวเลขตั้งแต่ 0 - 30)

เราสามารถนำคำสั่งไปใช้ร่วมกับคำสั่งอื่นได้ เช่น

repeat 300 [fd random 30 rt random 360]

ภาพที่ปรากฏบนจอหน้าต่างแสดงผลอาจเป็นภาพลายเส้น ภาพแต่ละภาพอาจแตกต่างกันไม่จำเป็นต้องเหมือนกันก็ได้ เนื่องจากคอมพิวเตอร์สุ่มข้อมูลเอง



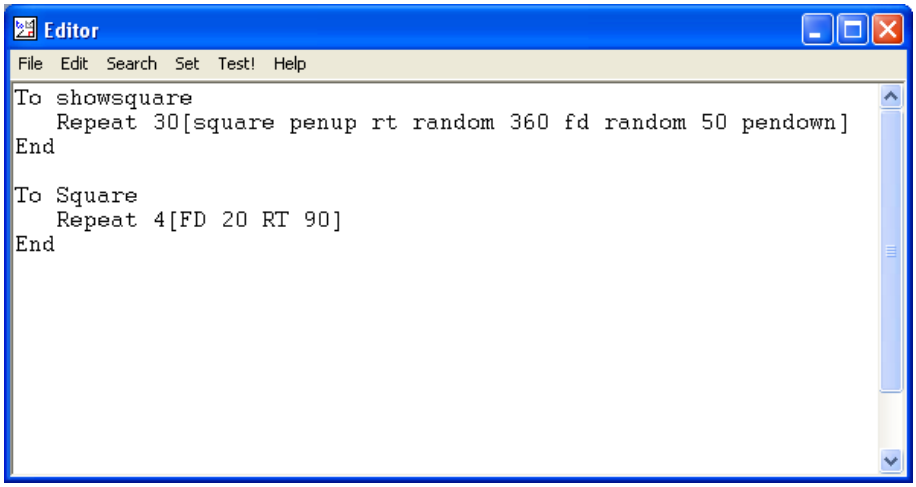
รูปที่ 5.7 แสดงผลลัพธ์การใช้คำสั่งสร้างแบบสุ่มข้อมูล ร่วมกับคำสั่งทำซ้ำ

นอกจากการใช้คำสั่งพื้นฐานแล้วนักเรียนสามารถผสมผสานคำสั่งอื่นเข้ามาร่วมด้วยก็ได้ เช่น คำสั่ง PU (Pen Up) หรือ PD (Pen Down) ในการสุ่มการสร้างภาพบนจอแสดงผลโดยการเขียนกระบวนงานดังนี้

```
to square
   repeat 4 [fd 20 rt 90]
end

to showsquare
   repeat  30 [square penup rt random 360 fd random 50 pendown]
end
```

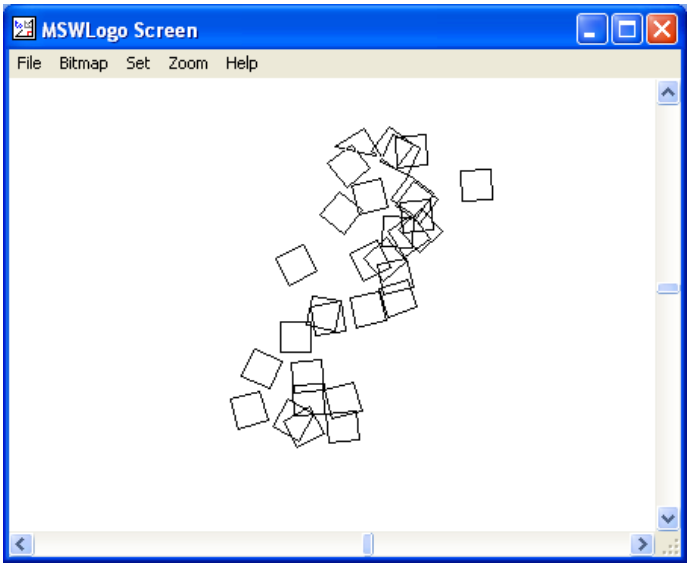
ใช้หน้าต่าง Editor เขียนกระบวนงานได้ตามตัวอย่างดังรูป



รูปที่ 5.8 แสดงผลลัพธ์การใช้คำสั่งสร้างแบบสุ่มข้อมูลรูปสี่เหลี่ยม ร่วมกับคำสั่งทำซ้ำ

จากคำสั่งข้างต้น จะเห็นว่าการเขียนกระบวนการทำงานมี 2 ขั้นตอนกระบวนงานที่ 1 คือ To Square กระบวนงานนี้เป็นการสร้างภาพสี่เหลี่ยมจัตุรัสความยาวแต่ละด้านมีค่าเท่ากับ 20 หน่วย

ส่วนกระบวนงานที่สองคือ To Showsquare เป็นการเรียกกระบวนงานแรกมาสร้างภาพแบบสุ่มค่ามุมและระยะห่างกันของภาพสี่เหลี่ยม มีการยกปากกาขึ้นมีวาดภาพสี่เหลี่ยมเสร็จ และวางปากกาลงเมื่อเริ่มวาดภาพสี่เหลี่ยมภาพใหม่ ภาพสี่เหลี่ยมจัตุรัสจะมีจำนวน 30 รูปบนหน้าจอ ดังแสดงในรูปที่ 5.8 ข้างต้น



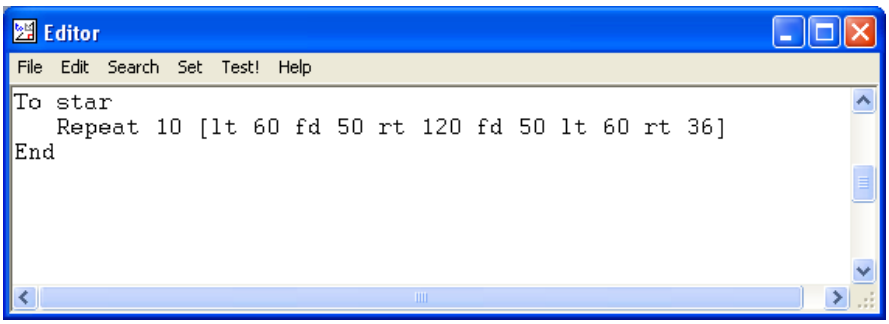
รูปที่ 5.9 แสดงผลลัพธ์การใช้คำสั่งสร้างแบบสุ่ม ร่วมกับการใช้กระบวนงาน

3. การสร้างภาพหลายเหลี่ยม

การสร้างภาพหลายเหลี่ยม เช่น ภาพดาว นักเรียนต้องทำความเข้าใจเกี่ยวกับภาพที่ต้องการสร้างก่อน เนื่องจากบางคนอาจชอบให้มีจำนวนแฉกมาก บางคนต้องการดาวขนาดเล็กหรือใหญ่แตกต่างกัน ภาพดาวมีลักษณะเป็นแฉกแบบใด ดังนั้นต้องมีการออกแบบแล้วจึงมาทำการกำหนดคำสั่งในการสร้าง ตัวอย่างเช่น การสร้างดาวสิบแฉก

การสร้างดาวสิบแฉก มีขั้นตอนการกระบวนงานในสร้างภาพกราฟิกรูปดาวสิบแฉก ดังนี้

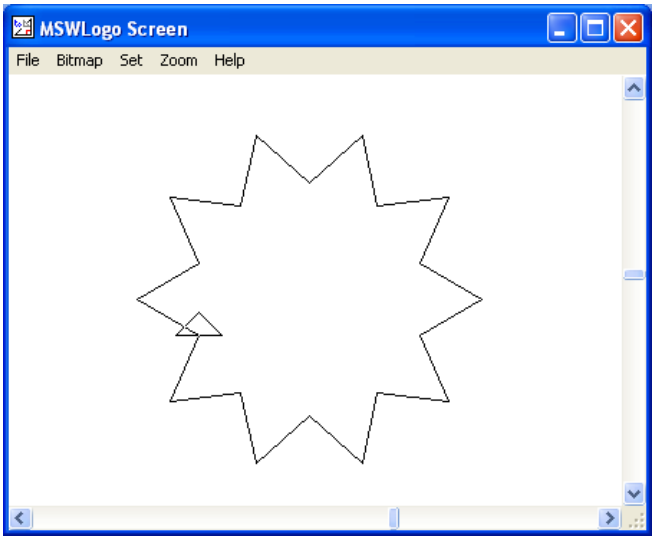
ขั้นตอนที่ 1 เรียกโปรแกรม Editor เพื่อพิมพ์คำสั่งดังนี้



รูปที่ 5.10 แสดงหน้าต่าง Editor การพิมพ์คำสั่งสร้างรูปหลายแฉก

ขั้นตอนที่ 2 ทำการบันทึกที่เมนู File แล้วเลือก save and exit จากหน้าต่าง Editor

ขั้นตอนที่ 3 พิมพ์คำสั่ง star ในช่อง input box แล้วกด Enter จะปรากฏดาว 10 แฉกในหน้าต่างแสดงผล ดังแสดงในรูป



รูปที่ 5.11 ผลลัพธ์ที่ได้จากการพิมพ์สั่งสร้างรูปหลายแฉก

4. การเขียนข้อความบนกระดานทำงาน

การวาดภาพอาจมีความจำเป็นในการแสดงภาพและข้อความบนกระดานทำงานพร้อมกัน การแสดงข้อความบนกระดานทำงานพร้อมกัน การแสดงข้อความบนกระดานทำงานทำได้โดยการใช้คำสั่ง Label ซึ่งมีรูปแบบคำสั่งดังนี้

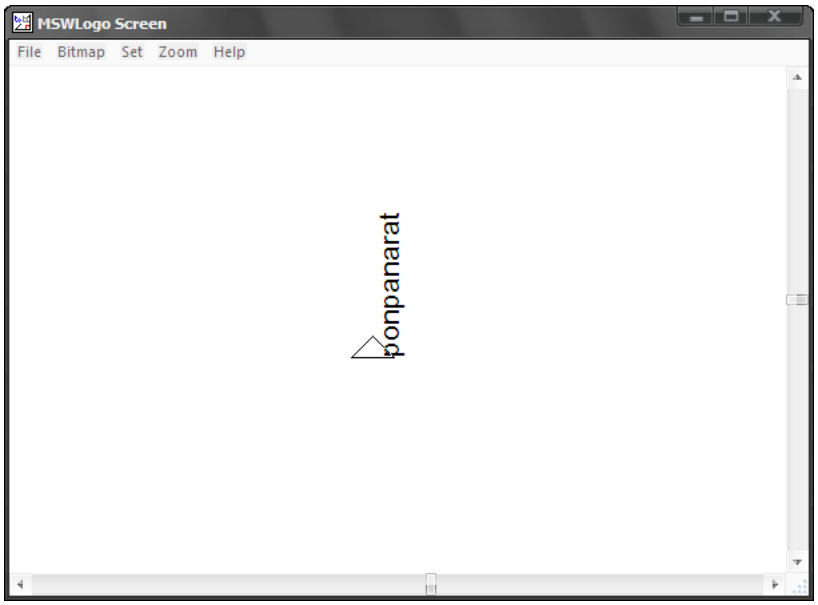
รูปแบบคำสั่ง

Label "ข้อความ

ตัวอย่างการใช้คำสั่ง

Label "ponpanarat

ผลลัพธ์ที่ได้คือ



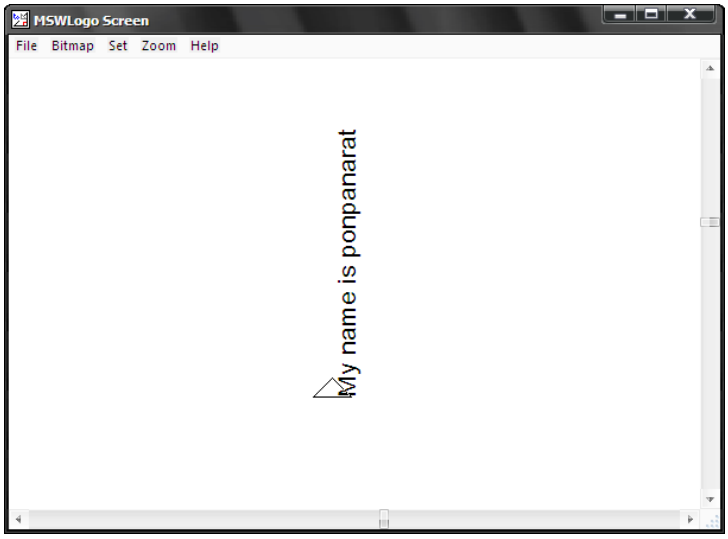
รูปที่ 5.12 ผลลัพธ์ที่ได้จากการใช้คำสั่ง Label

ถ้าต้องการแสดงเป็นประโยคหรือข้อความจำนวนมาก สามารถใส่ข้อความที่ต้องการในวงเล็บ [ ] (square bracket)

ตัวอย่างเช่น

Label [My name is ponpanarat]

ผลลัพธ์ที่ได้คือ



รูปที่ 5.13 ผลลัพธ์ที่ได้จากการใช้คำสั่ง Label แสดงข้อความแบบเป็นประโยค

ข้อสังเกต

ข้อความที่ต้องการแสดงจะต้องอยู่หลังเครื่องหมายอัญประกาศ (") การแสดงข้อความในจอแสดงผลจะสัมพันธ์กับตำแหน่งของเต่าโลโก้ในขณะใช้คำสั่งนั้น ดังนั้นการเปลี่ยนทิศทางของเต่าจะเปลี่ยนทิศทางแสดงข้อความบนจอภาพด้วย

5. ตัวแปร (Variables)

5.1 นิยามตัวแปร (Variable)

ตัวแปร คือ อักษรหรือตัวอักษรปนตัวเลขที่สามารถทดแทนค่าใดค่าหนึ่ง โดยค่าของตัวแปรนั้นเปลี่ยนแปลงได้ การกำหนดค่าตัวแปรในโปรแกรม MSWLogo มีรูปแบบคำสั่งดังนี้

รูปแบบ

make "ชื่อตัวแปร ค่าตัวแปร

ตัวอย่างเช่น

make "size 20

จากตัวอย่าง หมายถึง ตัวแปรชื่อ size และตัวแปรมีค่าเท่ากับ 20

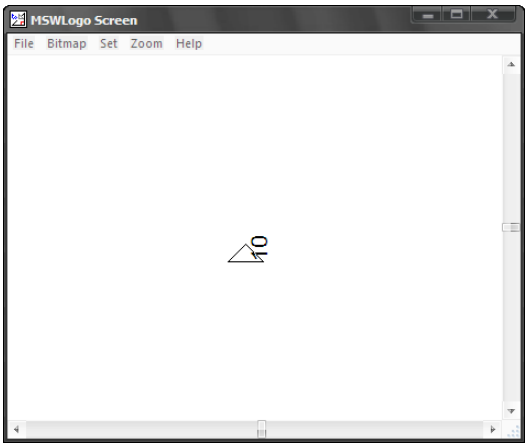
5.2 การใช้ตัวแปร

การใช้ประโยชน์ของตัวแปรในการเขียนกระบวนงานจะต้องมีการอ้างถึงตัวแปร โดยการพิมพ์เครื่องหมาย **:** (Colon) ไว้ด้านหน้าชื่อตัวแปรนั้น จากตัวอย่างข้างต้น สามารถอ้างถึงค่าตัวแปรในรูปของคำสั่งได้ดังนี้คือ **:** size

ตัวอย่างการสร้างตัวแปรที่กำหนดค่าตัวแปรแบบตัวเลข เราสามารถใช้ตัวแปรในการเขียนกระบวนงานได้ดังนี้

```
to number
   make "size 10
   label :size
end
```

ผลลัพธ์ที่ได้คือ



รูปที่ 5.14 แสดงการใช้คำสั่ง Make เพื่อสร้างตัวแปรและการใช้คำสั่ง Label เพื่อแสดงค่าที่เก็บไว้ในตัวแปร size

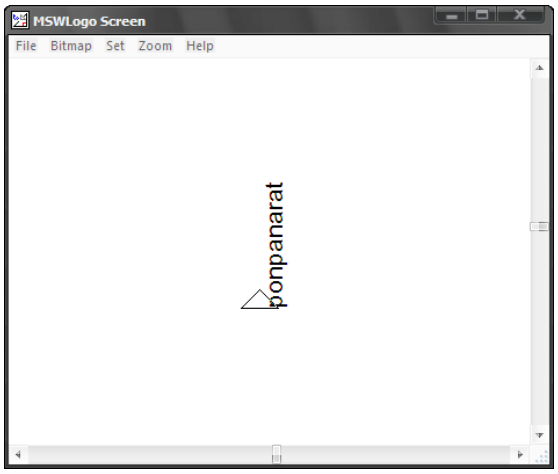
ค่าของตัวแปรอาจเป็นข้อความก็ได้ เมื่อมีการกำหนดค่าให้กับตัวแปรที่เป็นข้อความ ให้ค่าเหล่านั้นจะต้องอยู่ในเครื่องหมาย [ ] (square bracket)

ตัวอย่างการสร้างกระบวนงานที่กำหนดค่าตัวแปรแบบข้อความ

```
to name
   make "name [ponpanarat]
   label :name
end
```

หมายถึง ตัวแปร name มีค่าของตัวแปรเป็นข้อความเท่ากับ ponpanarat

ผลลัพธ์ที่ได้คือ



รูปที่ 5.15 แสดงการใช้คำสั่ง Make เพื่อสร้างตัวแปรและการใช้คำสั่ง Label เพื่อแสดงค่าที่เก็บไว้ในตัวแปร size

การประยุกต์ใช้งานตัวแปรในการสร้างภาพหลายเหลี่ยม จากที่เรียนการสร้างภาพหลายเหลี่ยมมาแล้วนั้น เราสามารถนำวิธีการกำหนดตัวแปรขึ้นมาใช้งานในการสร้างภาพหลายเหลี่ยมได้ ในที่นี้เป็นตัวอย่างการสร้างกระบวนงานที่ชื่อว่า polygon

ในที่นี้จะกำหนดตัวแปรขึ้นมา 2 ตัวแปรคือ

size แทน ค่าจำนวนด้านหรือเหลี่ยม

edge แทน ค่าความยาวของด้าน

ตัวอย่างการสร้างกระบวนงานที่ชื่อว่า polygon

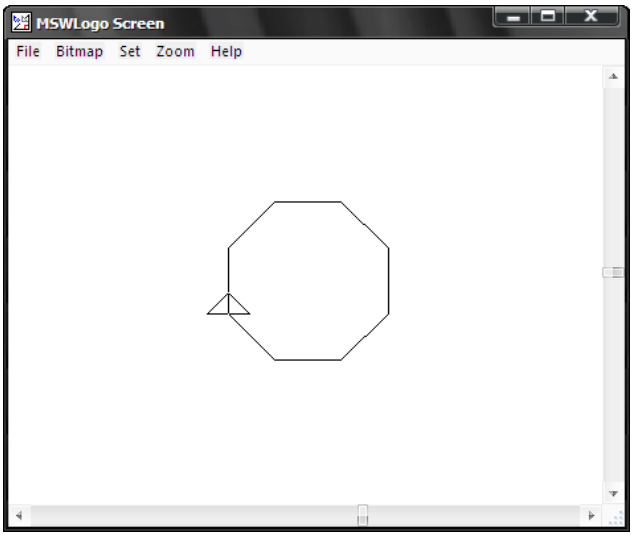
```
to polygon :size :edge
   repeat :size [ fd :edge rt 360/:size ]
end
```

หมายเหตุ คำสั่งที่กำหนดให้ 360/:size หมายถึง การหาค่ามุมเลี้ยว โดยนำ 360 องศาหารด้วยจำนวนด้านของภาพที่ต้องการ จะเป็นค่าของมุมที่เต่าโลโก้ต้องเลี้ยวในแต่ละด้าน มาทดลองเรียกใช้กระบวนงานกันนะครับ

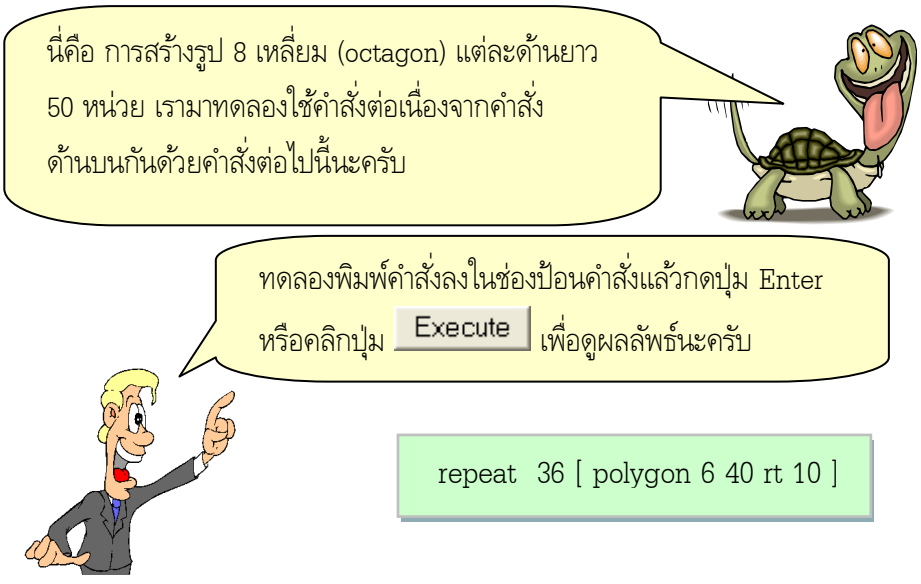
หลังจากสร้างกระบวนงานชื่อ polygon เสร็จแล้ว เราสามารถเรียกใช้กระบวนได้โดยการพิมพ์ชื่อกระบวนงาน และต้องกำหนดค่าตัวแปรลงในช่องป้อนคำสั่ง แล้วกดปุ่ม enter หรือคลิกที่ปุ่ม Execute ดังตัวอย่างการเรียกใช้กระบวนงาน polygon ดังนี้

```
polygon  8  50
```

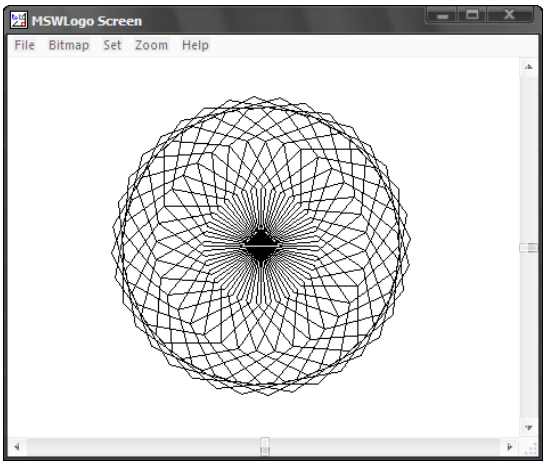
ผลลัพธ์ที่ได้คือ



รูปที่ 5.16 แสดงการใช้ตัวแปรในการรับค่าเพื่อวาดภาพ 8 เหลี่ยม



ผลลัพธ์ที่ได้คือ จะได้ภาพกราฟิกวงกลมที่ได้จากการสร้างภาพ 8 เหลี่ยม



รูปที่ 5.17 แสดงการหมุดภาพ 8 เหลี่ยมได้ตามความต้องการโดยป้อนข้อมูลขนาดที่ต้องการผ่านตัวแปรรับค่าในกระบวนงาน

6. การหยุดการทำงานของกระบวนความ

การหยุดการทำงานของเต่านั้น แต่เดิมเราทำโดยการคลิกที่ปุ่ม Halt หมายถึงการสั่งหยุดการทำงานของเต่า ไม่ใช่การทำงานเมื่อสิ้นสุดคำสั่งตามปกติ ซึ่งไม่เหมาะกับการนำไปใช้งานในกระบวนความทั่วไป การจะทำให้หยุดโดยไม่กด Halt นั้นคือ การใส่เงื่อนไขการหยุดทำงาน คำสั่งที่ใช้คือ IF ซึ่งมีรูปแบบดังนี้

IF เงื่อนไข [Stop]

เช่น ถ้าต้องการให้กระบวนความหยุดทำงานเมื่อตัวแปร size มากกว่า 100 คำสั่งที่ต้องใช้คือ

```
IF :size>100 [stop]
```

หมายเหตุ การหยุดด้วยวิธีนี้ส่วนมากจะใช้ตัวแปรมาควบคุมเงื่อนไข กระบวนความนั้นๆ จึงมีตัวแปรรวมอยู่

ตัวอย่าง

ขั้นตอนที่ 1. สร้างกระบวนความชื่อ s_sqr

```
To s_sqr :size
Repeat 4 [ FD :size RT 90 ]
End
```

ขั้นตอนที่ 2. สร้างกระบวนความ grow1

```
To grow1 :size
s_sqr :size
grow1 :size+10
IF :size > 100 [Stop]
End
```

ขั้นตอนที่ 3. สร้างกระบวนความ grow2

```
To grow2 :size
s_sqr :size
IF :size>100 [stop]
grow2 : size+10
End
```

ขั้นตอนที่ 4. สร้างกระบวนความ grow3

```
To grow3 :size
IF :size>100 [stop]
s_sqr :size
grow3 :size+10
End
```

เมื่อพิมพ์เสร็จแล้วให้เต่าวาด grow1 10 แล้วสั่งให้เต่าวาด grow2 10 แล้วก็สั่ง grow3 10 เปรียบเทียบผลลัพธ์ทั้งสามคำสั่งแล้วบันทึกผล เปรียบเทียบคำสั่งที่แตกต่างกันในกระบวนความทั้งสาม

ทั้งสามกระบวนความนั้นแตกต่างกันที่ตำแหน่งของเงื่อนไข ซึ่งเป็นสิ่งที่สำคัญในการควบคุมการทำงานของกระบวนความ การใส่เงื่อนไขใน grow1 ไม่สามารถหยุดได้เนื่องจากเต่าไม่มีโอกาสที่จะไปทำคำสั่งเงื่อนไขหยุดการทำงาน (แบบนี้ใส่กับไม่ใส่ก็มีค่าเท่ากัน) มาดู grow2 จะหยุดเมื่อมีการวาดรูปที่ใหญ่กว่า 100 แล้ว เพราะจะวาดรูปก่อนที่จะไปตรวจสอบว่ามีขนาดใหญ่กว่า 100 หรือไม่ ส่วน grow3 จะตรวจก่อนเลยว่าใหญ่กว่า100 หรือไม่ แล้วจึงวาดรูป ดังนั้นจะหยุดรูปที่วาดจึงไม่มีรูปไหนที่ขนาดใหญ่กว่า 100 เลยหลักสำคัญที่สุดของกระบวนความเรียกซ้ำนี้คือ การตรวจสอบว่าจะหยุดทำงานเมื่อไร วิธีที่ดีที่สุดในการไล่กระบวนความนี้คือ การไล่ดูคำสั่งทีละบรรทัดโดยสมมติว่าเราคือเต่า ซึ่งทำหน้าที่ทำตามคำสั่งทีละบรรทัดแล้วจะได้ผลลัพธ์อย่างไร

6. การใช้คำสั่ง Wait

เป็นคำสั่งที่ควบคุมการทำงานของเต่าให้หยุดอยู่กับที่ตามจำนวนเวลาที่ที่กำหนด มีรูปแบบคำสั่งดังนี้

รูปแบบคำสั่ง

wait จำนวนเวลาที่ต้องการให้หยุด

เช่น wait 5

หมายถึง ให้เต่าโลโก้หยุดเป็นเวลา 5 มิลลิวินาที

กิจกรรมเสริมทักษะ

เพื่อน ๆ มาลองสร้างกระบวนงานตามขั้นตอนด้านล่าง กันนะคะ แล้วสังเกตผลที่ได้นะคะ

```
สร้างกระบวนความชื่อ Triangle
To polygon :size :edge
   repeat :size [ fd :edge rt 360/:size ]
   repeat 36 [ polygon 6 40 rt 10 wait 10]
end
```

7. การสร้างภาพหลายเหลี่ยมที่ขดซ้อนกัน

เราสามารถอาศัยคำสั่งหลายคำสั่งร่วมกันในการสร้างภาพที่รูปแบบสวยงามต่าง ๆ ได้มากมาย เช่น การสร้างภาพลายเส้นขดซ้อนกัน โดยสร้างให้อยู่ในรูปแบบกระบวนงานได้ดังนี้

ขั้นตอนที่ 1 สร้างกระบวนงาน สามารถกำหนดชื่อได้ตามต้องการ ในที่นี้กำหนดชื่อกระบวนงานว่า spiral และมีตัวแปรที่ใช้ร่วมในการสร้างภาพ 2 ตัวแปร คือ

size แทน ค่าความยาวของด้าน

angle แทน มุมเลี้ยวของรูป

ตัวอย่างการสร้างกระบวนงาน

```
to spiral :size :angle
  if :size > 400 [stop]
  fd :size rt :angle
  spiral :size+2 :angle
end
```

ขั้นตอนที่ 2 เมื่อสร้างกระบวนงานเสร็จแล้ว สามารถเรียกใช้กระบวนงานได้โดย การพิมพ์ชื่อกระบวนงาน และค่าตัวแปรลงในช่องป้อนคำสั่ง แล้วกดปุ่ม enter หรือคลิกที่ปุ่ม Execute ตัวอย่างเช่น

ตัวอย่างที่ 1

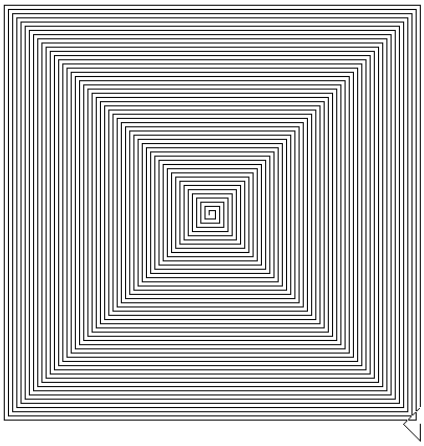
```
spiral 5 152
```

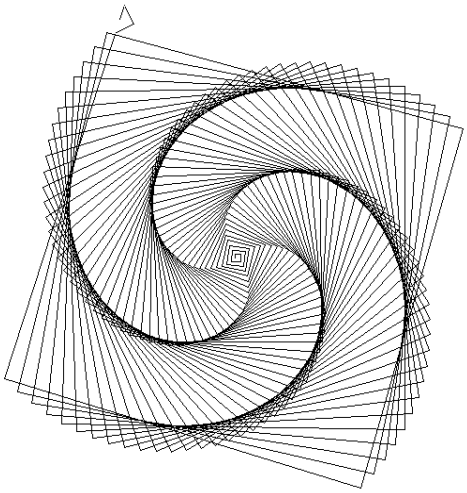
ตัวอย่างที่ 2

```
spiral 3 91
```

4 123

5 456666





รูปที่ 5.18 แสดงการวาดภาพหลายเหลี่ยมที่ขดซ้อนกัน

7. การสร้างภาพรูปทรงกลมและทรงเหลี่ยม

การสร้างภาพรูปทรงกลมจากโปรแกรม MSWLogo เป็นอีกรูปแบบหนึ่งของภาพกราฟิก สามารถทำให้ภาพที่สร้างขึ้นมีลักษณะที่มองดูแล้วเป็นภาพ 3 มิติได้ ซึ่งจะแตกต่างจากภาพลายเส้นธรรมดาที่มีลักษณะ 2 มิติ ดังนั้นการสร้างภาพที่เกิดจากจินตนาการได้ จึงเป็นสิ่งที่ท้าทายความคิดของ

นักเรียนในการใช้องค์ความรู้ที่มีอยู่ให้ได้ในสิ่งที่ประสงค์ คำสั่งสร้างภาพ 3D ในภาษาโลโก้เป็นคำสั่งในการควบคุมเต่าโลโก้เคลื่อนที่ไปตามทิศทางที่เราสั่ง โดยสมมติว่าตัวเต่าโลโก้ทำตัวเหมือนนักบินนั่งอยู่ในห้องควบคุมเครื่องบิน ที่สามารถบินได้ทุกทิศทาง

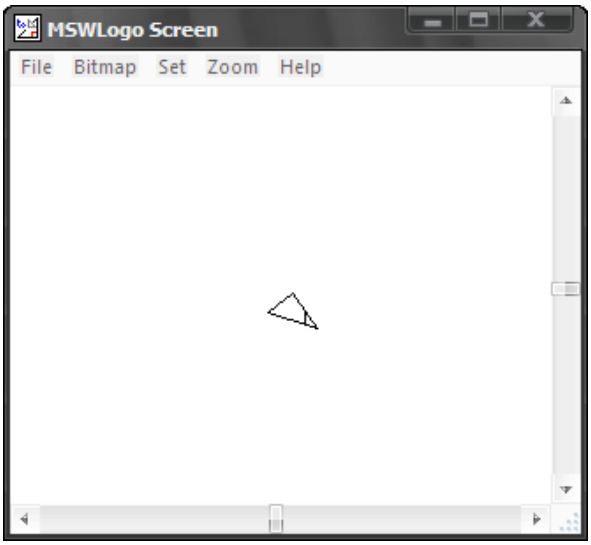
คำสั่งที่ใช้ในการสร้างภาพรูปทรงกลมและทรงเหลี่ยม ประกอบด้วย

7.1 คำสั่ง perspective

คำสั่ง perspective เป็นคำสั่งที่กำหนดให้จอภาพอยู่ในรูปแบบ 3 มิติ เพื่อให้ได้ภาพเหมือนจริงตามสัดส่วนที่ถูกต้อง ตัวเต่าโลโก้ที่ปรากฏในจอจะแตกต่างจากปกติ เนื่องจากเป็นภาพเต่าที่มองจากสัดส่วนใหม่ จะมีลักษณะชี้ขึ้นและบิดเล็กน้อย ดังแสดงในรูป

รูปที่ 5.19 แสดงภาพเต่าโลโก้ในแบบปกติ กับ แบบ 3 มิติ โดยคำสั่ง perspective

หากป้อนคำสั่ง perspective ลงในช่องป้อนคำสั่งแล้ว เต่าโลโก้ก็จะเปลี่ยนตำแหน่งให้อยู่ในรูปแบบ 3 มิติดังรูป



รูปที่ 5.20 แสดงสถานะของเต่าที่เกิดจากคำสั่ง perspective

7.2 คำสั่ง rightroll และ leftroll

เป็นคำสั่งที่ควบคุมทิศทางการม้วนตัวของเต่าโลโก้ให้ม้วนตัวไปตามทิศทางของคำสั่ง rightroll หรือ leftroll ซึ่งมีรูปแบบการใช้คำสั่งดังนี้

- **คำสั่ง rightroll** สามารถใช้คำสั่งได้ 2 รูปแบบ ดังนี้

แบบที่ 1

rightroll ค่ามุม

แบบที่ 2

rr ค่ามุม

ตัวอย่างเช่น

rightroll 90 หรือ rr 90

หมายถึง กำหนดให้เต่าม้วนตัวไปทางขวามือ 90 องศา

- **คำสั่ง leftroll** สามารถใช้คำสั่งได้ 2 รูปแบบ ดังนี้

แบบที่ 1

leftroll ค่ามุม

แบบที่ 2

lr ค่ามุม

ตัวอย่างเช่น

leftroll 90 หรือ lr 90

หมายถึง กำหนดให้เต่าม้วนตัวไปทางซ้ายมือ 90 องศา

7.3 คำสั่ง setpitch

เป็นคำสั่งปรับระดับให้เต่าก้มหรือเงยตามมุมที่เราต้องการให้วาดภาพ มีรูปแบบการใช้คำสั่งดังนี้

รูปแบบคำสั่ง

setpitch ค่ามุม

ตัวอย่างเช่น

setpitch 10

หมายถึง กำหนดให้เต่าเงยขึ้น 10 องศา

setpitch -10

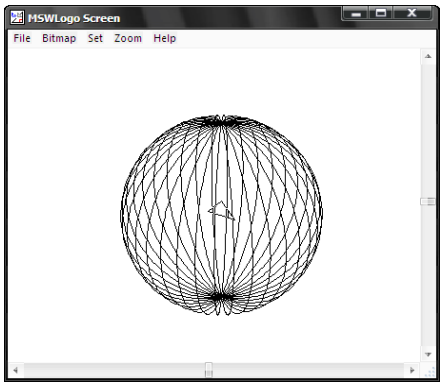
หมายถึง กำหนดให้เต่าก้มลง 10 องศา ซึ่งจะมีค่าเท่ากับการเงยขึ้น 350 องศา

กิจกรรมเสริมทักษะ

เพื่อน ๆ มาลองสร้างกระบวนงานที่ใช้ในการเขียนรูปทรงกลม 3 มิติ แบบง่ายที่มีชื่อกระบวนงาน sphere กันนะคะ

```
to      sphere
        perspective
        repeat 36 [circle  100  rr  10]
end
```

ผลลัพธ์ที่ได้ คือ



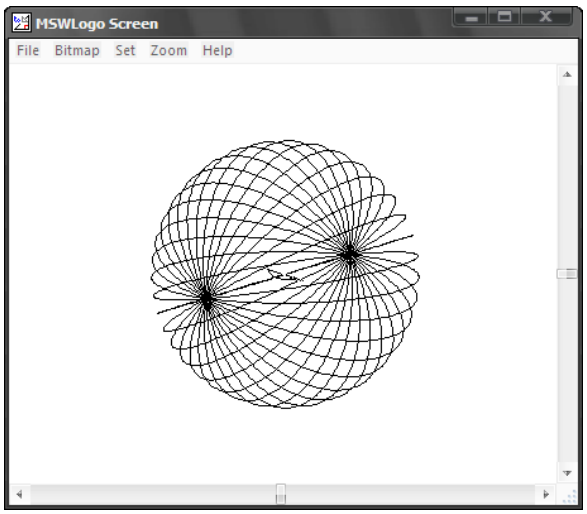
รูปที่ 5.21 แสดงผลลัพธ์ที่ได้จากการหมุนภาพวงกลมในแนวแกนตั้ง

จากตัวอย่าง เป็นการใช้คำสั่ง perspective ให้จอภาพแสดงผลในรูปแบบ 3 มิติ คำสั่งวาดวงกลมที่มีรัศมี 100 หน่วย ซ้ำกัน 36 ครั้ง โดยแต่ละครั้งเต่าโลโก้จะเอียงไปทางขวา 10 องศา ซึ่งการใช้คำสั่ง rr 10 ในกรณีที่ต้องการแกนของรูปทรงกลมอยู่ในตำแหน่งอื่น อาจใช้คำสั่ง setpitch เข้ามาช่วยในการวาดภาพได้

กิจกรรมเสริมทักษะ

เพื่อน ๆ มาลองสร้างกระบวนงานที่ใช้ในการเขียนรูปทรงกลม 3 มิติ ที่มีคำสั่ง setpitch มาช่วยในการกำหนดทิศทางของแกนรูปทรงกลมกันนะคะ

```
to  sphere
     perspective
    setpitch 75
     repeat 36 [circle  100  rr  10]
end
```

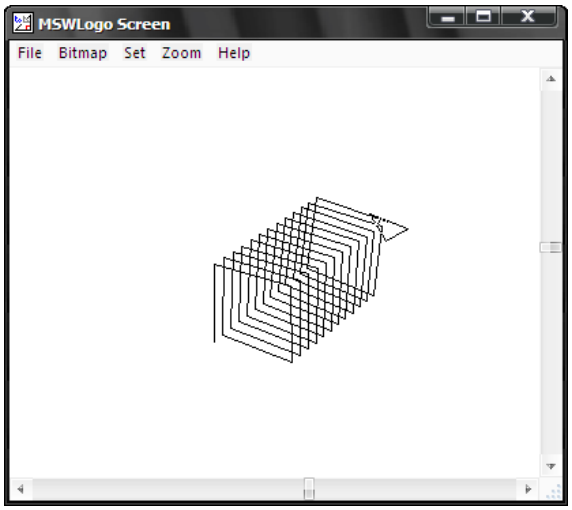


รูปที่ 5.22 แสดงภาพทรงกลมเปลี่ยนแปลงตำแหน่งแกนทรงกลม

กิจกรรมเสริมทักษะ

เพื่อน ๆ มาลองสร้างกระบวนงานวาดภาพรูปสี่เหลี่ยมลูกบาศก์ กันนะคะ

```
to  box
    perspective
    repeat 100 [fd  60  rt  90  100  rr  5]
end
```

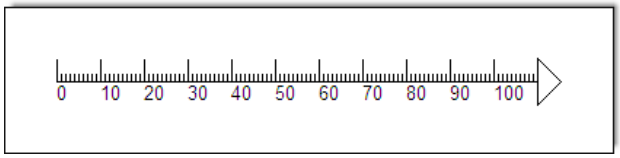


รูปที่ 5.23 ภาพกล่องสี่เหลี่ยมแบบสัดส่วน

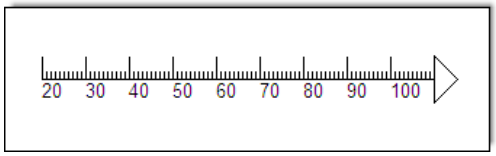
กิจกรรมเสริมทักษะ

เพื่อน ๆ มาลองสร้างกระบวนงานเพื่อสร้างภาพไม้บรรทัดกันนะคะ

```
to  ruler :n
    label :n
    lt   90     fd 15
    rt 180      fd 15
    lt   90     fd  3
    repeat  9 [ lt 90  fd 5  rt 180  fd 5  lt 90  fd 3 ]
    if :n>90 [stop]
    ruler :n+10
end
```



(ก)



(ข)

รูปที่ 5.24 แสดงภาพการสร้างไม้บรรทัด

(ก) ภาพไม้บรรทัดที่ได้จากการใช้คำสั่ง ruler 0

(ข) ภาพไม้บรรทัดที่ได้จากการใช้คำสั่ง ruler 20

8. ฟังก์ชันคณิตศาสตร์

โปรแกรมภาษาโลโก้เป็นโปรแกรมหนึ่งที่มีการพัฒนามาจากโปรแกรมพื้นฐานการคำนวณ แต่ปรับปรุงให้เป็นโปรแกรมฝึกทักษะสำหรับเด็กในการใช้คำสั่งสร้างภาพกราฟิกตามจินตนาการ ขณะเดียวกัน ก็สามารถใช้ประโยชน์ในการคำนวณทางคณิตศาสตร์ได้ด้วย เช่น การบวก ลบ คูณ หาร และอื่น ๆ

8.1 ฟังก์ชันคณิตศาสตร์พื้นฐาน ที่ใช้งานมี 4 ฟังก์ชันดังนี้คือ

- การบวก (Add) สัญลักษณ์คือ +
- การลบ (Subtract) สัญลักษณ์คือ -
- การคูณ (Multiply) สัญลักษณ์คือ *
- การหาร (Devide) สัญลักษณ์คือ /

8.2 คำสั่งฟังก์ชันคณิตศาสตร์ สำหรับภาษาโลโก้ คือ

- **การประการตัวแปรและการกำหนดค่าให้กับตัวแปร** ทำได้โดยใช้คำสั่ง make ได้ดังนี้

รูปแบบ

```
make  ชื่อตัวแปร  ค่าของตัวแปร
```

ตัวอย่างเช่น

make size 9 (หมายถึง กำหนดตัวแปรชื่อว่า size มีค่าเท่ากับ 9)

- **การใช้งานฟังก์ชันคณิตศาสตร์** ทำได้โดยใช้คำสั่ง label เพื่อกำหนดให้มีการคำนวณจากค่าที่กำหนดได้ โดยมีรูปแบบการใช้งานดังนี้

รูปแบบ

```
label   ตัวเลข  ฟังก์ชัน   ตัวเลข
label   :ตัวแปร  ฟังก์ชัน  ตัวเลข
label   :ตัวแปร  ฟังก์ชัน  :ตัวแปร
```

ตัวอย่างที่ 1

label 2 * 4

(หมายถึง แสดงผลลัพธ์การคำนวณค่า 2 คูณ 4 ผลลัพธ์ที่ได้คือ 8)

ตัวอย่างที่ 2

make size 9

(หมายถึง กำหนดตัวแปรชื่อว่า size มีค่าเท่ากับ 9)

label 360/:size

(หมายถึง แสดงผลลัพธ์การคำนวณค่าของ 360 หารด้วยค่าของตัวแปรที่ชื่อว่า size ในที่นี้คือ 9 ผลลัพธ์ที่ได้คือ 360/9=40)

9. การใช้คำสั่งตารางสีกับภาพกราฟิก

การเลือกสีจากตารางสีให้กับภาพกราฟิก เช่น สีของปากกา สีพื้นหลังในหน้าต่างแสดงผล และในการเติมสีให้กับภาพกราฟิก สามารถเลือกสีได้จาก Menu Bar ในเมนู Set แล้วเลือกรายการที่ต้องการเพื่อกำหนดสีตามต้องการ แต่ในกระบวนงานไม่สามารถทำแบบนั้นได้ ต้องเขียนกระบวนงานให้ควบคุมการเปลี่ยนสีโดยเลือกใช้สีได้จากตารางต่อไปนี้

ตารางสีสำหรับโปรแกรม MSWLogo

ชื่อสี	ดรรชนี	ค่า RGB	ชื่อสี	ดรรชนี	ค่า RGB
Black (ดำ)	0	[0 0 0]	Brown (น้ำตาล)	8	[155 96 59]
Blue (น้ำเงิน)	1	[0 0 255]	Light Brown	9	[197 136 18]
Green (เขียว)	2	[0 255 0]	Mid-Green	10	[100 162 64]
Cyan	3	[0 255 255]	Blue-Green	11	[120 187 187]
Red (แดง)	4	[255 0 0]	Salmon	12	[255 149 119]
Magenta	5	[255 0 255]	Blue-ish	13	[144 113 208]
Yellow (เหลือง)	6	[255 255 0]	Orange (ส้ม)	14	[255 163 0]
White (ขาว)	7	[255 255 255]	Silver (เงิน)	15	[183 183 183]

9.1 คำสั่งกำหนดสีของปากกา (Pen color)

รูปแบบคำสั่ง setpencolor [R G B] หรือ setpc [R G B]

เช่น setpc [255 0 0]

หมายถึง กำหนดสีปากกาเป็นสีแดง

9.2 คำสั่งกำหนดสีพื้นหลังในหน้าต่างแสดงผล (screen color)

รูปแบบคำสั่ง setscreencolor [R G B] หรือ setsc [R G B]

เช่น setsc [0 0 255]

หมายถึง กำหนดสีพื้นหลังในหน้าต่างแสดงผลเป็นสีเขียว

9.3 คำสั่งกำหนดสีเติมภาพกราฟิก (flood color)

รูปแบบคำสั่ง setsfloodcolor [R G B] หรือ setfc [R G B]

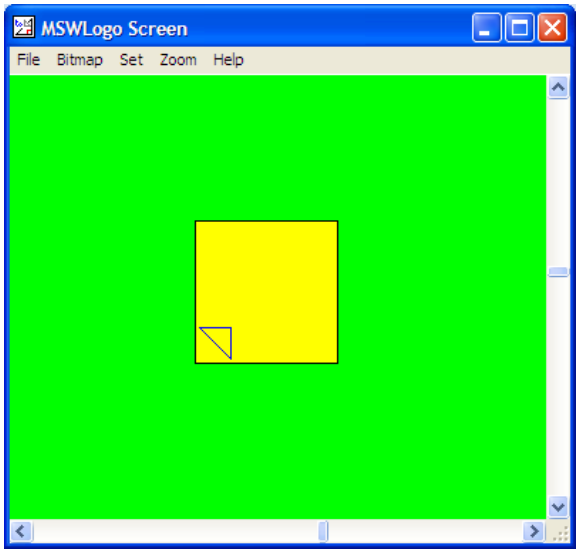
เช่น setfc [0 255 0]

หมายถึง กำหนดสีเติมภาพกราฟิกเป็นสีน้ำเงิน

การระบายสีภาพกราฟิกด้วยคำสั่ง floodcolor สามารถทำได้โดยใช้คำสั่ง fill ช่วยอีกครั้งหนึ่ง คือ หลังจากกำหนดสีเติมภาพกราฟิกตามต้องการแล้ว ให้ใช้คำสั่ง fill ในการเติมสีให้กับภาพนั้น ๆ โดยเต่าต้องอยู่ในตำแหน่งพื้นที่ของภาพกราฟิกเสียก่อนจึงจะใช้คำสั่ง fill เพื่อเติมสีให้กับภาพได้

ตัวอย่างการใช้คำสั่ง

```
setsc [ 0  255  0 ]
setpc [ 0  0  0 ]
repeat 4 [fd 100 rt 90]
rt  45
pu
fd  20
setfc [ 255  255  0 ]
fill
```



รูปที่ 5.24 ภาพที่ได้จากการใช้คำสั่ง setflood color

9.4 คำสั่งทดสอบดูสีที่สร้างขึ้น (show color) โดยป้อนคำสั่งในช่องป้อนคำสั่ง (Input Box) เพื่อทดสอบสีที่สร้างขึ้น

รูปแบบคำสั่ง

show pen color หรือ show pc

หมายถึง การทดสอบดูสีปากกาที่สร้างขึ้น

show screen color หรือ show sc

หมายถึง การทดสอบดูสีพื้นหลังในหน้าต่างแสดงผล

show flood color หรือ show fc

หมายถึง การทดสอบดูสีสีเติมภาพกราฟิก