บทที่ 2
การแก้ปัญหาด้วยไพทอน

หัวข้อการเรียนรู้

1. ตัวอย่างระบบคำนวณค่าโดยสารรถประจำทาง
2. ตัวดำเนินการบูลีน
3. การวนซ้ำด้วยคำสั่ง while
4. เงื่อนไข
5. ฟังก์ชัน

จุดประสงค์ของบทเรียน

เมื่อนักเรียนเรียนจบบทนี้แล้ว นักเรียนสามารถ

1. เขียนโปรแกรมไพทอนที่มีการใช้งานฟังก์ชันที่สร้างขึ้นเอง
2. เขียนโปรแกรมไพทอนที่มีการใช้ตัวดำเนินการบูลีน

การแก้ปัญหาด้วยไพทอน

ในการดำเนินชีวิตประจำวัน มีหลายเหตุการณ์ที่ต้องอาศัยการตัดสินใจ ภายใต้เงื่อนไขต่าง ๆ เพื่อให้ได้ผลลัพธ์ที่ต้องการ เช่น การเลือกโปรโมชันโทรศัพท์มือถือ การเลือกเส้นทางไปโรงเรียนในวันที่ฝนตกน้ำท่วม การเลือกสวมใส่เสื้อผ้าให้เหมาะสมกับสภาพอากาศเพื่อที่จะออกไปซื้อของกับเพื่อนในวันหยุด

```
num = int (input ('Enter your guess: '))
print ("You've got it right.")
```

ทบทวนความรู้ก่อนเรียน

เขียนเครื่องหมาย ✓ หน้าข้อความที่ถูกต้อง

☐ อัลกอริทึมเป็นองค์ประกอบของแนวคิดเชิงคำนวณ

☐ การแตกปัญหาเป็นส่วนย่อย เป็นการลดความซับซ้อนของปัญหาใหญ่ ทำให้ออกแบบวิธีการแก้ปัญหาได้ง่ายยิ่งขึ้น

☐ วิธีการแก้ปัญหาหนึ่งอาจนำมาใช้ในการแก้ปัญหาย่อยของอีกปัญหาหนึ่งได้

☐ ในการทำความสะอาดห้อง นักเรียนสามารถแบ่งการทำงานออกเป็นงานย่อยให้เพื่อนแต่ละคนช่วยทำได้

ลองทำดู

1. ให้นักเรียนพิจารณาขั้นตอนการทำความสะอาดเสื้อผ้าที่ใส่แล้ว ตลอดจนการเก็บเสื้อผ้าที่ทำความสะอาดแล้วไว้ในตู้เสื้อผ้า ว่ามีการทำงานอะไรบ้าง
2. ใช้โมดูล turtle ของไพทอนวาดรูปวงแหวนที่ประกอบด้วยวงกลมสองวงซ้อนกัน

นักเรียนได้เรียนรู้การโปรแกรมด้วยไพทอนที่มีการทำงานแบบลำดับ มีทางเลือก และวนซ้ำโดยมีการรับข้อมูล แสดงผล ใช้งานตัวดำเนินการ ตัวแปร เพื่อคำนวณทางคณิตศาสตร์อย่างง่ายมาแล้ว สำหรับบทเรียนนี้ นักเรียนจะได้เขียนโปรแกรมที่มีฟังก์ชันและมีการใช้งานตัวดำเนินการบูลีน เพื่อให้สามารถแก้ปัญหาที่ซ้อนซ้อนขึ้นได้อย่างมีประสิทธิภาพ

2.1 ตัวอย่างระบบคำนวณค่าโดยสารรถประจำทาง

หากผู้ปกครองพานักเรียนอนุบาลขึ้นรถโดยสารประจำทางไปโรงเรียน คนเก็บค่าโดยสารต้องคำนวณว่าผู้ปกครองและนักเรียนจะต้องเสียค่าโดยสารคนละเท่าใด โดยขึ้นอยู่กับเงื่อนไขอายุหรือส่วนสูง เป็นต้น หากต้องการระบบเก็บค่าโดยสารอัตโนมัติ ระบบนี้จะต้องคำนวณค่าโดยสารได้เอง จากเงื่อนไขที่ถูกกำหนดไว้ล่วงหน้าแล้ว

ตัวอย่างที่ 2.1 ค่ารถโดยสาร

รถโดยสารสาธารณะในอำเภอหนึ่ง ประกาศอัตราค่าโดยสารไว้ดังนี้

☞ ผู้โดยสารทั่วไป คิดอัตราคนละ 10 บาทตลอดเส้นทาง

☞ ผู้โดยสารที่เป็นเด็กอายุต่ำกว่า 3 ขวบ โดยสารฟรี

☞ ผู้โดยสารสูงอายุที่มีอายุตั้งแต่ 60 ปีขึ้นไป คิดค่าโดยสารครึ่งราคา

☞ ราคาค่าโดยสารสำหรับหมู่คณะที่ประกอบด้วยผู้โดยสารไม่เกิน 30 คน ถ้าค่าโดยสารรวมเป็นเป็นจำนวนตั้งแต่ 200 บาทขึ้นไปจะมีส่วนลดเพิ่มอีก 10%

ถ้านักเรียนต้องการนำคนในหมู่บ้านไปทัศนศึกษา และต้องการคำนวณค่ารถโดยสารจะออกแบบอัลกอริทึมอย่างไร

เพื่อแก้ปัญหาข้างต้น นักเรียนควรพิจารณาการแก้ปัญหาย่อยต่อไปนี้

1. ทราบได้อย่าไรว่ามีผู้โดยสารเป็นเด็กอายุต่ำกว่า 3 ขวบกี่คน เป็นผู้ใหญ่อายุมากกว่า 60 ปีกี่คน และที่เหลือกี่คน

ตอบ นับจำนวนผู้โดยสารว่ามีทั้งหมดกี่คน เป็นเด็กที่มีอายุต่ำกว่า 3 ขวบกี่คน เป็นผู้สูงอายุกี่คน จำนวนผู้โดยสารที่ต้องชำระค่าโดยสารเต็มราคากี่คน ซึ่งคำนวณได้จากจำนวนผู้โดยสารทั้งหมด ลบด้วยจำนวนเด็กที่อายุต่ำกว่า 3 ขวบ และลบด้วยจำนวนผู้สูงอายุ

2. ค่าโดยสารรวมของผู้สูงอายุเป็นเท่าไร

ตอบ ค่าโดยสารรวมของผู้สูงอายุ คำนวณได้จาก จำนวนผู้สูงอายุคูณกับค่าโดยสาร 5 บาท (ครึ่งหนึ่งของ 10 บาท)

3. ค่าโดยสารรวมของผู้โดยสารที่จ่ายเต็มราคาเป็นเท่าไร

ตอบ ค่าโดยสารรวมของผู้โดยสารที่จ่ายเต็มราคา คำนวณได้จาก จำนวนผู้โดยสารที่เหลือ (ผลลัพธ์จากข้อ 1) คูณกับ 10 บาท

4. ค่าโดยสารรวมทั้งหมดเป็นเท่าไรก่อนลดราคา

ตอบ ค่าโดยสารรวมทั้งหมด คำนวณได้จาก ค่าโดยสารรวมของผู้สูงอายุ (ผลลัพธ์จากข้อ 2) บวกับค่าโดยสารรวมของผู้โดยสารที่จ่ายเต็มราคม (ผลลัพธ์จากข้อ 3)

5. ได้รับส่วนลด 10% หรือไม่

ตอบ ได้รับส่วนลดเมื่อมีจำนวนผู้โดยสารไม่เกิน 30 คน และมีค่าโดยสารรวมตั้งแต่ 200 บาทขึ้นไป

6. หากได้รับส่วนลด ค่าโดยสารสุทธิเป็นเท่าไร

ตอบ ค่าโดยสารสุทธิคำนวณได้จาก ค่าโดยสารรวมทั้งหมด (ผลลัพธ์จากข้อ 4) ลบด้วย 10% ของค่าโดยสารรวมทั้งหมด

7. แสดงผลอะไรบ้าง

ตอบ แสดงจำนวนผู้โดยสารทั้งหมด จำนวนผู้โดยสารที่จ่ายเต็มราคา จำนวนผู้โดยสารที่เป็นเด็กอายุต่ำกว่า 3 ขวบ จำนวนผู้โดยสารที่เป็นผู้สูงอายุ ค่าโดยสารรวมก่อนลดราคา และค่าโดยสารสุทธิหลังหักส่วนลด

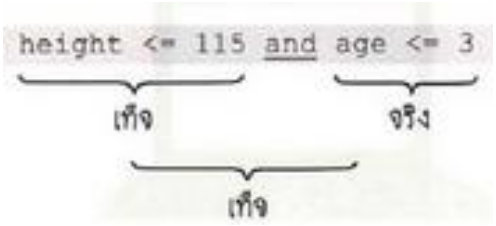
นำแนวทางการแก้ปัญหาย่อยที่ได้มาเรียบเรียงเป็นอัลกอริทึมในรูปแบบรหัสลำลอง ได้ดังนี้

1. all ← รับจำนวนผู้โดยสารทั้งหมด
2. children ← รับจำนวนผู้โดยสารที่มีอายุต่ำกว่า 3 ขวบ
3. elders ← รับจำนวนผู้สูงอายุ (มีอายุตั้งแต่ 60 ปีขึ้นไป)
4. regular ← (all – children – elders)
5. fare_elders ← elders * (10/2)
6. fare_regular ← regular * 10
7. total ← fare_elders + fare_regalar
8. แสดงค่า all, children, elders, regular
9. แสดงค่า total
10. ถ้า all <= 30 และ total >=200
    - 10.1 total_discounted ← total – (total * 0.1)
    - 10.2 แสดงค่า total_discounted

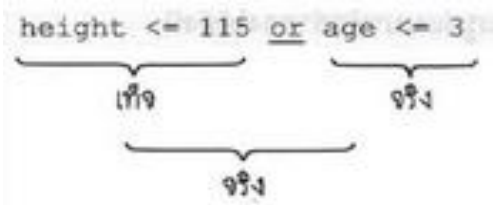
2.2 ตัวดำเนินการบูลีน

ในชั้นมัธยมศึกษาปีที่ 1 นักเรียนสามารถเขียนโปรแกรมไพทอนให้มีการตัดสินใจทำงานแบบมีทางเลือกโดยใช้คำสั่ง if และ if – else ในเบื้องต้นมาแล้ว ในบทนี้นักเรียนจะได้เรียนรู้นิพจน์เปรียบเทียบ และตัวดำเนินการบูลีน เพื่อที่จะสามารถกำหนดเงื่อนไขสำหรับการทำงานแบบมีทางเลือกของคำสั่ง if ได้อย่างมีประสิทธิภาพมากขึ้น

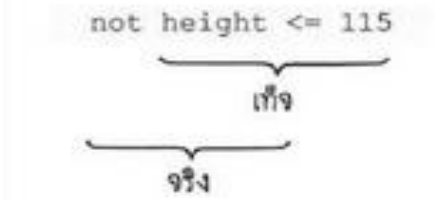
นิพจน์เปรียบเทียบที่ได้กล่าวมาแล้วเป็นนิพจน์เปรียบเทียบอย่างง่าย โดยเป็นการเปรียบเทียบค่าชนิดเดียวกันด้วยตัวดำเนินการเปรียบเทียบเท่านั้น แต่หากต้องการกำหนดเงื่อนไขที่ซับซ้อนขึ้น สามารถใช้ตัวดำเนินการบูลีนได้แก่ and, or หรือ not ในการเชื่อมต่อนิพจน์เปรียบเทียบอย่างง่ายเข้าด้วยกันได้



ผลลัพธ์ที่ได้จากการเชื่อมนิพจน์ 2 นิพจน์ด้วย and จะเป็นจริง (True) ถ้าทั้งสองนิพจน์เป็นจริงทั้งคู่ ส่วนกรณีอื่นๆ จะเป็นเท็จ (False) เช่น ถ้าตัวแปร height เก็บค่า 120 และตัวแปร age เก็บค่า 3 แล้วมีเงื่อนไข height <= 115 and age <= 3 ผลลัพธ์ที่ได้จะเป็นเท็จ



นิพจน์ที่เชื่อมด้วย or จะเป็นจริง ถ้าอย่างน้อยหนึ่งนิพจน์เป็นจริง (True) ตัวอย่างเช่น ถ้าตัวแปร height เก็บค่า 120 และตัวแปร age เก็บค่า 3 แล้ว มีเงื่อนไข height <= 115 or age <= 3 ผลลัพธ์ที่ได้จะเป็นจริง



นิพจน์ที่มี not นำหน้าจะมีค่าความจริงเป็นตรงกันข้าม ตัวอย่างเช่น ถ้าตัวแปร height เก็บค่า 120 แล้วมีเงื่อนไข not height <= 115 ผลลัพธ์ที่ได้จะเป็นจริง

ตัวอย่างที่ 2.2 การหาผลลัพธ์ค่าความจริงจากนิพจน์เปรียบเทียบที่มีการใช้ตัวดำเนินการบูลีน ทดลองพิมพ์คำสั่งต่อไปนี้ในคอลโซล

```
age = 12
age > 13 and age <19
age >13 or age <19
not age > 13
```

จะได้ผลลัพธ์ คือ

```
False
True
True
```

ชวนคิด

จากตัวอย่างที่ 2.2 ถ้าเปลี่ยนค่าของตัวแปร age ไปเป็นค่าอื่นๆ ผลลัพธ์ที่ได้จะเปลี่ยนไปอย่างไร

ตัวอย่างที่ 2.3 เขียนโปรแกรมภาษาไพทอนเพื่อแก้ปัญหาการคิดค่าโดยสารในตัวอย่างที่ 2.1

```
all = int (input ('มีผู้โดยสารทั้งหมดกี่คน: '))                    # บรรทัดที่ 1
children = int (input ('มีผู้โดยสารอายุต่ำกว่า 3 ขวบกี่คน: '))      # บรรทัดที่ 2
elders = int (input ('มีผู้โดยสารอายุตั้งแต่ 60 ปีขึ้นไปกี่คน: ')) # บรรทัดที่ 3
regular = (all – children – elders)                                # บรรทัดที่ 4
fare_elders = elders * (10/2)                                      # บรรทัดที่ 5
fare_regular = regular * 10                                        # บรรทัดที่ 6
total = fare_elders + fare_regular                                 # บรรทัดที่ 7
print ('จำนวนผู้โดยสารทั้งหมด', all, 'คน')                         # บรรทัดที่ 8
print ('เป็นเด็กอายุต่ำกว่า 3 ขวบ', children, 'คน')                # บรรทัดที่ 9
print ('เป็นผู้สูงอายุ', elders, 'คน')                             # บรรทัดที่ 10
print ('เป็นผู้โดยสารคิดอัตราปกติ', regular, 'คน')                 # บรรทัดที่ 11
print ('ค่าโดยสารรวมทั้งสิ้น', total, 'บาท')                      # บรรทัดที่ 12
if all <= 30 and total >= 200:                                     # บรรทัดที่ 13
    total_discounted = total – (total * 0.1)                       # บรรทัดที่ 14
    print ('ลดราคา 10% คิดเป็นค่าโดยสารทั้งสิ้น', total_discounted, 'บาท'  # บรรทัดที่ 15
```

ตัวอย่างที่ 2.3 อธิบายได้ดังนี้

1. บรรทัดที่ 1 ถึง 12 เป็นการใช้คำสั่งไพทอนตามที่ได้เคยศึกษามาแล้ว
2. บรรทัดที่ 13 กำหนดเงื่อนไขสำหรับตรวจสอบว่าต้องลดค่าโดยสาร 10% หรือไม่ ในกรณีที่จำนวนผู้โดยสารรวม (ตัวแปร all) ไม่เกิน 30 คน ค่าโดยสารรวม (ตัวแปร total) มีค่าตั้งแต่ 200 บาทขึ้นไป และถ้าเงื่อนไขเป็นจริงจะคำนวณราคาที่ลดแล้วและแสดงผลในบรรทัดที่ 15

ชวนคิด

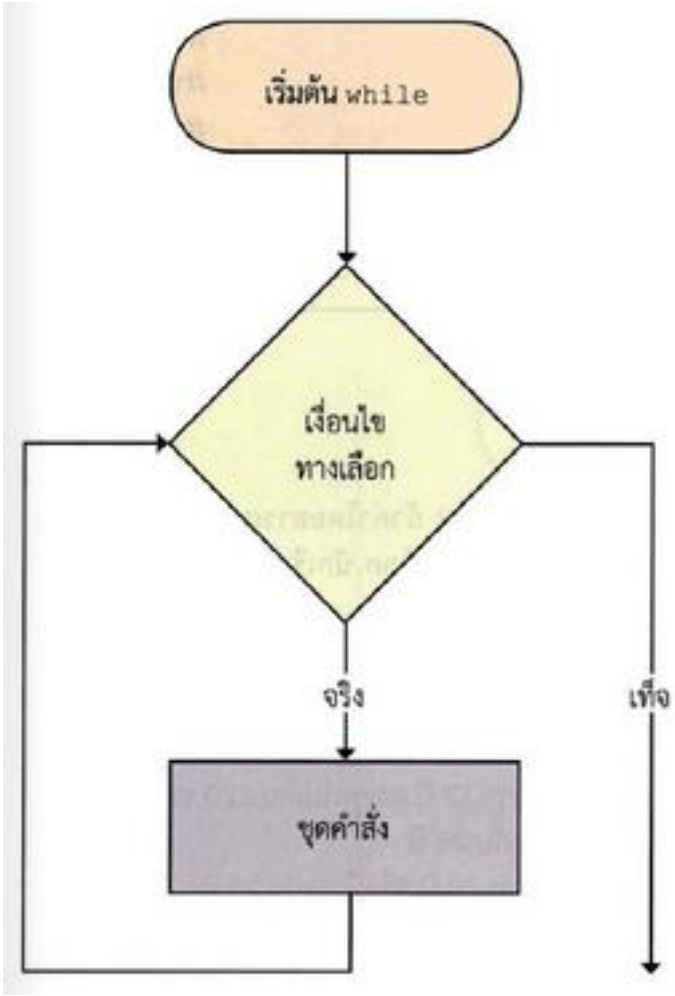
1. จากตัวอย่างที่ 2.3 ถ้าค่าโดยสารเปลี่ยนแปลงตามราคาน้ำมันของตลาดโลก นักเรียนจะปรับโปรแกรมที่บรรทัดใดอย่างไร
2. ถ้ามีตัวแปร age และ height เก็บค่าอายุเป็นปีและส่วนสูงเป็นเซนติเมตร ให้เขียนนิพจน์ที่แทนข้อความต่อไปนี้

- อายุน้อยกว่า 12 ปี และสูงไม่เกิด 120 ซม.
- อายุไม่เท่ากับ 25 ปี
- อายุมากกว่า 60 ปี หรือน้อยกว่า 15 ปี
- ความสูงอยู่ในช่วงตั้งแต่ 125 ซม. ถึง 180 ซม.

2.3 การวนซ้ำด้วยคำสั่ง while

ในการเขียนโปรแกรมให้ทำงานวนซ้ำชุดคำสั่งเดิม นอกจากคำสั่ง for แล้ว ภาษาไพทอนยังมีคำสั่ง while ให้เลือกใช้งาน ซึ่งคำสั่ง while จะเหมาะสมกับกรณีการวนซ้ำที่ไม่ทราบจำนวนรอบหรือจำนวนครั้งของการวนซ้ำที่แน่นอน มีรูปแบบการใช้ดังนี้

```
while เงื่อนไขทางเลือก:
        ชุดคำสั่ง
```



การทำงานของคำสั่ง while คือ ถ้า *เงื่อนไขทางเลือก* เป็นจริง *ชุดคำสั่ง*จะถูกเรียกให้ทำงานเป็นจำนวนหนึ่งครั้ง แล้วจะวนกลับไปตรวจสอบ*เงื่อนไขทางเลือก* อีกจนกว่า *เงื่อนไขทางเลือก*จะเป็นเท็จ แล้วจึงจะออกจากการวนซ้ำและไปทำงานในคำสั่งถัดไป

ตัวอย่างที่ 2.4 การใช้คำสั่ง while เมื่อไม่ทราบจำนวนรอบการวนซ้ำที่แน่นอน

เขียนโปรแกรมรับจำนวนเต็มบวกชุดหนึ่งไม่ทราบล่วงหน้าว่ามีกี่จำนวน แล้วหาผลรวมและค่ามากที่สุด

```
x = int (input (‘ป้อนจำนวนเต็มบวก หรือป้อน 0 หากไม่ต้องการป้อนข้อมูลตัวต่อไป ‘))     # บรรทัดที่ 1
sum = 0                                                                              # บรรทัดที่ 2
max = 0                                                                              # บรรทัดที่ 3
n = 0                                                                                # บรรทัดที่ 4
while x > 0:                                                                         # บรรทัดที่ 5
        n = n +1                                                                     # บรรทัดที่ 6
        sum = sum + x                                                                # บรรทัดที่ 7
        if (x > max):                                                                # บรรทัดที่ 8
                max = x                                                              # บรรทัดที่ 9
        x = int (input (‘ป้อนจำนวนเต็มบวก หรือป้อน 0 หากไม่ต้องการป้อนข้อมูลตัวต่อไป ‘))  # บรรทัดที่ 10
if n > 0:                                                                            # บรรทัดที่ 11
        print (‘ผลรวม’, sum, ‘และค่ามากที่สุด’, max)                                  # บรรทัดที่ 12
```

ตัวอย่างที่ 2.4 อธิบายได้ดังนี้

1. บรรทัดที่ 1 รับข้อมูลจำนวนเต็มบวกตัวแรก เก็บไว้ในตัวแปร x โดยผู้ใช้จะป้อน 0 หากไม่มีข้อมูลแล้ว เนื่องจากข้อมูลที่ต้องการให้หาผลลัพธ์เป็นจำนวนบวกเท่านั้น จึงใช้เลข 0 เป็นเงื่อนไขในการตรวจสอบว่าหมดชุดข้อมูลแล้ว
2. บรรทัดที่ 2 ถึง 4 กำหนดค่าเริ่มต้นให้กับตัวแปรที่จะใช้เก็บค่าผลรวม (sum) ค่ามากที่สุด (max) และจำนวนข้อมูล (n)
3. บรรทัดที่ 5 เป็นคำสั่ง while และกำหนดเงื่อนไขตรวจสอบการวนซ้ำ คือ x ต้องเป็นจำนวนเต็มบวก
4. บรรทัดที่ 6 ถึง 10 เป็นชุดคำสั่งภายใต้บล็อก while โดยบรรทัดที่ 6 นับจำนวนข้อมูลเก็บไว้ในตัวแปร n บรรทัดที่ 7 เก็บผลรวมข้อมูลไว้ในตัวแปร sum บรรทัดที่ 8 ถึง 9 หาค่ามากที่สุดจากข้อมูลทั้งหมด และบรรทัดที่ 10 รับข้อมูลตัวถัดไป
5. บรรทัดที่ 11 ถึง 12 ถ้ามีข้อมูลตั้งแต่ 1 ตัวขึ้นไป ให้พิมพ์ผลลัพธ์ที่หาได้

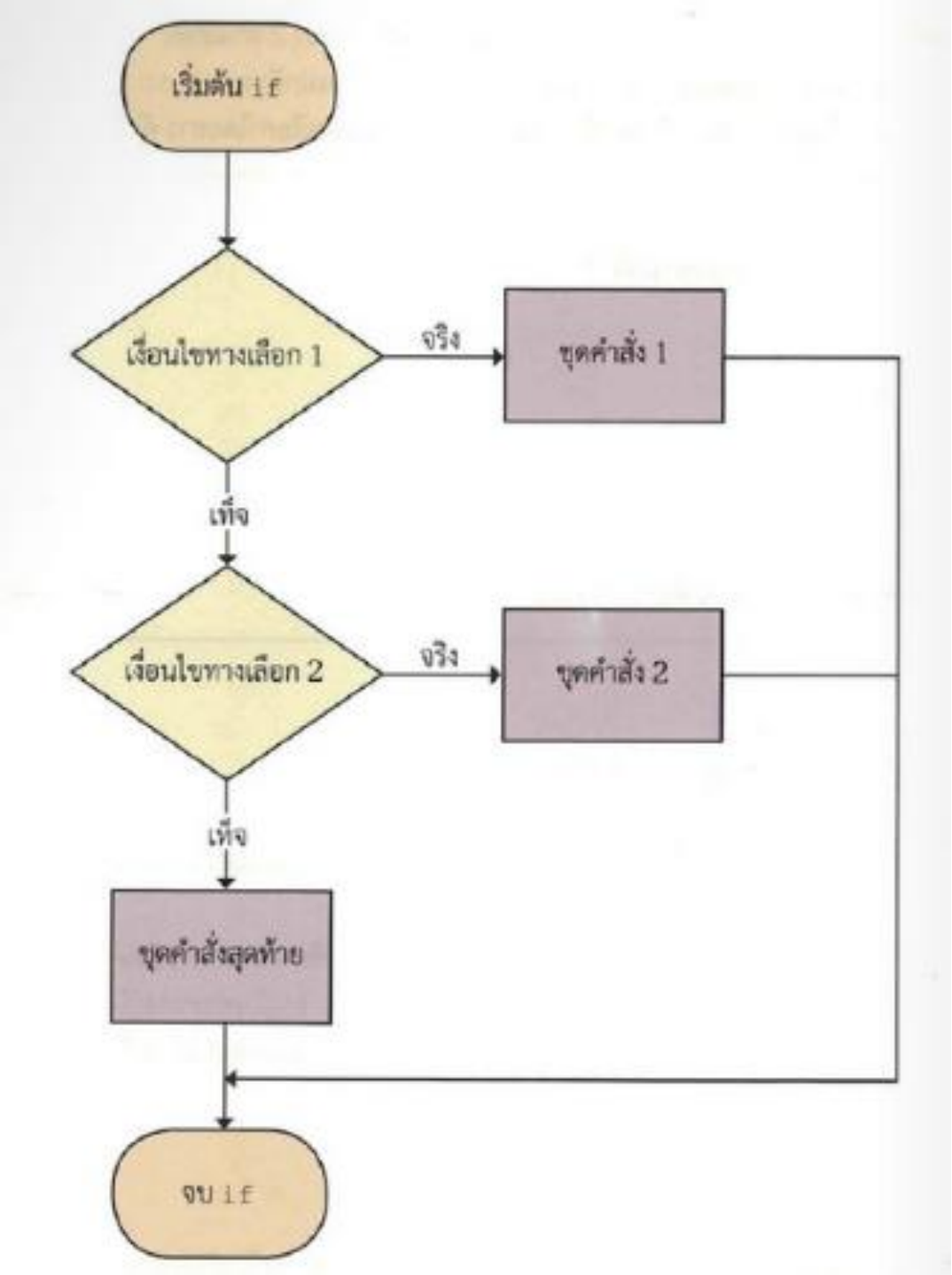
ชวนคิด

ในชีวิตประจำวันมีสถานการณ์ใดบ้างที่มีการวนซ้ำคล้ายการใช้คำสั่ง while

2.4 เงื่อนไขทางเลือก

คำสั่ง if – else ช่วยให้โปรแกรมไพทอนสามารถตัดสินใจเลือกทำงานชุดคำสั่งตามผลลัพธ์ของเงื่อนไข ซึ่งมีสองทางเลือก และหากมีทางเลือกมากกว่านั้น ผู้เขียนโปรแกรมต้องใช้คำสั่ง if – else ร่วมกัน หรือซ้อนกัน เพื่อให้ได้จำนวนทางเลือกตามที่ต้องการ ไพทอนยังมีคำสั่งให้ใช้ในกรณีที่หลายทางเลือก คือ คำสั่ง if – elif – else ซึ่งจะช่วยให้สะดวกขึ้น โดยมีรูปแบบการใช้ดังนี้

```
if เงื่อนไขทางเลือก 1:
        ชุดคำสั่ง 1
elif เงื่อนไขทางเลือก 2:
        ชุดคำสั่ง 2
elif เงื่อนไขทางเลือก 3:
        ชุดคำสั่ง 3
...
else :
        ชุดคำสั่งสุดท้าย
```



ผู้เขียนโปรแกรมสามารถเพิ่มเติมเงื่อนไขทางเลือก และชุดคำสั่งที่สัมพันธ์กันภายใต้บล็อกของ elif ไปได้อีกและสำหรับเงื่อนไข else อาจจะไม่จำเป็นต้องมีก็ได้

ตัวอย่างที่ 2.5 การใช้คำสั่ง if – elif – else สำหรับกรณีที่มากกว่า 2 ทางเลือก

เขียนโปรแกรมไพทอนเพื่อแจ้งราคาค่าโดยสารรถประจำทาง ซึ่งปกติราคา 6.50 บาท แต่ถ้าเป็นผู้สูงอายุจะเสียค่าโดยสารเพียงครึ่งเดียว และถ้าเป็นเด็กที่อายุไม่เกิน 3 ขวบ ไม่ต้องเสียค่าโดยสาร ดังนี้

```
fare = 6.50                                                    # บรรทัดที่ 1
age = int (input (“อายุของท่าน คือ : “))                       # บรรทัดที่ 2
if age >=60 :                                                  # บรรทัดที่ 3
        fare =  fare / 2                                       # บรรทัดที่ 4
elif age <= 3 :                                                # บรรทัดที่ 5
        fare = 0                                               # บรรทัดที่ 6
else :                                                         # บรรทัดที่ 7
        fare = 6.50                                            # บรรทัดที่ 8
print (“ค่าโดยสารของท่าน คือ: ”, fare)                         # บรรทัดที่ 9
```

ตัวอย่างที่ 2.5 อธิบายได้ดังนี้

1. บรรทัดที่ 1 กำหนดค่าโดยสาร 6.50 ไว้ในตัวแปร fare
2. บรรทัดที่ 2 รับอายุเก็บไว้ในตัวแปร age
3. บรรทัดที่ 3 ตรวจสอบเงื่อนไข ถ้าอายุตั้งแต่ 60 ปีขึ้นไปเป็นจริง จะลดราคาค่าโดยสารครึ่งหนึ่งแล้วเก็บไว้ในตัวแปร fare จะเป็น fare/2
4. บรรทัดที่ 5 เป็นกรณีที่อายุน้อยกว่า 60 และไม่เกิน 3 ขวบ ซึ่งใช้คำสั่ง elif เพื่อสร้างทางเลือกนี้
5. บรรทัดที่ 7 เป็นกรณีอื่นนอกเหนือจากกรณีที่ระบุไปแล้ว คือ อายุในช่วงของผู้โดยสารทั่วไป จะพบว่าการใช้ if – elif – else ทำให้การเขียนโปรแกรมกะทัดรัด และตรวจสอบได้ง่ายขึ้น

ชวนคิด

ในโปรแกรมตัวอย่างที่ 2.5 นักเรียนสามารถตัดบรรทัดใดออกได้บ้าง โดยโปรแกรมยังทำงานได้ผลลัพธ์เหมือนเดิม

ตัวอย่างที่ 2.6 การหาพื้นที่ของวงกลม สามเหลี่ยม หรือสี่เหลี่ยม

ให้เขียนโปรแกรมภาษาไพทอนเพื่อผู้ใช้สามารถเลือกได้ว่าต้องการโปรแกรมคำนวณหาพื้นที่ของวงกลม สามเหลี่ยม หรือสี่เหลี่ยม

```
from math import pi                                                          # บรรทัดที่ 1
                                                                             # บรรทัดที่ 2
shape = input (“ต้องการหาพื้นที่ของ วงกลม (c), สามเหลี่ยม (t), หรือ สี่เหลี่ยม (r) : “)   # บรรทัดที่ 3
if shape == ‘c’ :                                                            # บรรทัดที่ 4
        r =  float (input (‘ป้อนรัศมีของวงกลม: ‘)                            # บรรทัดที่ 5
        area = pi * r * r                                                    # บรรทัดที่ 6
elif shape == ‘t’:                                                           # บรรทัดที่ 7
        b = float (input (‘ป้อนความยาวฐาน: ’)                                # บรรทัดที่ 8
        h = float (input (‘ป้อนความสูง: ’)                                   # บรรทัดที่ 9
        area = (1/2) * b * h                                                 # บรรทัดที่ 10
```

```
else :                                                    # บรรทัดที่ 11
        w = float (input (‘ป้อนความกว้าง: ’)              # บรรทัดที่ 12
        l = float (input (‘ป้อนความยาว: ’)                # บรรทัดที่ 13
        area = w * l                                      # บรรทัดที่ 14
print (‘รูปดังกล่าวมีพื้นที่, area, ‘ตารางหน่วย’)          # บรรทัดที่ 15
```

ตัวอย่างที่ 2.6 อธิบายได้ดังนี้

1. บรรทัดที่ 3 รับข้อมูลเข้าจากผู้ใช้เป็นตัวอักษร เก็บไว้ในตัวแปร shape ที่ระบุทางเลือกของรูปหลายเหลี่ยมที่ต้องการให้หาพื้นที่
2. บรรทัดที่ 4, 7 และ 11 เป็นการเปรียบเทียบข้อมูลที่ผู้ใช้กรอกว่าตรงกับการหาพื้นที่วงกลม สามเหลี่ยม และสี่เหลี่ยม ตามลำดับ

กิจกรรมที่ 2.1

1. ปรับปรุงตัวอย่างที่ 2.6 ดังนี้
   - ✓ ให้ตรวจสอบว่า ถ้าผู้ใช้ป้อนตัวอักษรอื่นที่ไม่ใช่ ‘c’, ‘t’ หรือ ‘r’ ให้โปรแกรมแจ้งข้อความแสดงความผิดพลาดออกมา
   - ✓ ปรับการแสดงผลลัพธ์ในบรรทัดที่ 16 จาก “รูปดังกล่าวมีพื้นที่” เป็น “รูปวงกลมมีพื้นที่” “รูปสามเหลี่ยมมีพื้นที่” หรือ “รูปสี่เหลี่ยมมีพื้นที่” ให้ถูกต้องตรงตามตัวอักษรที่ป้อนในบรรทัดที่ 3
2. เขียนโปรแกรมไพทอนดังนี้
   - ✓ รับค่ารัศมีวงกลม หรือความยาวด้านสี่เหลี่ยม
   - ✓ ใช้เต่าไพทอนวาดรูป ให้มีขนาดตามค่าที่รับเข้ามา
3. เขียนโปรแกรมไพทอนเพื่อวิเคราะห์ผลคะแนนสอบวิชาหนึ่ง ที่มีช่วงคะแนนตั้งแต่ 0 ถึง 100 และให้แสดงผลการวิเคราะห์ตามช่วงคะแนนที่ได้ดังตารางต่อไปนี้ แต่ถ้ามีงานค้างส่ง จะได้ผลการวิเคราะห์เป็น “ร” โดยไม่ขึ้นกับคะแนนที่ได้

คะแนนที่ได้	ผลการวิเคราะห์
ตั้งแต่ 0 แต่ไม่ถึง 50 และไม่มีงานค้างส่ง	ไม่ผ่าน
ตั้งแต่ 50 แต่ไม่ถึง 70 และไม่มีงานค้างส่ง	ผ่าน
ตั้งแต่ 70 แต่ไม่ถึง 90 และไม่มีงานค้างส่ง	ดี
ตั้งแต่ 90 ถึง 100 และไม่มีงานค้างส่ง	เยี่ยม
มีงานค้างส่ง	ร

2.5 ฟังก์ชัน

ฟังก์ชัน (function) เป็นโปรแกรมย่อยที่เขียนขึ้นเพื่อให้ทำงานเฉพาะตามที่กำหนด ผู้เขียนโปรแกรมสามารถเรียกใช้ฟังก์ชันได้สะดวกโดยไม่ต้องเขียนชุดคำสั่งซ้ำอีก ทำให้การเขียนโปรแกรมขนาดใหญ่ทำได้รวดเร็วและตรวจสอบความถูกต้องของโปรแกรมได้ง่ายขึ้น

ไพทอนมีฟังก์ชันให้ใช้งานเป็นจำนวนมาก นักเรียนเคยใช้งานมาแล้วหลายคำสั่ง เช่น input (), print (), int (), float () และ type () การใช้งานฟังก์ชันทำได้โดยเรียกชื่อฟังก์ชัน พร้อมกับส่งค่าของข้อมูลตามจำนวนที่ฟังก์ชันกำหนด โดยระบุอยู่ภายในเครื่องหมาย ( ) ตามหลังชื่อฟังก์ชัน หากค่าของข้อมูลที่ส่งไปให้กับฟังก์ชันมีมากกว่าหนึ่งจำนวนจะคั่นด้วยเครื่องหมายจุลภาค (,) โดยจำนวนและชนิดข้อมูลของค่าที่จะส่งให้กับแต่ละฟังก์ชันจะต้องขึ้นอยู่กับฟังก์ชันนั้น ๆ ว่าถูกออกแบบไว้ให้รับค่าข้อมูลชนิดใด กี่จำนวน และเรียงลำดับกันอย่างไร เช่น

print (‘area =’, area) เป็นการเรียกฟังก์ชัน print () ที่ส่งค่าสตริง ‘area =’ และค่าของตัวแปร area เมื่อฟังก์ชัน print () ทำงาน ก็จะพิมพ์ค่าที่ส่งให้ออกมาทางจอภาพ ตามลำดับจากซ้ายไปขวา

int (‘20’) เป็นการเรียกฟังก์ชัน int () ที่ส่งค่าสตริงไปให้เพียงหนึ่งจำนวน คือ 20

นอกจากนี้ เรายังสามารถกำหนดให้ฟังก์ชันมีการคืนค่า (return) หรือส่งค่ากลับเมื่อฟังก์ชันทำงานเสร็จแล้ว ตัวอย่างเช่น การเรียกใช้ฟังก์ชัน int (‘20’) เป็นการส่งค่าสตริง ‘20’ ให้กับฟังก์ชัน int () เพื่อแปลงสตริงดังกล่าวให้เป็นค่าจำนวนเต็ม ซึ่งเมื่อฟังก์ชัน int () ทำงานเสร็จ จะคืนค่าจำนวนเต็ม 20 กลับมา เราจึงสามารถใช้ int (‘20’) ได้ในลักษณะเดียวกันกับค่าจำนวนเต็ม 20

ตัวอย่างที่ 2.7 การส่งค่าและรับคืนค่าจากฟังก์ชันจากตัวอย่างที่ 2.5

```
age = int (input (“อายุของท่าน คือ: “))                    # บรรทัดที่ 1
if age >= 60:                                               # บรรทัดที่ 2
        fare = 3.25                                         # บรรทัดที่ 3
elif ege <= 3:                                              # บรรทัดที่ 4
        fare = 0                                            # บรรทัดที่ 5
else:                                                       # บรรทัดที่ 6
        fare = 6.50                                         # บรรทัดที่ 7
print (“ค่าโดยสารของท่านคือ: ”, fare)                       # บรรทัดที่ 8
```

ตัวอย่างที่ 2.7 อธิบายการเรียกใช้ฟังก์ชันได้ดังนี้

1. บรรทัดที่ 1 มีการเรียกใช้ฟังก์ชัน int () และ input () ซ้อนกันอยู่ในกรณีที่มีฟังก์ชันซ้อนกัน ไพทอนจะเริ่มประมวลผลจากฟังก์ชันที่อยู่ในสุดก่อน โดยเริ่มต้นฟังก์ชัน input () ถูกเรียกใช้ก่อน และไพทอนจะส่งค่าสตริง “อายุของท่าน คือ” ไปให้กันฟังก์ชัน แล้วฟังก์ชัน input () จะแสดงสตริงนี้ออกทางจอภาพพร้อมกับรอรับการป้อนข้อมูลจากผู้ใช้ เมื่อผู้ใช้ป้อนข้อมูลแล้วกดแป้น Enter ฟังก์ชัน input () จะคืนค่าข้อมูลที่ผู้ใช้ป้อนมากเป็นข้อมูลชนิดสตริง เช่น ผู้ใช้ป้อนข้อความเป็น ‘20’ การเรียกใช้ฟังก์ชัน int (input (“อายุของท่านคือ: ”)) จะเปรียบเสมือนกับ int (‘20’)

2. จากนั้นไพทอนจึงประมวลผลฟังก์ชัน int (‘20’) ในทำนองเดียวกันก็คือ เป็นเรียกใช้ฟังก์ชัน int () โดยส่งค่าสตริง ‘20’ ไปให้ การทำงานของฟังก์ชัน int () ก็จะแปลงข้อมูลสตริงที่ถูกส่งมาให้เป็นค่าจำนวนเต็ม 20 แล้วคืนค่าจำนวนเต็มนี้กลับออกมา ดังนั้น การเรียกใช้คำสั่ง age = int (input (“อายุของท่านคือ: ”)) จึงเปรียบเสมือนกับ age = int (‘20’) นั่นคือ age = 20 นั่นเอง

3. บรรทัดที่ 8 มีการเรียกใช้ฟังก์ชัน print () โดยส่งข้อมูลจำนวน 2 ตัว คือ สตริง “ค่าโดยสารของท่าน คือ: “ และค่าตัวแปร fare การทำงานของฟังก์ชัน print () จะพิมพ์ค่าที่ส่งไปให้ออกทางจอภาพเรียงตามลำดับ

เกร็ดน่ารู้

ฟังก์ชันและการคิดเชิงนามธรรม

การใช้ฟังก์ชันเป็นรูปแบบหนึ่งของการซ่อนรายละเอียดในการคิดเชิงนามธรรม เนื่องจากการเรียกใช้ฟังก์ชัน เช่น print () หรือ input () ไม่จำเป็นต้องทราบรายละเอียดว่าภายในฟังก์ชันนั้นมีการทำงานอย่างไร แต่ทราบเพียงหน้าที่ของฟังก์ชัน ก็สามารถเรียกใช้งานได้

เกร็ดน่ารู้

อาร์กิวเมนต์

ในการเรียกใช้ฟังก์ชัน ค่าที่ส่งไปให้กับฟังก์ชัน จะเรียกว่า อาร์กิวเมนต์ (argument) โดยอาจอยู่ในรูปของค่าข้อมูล ตัวแปร นิพจน์ ข้อความหรืออาจเป็นฟังก์ชันก็ได้ ซึ่งจำนวนอาร์กิวเมนต์อาจมีได้หลายตัว ขึ้นอยู่กับแต่ละฟังก์ชัน ถ้ามีอาร์กิวเมนต์หลายตัว จะเขียนโดยใช้เครื่องหมายคอมม่า (,) คั่นระหว่างอาร์กิวเมนต์แต่ละตัว

นอกจากฟังก์ชันที่ไพทอนมีเตรียมไว้ให้ใช้งานแล้ว นักเรียนยังสามารถสร้างฟังก์ชันขึ้นใช้เองได้ โดยฟังก์ชันประกอบด้วย 3 ส่วนย่อย ได้แก่ (1) ชื่อฟังก์ชัน (2) พารามิเตอร์ (parameter) ภายในวงเล็บ ปิดท้ายด้วยเครื่องหมายทวิภาค (:) และ (3) ชุดคำสั่งของฟังก์ชัน

ตัวอย่างที่ 2.8 จะแสดงการสร้างฟังก์ชันขึ้นใช้เอง โดยเป็นฟังก์ชันแบบที่ยังไม่มีการคืนค่า

ตัวอย่างที่ 2.8 ฟังก์ชัน hello1 ()

พิมพ์คำสั่งต่อไปนี้ในคอลโซล

```
def hello1 (name):                                  # บรรทัดที่ 1
    print (‘สวัสดี’, name)                          # บรรทัดที่ 2
                                                    # บรรทัดที่ 3
hello1 (‘ประวิทย์’)                                 # บรรทัดที่ 4
```

ผลลัพธ์ที่ได้ คือ

```
สวัสดี ประวิทย์
```

ตัวอย่างที่ 2.8 อธิบายได้ดังนี้

1. บรรทัดที่ 1 เป็นการสร้างฟังก์ชันขึ้นใช้เอง เริ่มต้นจากคำหลักของไพทอน def จะใช้ระบุการเริ่มต้นนิยามฟังก์ชันใหม่ โดยมีชื่อฟังก์ชัน คือ hello1 มีพารามิเตอร์จำนวน 1 ตัว คือ ตัวแปร name เขียนอยู่

ภายในวงเล็บต่อท้ายชื่อฟังก์ชัน โดยต้องมีเครื่องหมายโคลอน (:) อยู่หลังวงเล็บปิด เมื่อมีการเรียกใช้ฟังก์ชัน คำสั่งภายในฟังก์ชันจะสามารถอ้างถึงค่าที่ส่งให้กับฟังก์ชันได้โดยผ่านตัวแปร name

2. บรรทัดที่ 2 เป็นชุดคำสั่งภายในฟังก์ชัน ซึ่งมีเพียงคำสั่งเดียว คือ คำสั่ง print ('สวัสดี', name) โดยจะพิมพ์ข้อความสวัสดีต่อด้วยค่าของตัวแปร name

3. บรรทัดที่ 4 เป็นการเรียกใช้ฟังก์ชัน hello1 () แล้วส่งอาร์กิวเมนต์สตริง 'ประวิทย์' เพียงค่าเดียว ชุดคำสั่งในฟังก์ชัน hello1 () ที่นิยามไว้ในบรรทัดที่ 1 ถึง 2 จะถูกเรียกให้ทำงาน และได้ผลลัพธ์เป็น "สวัสดี ประวิทย์"

รูปแบบการนิยามฟังก์ชันมีดังนี้

```
def ชื่อฟังก์ชัน (พารามิเตอร์_1, พารามิเตอร์_2, ... ) :
    ชุดคำสั่ง
    ...
```

การตั้งชื่อฟังก์ชัน เป็นเช่นเดียวกับหลักการตั้งชื่อตัวแปรของไพทอน โดยต้องมีวงเล็บเปิดและวงเล็บปิดต่อท้าย ภายในวงเล็บให้ระบุชื่อตัวแปรที่เป็นพารามิเตอร์ ถ้าไม่มีพารามิเตอร์ให้ปล่อยว่างไว้ แต่ถ้ามีมากกว่า 1 ตัว ให้คั่นระหว่างแต่ละตัวด้วยเครื่องหมายคอมม่า (,) แล้วปิดท้ายด้วยเครื่องหมายโคลอน (:) โดยชุดคำสั่งจะต้องย่อหน้าเยื้องเข้าไปจากคำสั่งหลัก def และถ้ามีมากกว่าหนึ่งคำสั่งจะต้องมีย่อหน้าตรงกันทุกคำสั่ง

เกร็ดน่ารู้

บล็อก

ชุดคำสั่ง ที่เรียกว่า บล็อก (block) จะอยู่บรรทัดถัดจากเครื่องหมายโคลอน (:) ใช้ร่วมกับการใช้ย่อหน้า (indentation) ของบล็อก และสิ้นสุดชุดคำสั่งเมื่อยุติการย่อหน้านั้น ในการเขียนโปรแกรมไพทอนนิยมย่อหน้าด้วยการเคาะแป้นเว้นวรรค 4 ครั้ง

ชวนคิด

จากโปรแกรมในตัวอย่างที่ 2.5 ให้ระบุว่า บล็อกของคำสั่งภายใต้ if, elif และ else คือ บรรทัดใดบ้าง

ตัวอย่างที่ 2.9 การนิยามฟังก์ชันที่มีพารามิเตอร์มากกว่าหนึ่งตัว

พิมพ์คำสั่งต่อไปนี้ในคอลโซล

```
def hello2 (fname, lname)                                    # บรรทัดที่ 1
        print (‘สวัสดี’, fname, lname)                       # บรรทัดที่ 2
                                                             # บรรทัดที่ 3
hello2 (‘ประวิทย์’, ‘โพธิ์ทอง’)                              # บรรทัดที่ 4
```

ผลลัพธ์ที่ได้ คือ

```
สวัสดี ประวิทย์ โพธิ์ทอง
```

ตัวอย่างที่ 2.9 บรรทัดที่ 1 ถึง 2 เป็นการสร้างฟังก์ชันขึ้นใช้เอง เริ่มต้นจากคำหลักของไพทอน def ตามด้วยชื่อฟังก์ชัน hello2 มีพารามิเตอร์จำนวน 2 ตัว กำหนดเป็นตัวแปรชื่อ fname และ lname เขียนอยู่ในวงเล็บ

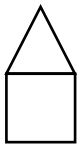
ตัวอย่างที่ 2.10 ฟังก์ชันวาดรูปบ้าน

จากตัวอย่างที่ 1.1 ในบทที่ 1 โปรแกรมวาดรูปบ้านสามารถนำมาเขียนเป็นฟังก์ชันในไพทอนได้ ให้พิมพ์คำสั่งต่อไปนี้ในโหมดสคริปต์ แล้วทดลองรันโปรแกรมเพื่อตรวจสอบผลลัพธ์

```
from turtle import *                                         # บรรทัดที่ 1
                                                             # บรรทัดที่ 2
def draw_square (size) :                                     # บรรทัดที่ 3
        pendown ()                                           # บรรทัดที่ 4
        for I in range (4) :                                 # บรรทัดที่ 5
                forward (size)                               # บรรทัดที่ 6
                left (90)                                    # บรรทัดที่ 7
        penup ()                                             # บรรทัดที่ 8
                                                             # บรรทัดที่ 9
def draw_triangle (size) :                                   # บรรทัดที่ 10
        pendown ()                                           # บรรทัดที่ 11
        for I in range (3) :                                 # บรรทัดที่ 12
                forward (size)                               # บรรทัดที่ 13
                left (120)                                   # บรรทัดที่ 14
        penup ()                                             # บรรทัดที่ 15
                                                             # บรรทัดที่ 16
def draw_house (size) :                                      # บรรทัดที่ 17
        draw_square (size)                                   # บรรทัดที่ 18
        left (90)                                            # บรรทัดที่ 19
        forward (size)                                       # บรรทัดที่ 20
        right (90)                                           # บรรทัดที่ 21
        draw_triangle (size)                                 # บรรทัดที่ 22
                                                             # บรรทัดที่ 23
```

```
shape ("turtle")                    # บรรทัดที่ 24
pensize (5)                         # บรรทัดที่ 25
speed (5)                           # บรรทัดที่ 26
draw_house (50)                     # บรรทัดที่ 27
hideturtle ()                       # บรรทัดที่ 28
exitonclick ()                      # บรรทัดที่ 29
```

ผลลัพธ์ที่ได้ คือ



ตัวอย่างที่ 2.10 อธิบายได้ดังนี้

1. บรรทัดที่ 1 เป็นการเรียกใช้งานโมดูล turtle เพื่อให้เรียกใช้ฟังก์ชันต่าง ๆ เกี่ยวกับการวาดรูปด้วยเต่าได้
2. บรรทัดที่ 3 ถึง 8 เป็นการสร้างฟังก์ชันวาดรูปสี่เหลี่ยมจัตุรัส โดยมีพารามิเตอร์จำนวน 1 ตัว กำหนดเป็นตัวแปรชื่อ size เพื่อระบุขนาดด้านของสี่เหลี่ยมจัตุรัส
3. บรรทัดที่ 10 ถึง 15 เป็นการสร้างฟังก์ชันวาดรูปสามเหลี่ยมด้านเท่า โดยมีพารามิเตอร์จำนวน 1 ตัว กำหนดเป็นตัวแปรชื่อ size เพื่อระบุขนาดด้านของสามาเหลี่ยม
4. บรรทัดที่ 17 ถึง 22 เป็นการสร้างฟังก์ชันวาดรูปบ้าน ที่มีการเรียกใช้ฟังก์ชันวาดรูปสี่เหลี่ยมจัตุรัส และฟังก์ชันวาดรูปสามเหลี่ยมด้านเท่า
5. บรรทัดที่ 24 ถึง 29 เป็นโปรแกรมหลัก โดยบรรทัดที่ 24 กำหนดให้มองเห็นปากกาเป็นรูปเต่า บรรทัดที่ 25 กำหนดขนาดความหนาของเส้นที่จะวาด บรรทัดที่ 26 กำหนดความเร็วในการเดินของเต่า บรรทัดที่ 27 เรียกฟังก์ชันที่จะสร้างบ้าน ขนาด 50 บรรทัดที่ 28 กำหนดให้ซ่อนรูปเต่าเมื่อวาดเสร็จ และบรรทัดที่ 29 เป็นคำสั่งกำหนดให้คงหน้าต่างที่วาดรูปเสร็จแล้วไว้ จนกว่าผู้ใช้จะคลิกที่ตำแหน่งใดๆ ในหน้าต่างจึงจะปิดหน้าต่างที่วาดรูป

การคืนค่าจากฟังก์ชัน

การสร้างฟังก์ชันขึ้นใช้งานเองนั้น หากต้องการคืนค่าให้กับโปรแกรมหลักเมื่อฟังก์ชันทำงานเสร็จ ให้ใช้คำสั่ง return ตามด้วยนิพจน์หรือค่าที่ต้องการคืน ในกรณีที่ไม่มีการคืนค่าก็ไม่ต้องใช้คำสั่ง return ดังเช่นในตัวอย่างที่ 2.8, 2.9 และ 2.10

ตัวอย่างที่ 2.11 การคืนค่าจากฟังก์ชัน

โปรแกรมไพทอนต่อไปนี้ ใช้ฟังก์ชันในการับค่ารัศมีของทรงกลม แล้วคืนค่าปริมาตรของทรงกลมดังกล่าว ให้พิมพ์คำสั่งต่อไปนี้ในโหมดสคริปต์ แล้วทดลองรันโปรแกรมเพื่อตรวจสอบผลลัพธ์

```
from math import pi                                          # บรรทัดที่ 1
                                                             # บรรทัดที่ 2
def sphere_volume (r) :                                      # บรรทัดที่ 3
        vol = (4/3) * pi * r * r * r                         # บรรทัดที่ 4
        return vol                                           # บรรทัดที่ 5
                                                             # บรรทัดที่ 6
radius = float (input ('Enter a radius: '))                  # บรรทัดที่ 7
print ('Volume is', sphere  volume (radius))                 # บรรทัดที่ 8
```

ตัวอย่างที่ 2.11 อธิบายได้ดังนี้

1. บรรทัดที่ 3 เริ่มการนิยามฟังก์ชันชื่อ sphere_volume () ที่มีพารามิเตอร์ 1 ตัว คือ รัศมีของทรงกลมเก็บไว้ในตัวแปร r
2. บรรทัดที่ 4 และ 5 เป็นชุดคำสั่งของฟังก์ชัน บรรทัดที่ 4 ทำหน้าที่คำนวณปริมาตรทรงกลมเก็บไว้ในตัวแปร vol และบรรทัดที่ 5 คืนค่าปริมาตรที่คำนวณได้กลับไปยังโปรแกรมที่เรียกฟังก์ชัน
3. บรรทัดที่ 7 เป็นต้นไป เป็นส่วนของโปรแกรมหลัก ที่จะมีการเรียกฟังก์ชัน sphere_volume () โดยบรรทัดที่ 7 รับค่ารัศมีของทรงกลมที่ผู้ใช้ป้อนเข้าไปเก็บไว้ในตัวแปร radius
4. บรรทัดที่ 8 เป็นการเรียกใช้ฟังก์ชัน sphere_volume () โดยส่งค่า radius เป็นอาร์กิวเมนต์ เมื่อฟังก์ชัน sphere_volume () ทำงานเสร็จจะคืนค่ากลับมา จากนั้นสตริง 'volume is' และค่าที่คืนกลับมาจะถูกส่งเป็นอาร์กิวเมนต์ไปให้กับฟังก์ชัน print () ทำงานต่อไป

เกร็ดน่ารู้

การคืนค่าเป็นนิพจน์

ฟังก์ชัน sphere_volume () ในตัวอย่างที่ 2.11 สามารถเขียนให้สั้นลงได้ เนื่องจากการคืนค่าด้วยคำสั่ง return สามารถคืนค่านิพจน์ได้ ดังนั้น sphere_volume () อาจเขียนได้เป็น

```
def sphere_volume (r) :
        return (4/3) * pi * r * r * r
```

กิจกรรมที่ 2.1

1. ใช้เต่าไพทอน วาดรูปสัญลักษณ์โอลิมปิก ซึ่งเป็นวงกลมห้าสีซ้อนกัน ดังรูป โดยให้สร้างฟังก์ชัน myCircle () ขึ้นเพื่อวาดรูปวงกลม

2. เขียนโปรแกรมพิมพ์รูปดาวด้วยเต่าไพทอน โดยสร้างฟังก์ชัน star (x) ที่รับค่าพารามิเตอร์หนึ่งตัว คือ x ที่ระบุขนาดของดาวแล้ว เขียนโปรแกรมเพื่อเรียกใช้ฟังก์ชันนี้วาดรูปหมู่ดาวเล็กใหญ่หลายดวง

สรุปท้ายบท

การเขียนโปรแกรมด้วยไพทอนมีคำสั่ง if, if – else สำหรับการทำงานแบบมีทางเลือก นอกากนี้ยังมีคำสั่งสำหรับการทำงานที่มีหลายเงื่อนไข หรือ if เชิงซ้อน ได้แก่ คำสั่ง if – elif – else และยังมีตัวดำเนินการบูลีนที่ใช้ในนิพจน์เปรียบเทียบสำหรับกำหนดเงื่อนไขที่ซับซ้อนขึ้น ได้แก่ and, or และ not และคำสั่งสำหรับการทำงานแบบวนซ้ำ ได้แก่ for ใช้สำหรับการทำงานที่ทราบจำนวนรอบ และ while ใช้ในกรณีที่ไม่ทราบจำนวนรอบที่แน่นอน

นอกจากนี้ผู้เขียนโปรแกรยังสามารถสร้างฟังก์ชัน และโปรแกรมย่อยให้ทำงานเฉพาะตามที่กำหนด เพื่อให้สามารถเรียกใช้งานได้โดยไม่ต้องเขียนชุดคำสั่งเดิมซ้ำอีก ทำให้สร้างโปรแกรมขนาดใหญ่ได้รวดเร็ว และตรวจสอบความถูกต้องของโปรแกรมได้ง่ายขึ้น

กิจกรรมท้ายบท

หมู่บ้านที่นักเรียนอาศัยอยู่ จัดงานฉลองวันปีใหม่ นักเรียนได้รับมอบหมายให้นับจำนวนกล่องสำหรับบรรจุจานที่ล้างแล้ว ซึ่งแต่ละกล่องสามารถบรรจุจานได้ 3 ตั้ง ตั้งละ 10 ใบ ให้นักเรียนแก้ปัญหาโดยดำเนินการตามขั้นตอนการแก้ปัญหาเพื่อคำนวณจำนวนกล่องที่น้อยที่สุด ที่ต้องเตรียมไว้สำหรับเก็บจานได้พอดี และไม่มีกล่องเหลือ

แบบฝึกหัดท้ายบท

ให้นักเรียนแก้ปัญหาโดยดำเนินการตามขั้นตอนการแก้ปัญหาเพื่อให้ได้โปรแกรมไพทอนสำหรับสถานการณ์ต่อไปนี้

1. นักเรียนหารายได้พิเศษด้วยการรับอาหารมาขาย โดยได้รับส่วนแบ่งยอดขายจากเจ้าของร้านดังนี้

รายการอาหาร	ราคาต่อหน่วย (บาท)	ส่วนแบ่งต่อหน่วย (บาท)	ส่วนแบ่งเพิ่มเติม
ขนมเค้ก	10	2	มากกว่า 20 ชิ้น เพิ่มเป็นชิ้นละ 3 บาท
ข้าวเหนียวไก่	20	5	มากกว่า 30 ห่อ เพิ่มเป็นห่อละ 7 บาท
น้ำส้ม	15	3	มากกว่า 30 ขวด เพิ่มเป็นขวดละ 5 บาท

ซึ่งหากนักเรียนทำยอดขายได้มากกว่า 500 บาท จะได้รับส่วนแบ่งจากยอดขายอีกร้อยละ 10 จากเงื่อนไขดังกล่าวให้เขียนโปรแกรม เพื่อสรุปยอดขายร่วมและส่วนแบ่งที่ได้รับ

2. ขึ้นรถอะไรดี

• นั่งรถโดยสารประจำทางปรับอากาศคิดอัตราค่าโดยสารคนละ 5 บาทต่อกิโลเมตร หากไม่ถึง 1 กิโลเมตรให้คิดราคา 5 บาท

• นั่งรถรับจ้างสาธารณะ ค่าโดยสารเริ่มต้น 35 บาทที่ระยะทาง 3 กิโลเมตรแรก ระยะทางที่เกินจากนั้นคิดอัตราค่าโดยสารเพิ่มขึ้น 2 บาทต่อกิโลเมตร โดยรถรับจ้างหนึ่งคันรับผู้โดยสารได้สูงสุด 4 คน

นักเรียนและกลุ่มเพื่อต้องการเดินทางไปยังสถานที่หนึ่ง นักเรียนจะเลือกเดินทางด้วยรถอะไรเพื่อให้ประหยัดค่าโดยสารที่สุด

3. ครูประจำชั้นมอบหมายให้นักเรียนบันทึกข้อมูลส่วนสูง น้ำหนัก เพศของเพื่อนและจัดทำรายงานสรุป ซึ่งประกอบด้วยข้อมูล ดังนี้

• ชื่อ นามสกุล เพศ ส่วนสูง น้ำหนัก

• ค่าเฉลี่ย ส่วนสูง น้ำหนัก ของทั้งห้อง และแยกตามเพศ

4. ครูจ้างป้าเชอรี่มาทำข้าวกะเพราไก่ไข่ดาวที่โรงเรียน จำนวน x กล่อง โดยป้าเชอรี่ต้องเตรียมวัตถุดิบมาเอง นักเรียนคิดว่าป้าเชอรี่ต้องใช้ต้นทุนในการทำข้าวกะเพราไก่ไข่ดาวทั้งสิ้นเท่าไร และจะคิดค่าจ้างในการทำกล่องละกี่บาท จึงจะได้กำไรร้อยละ 30

ข้าวกะเพราไก่ 15 กล่อง ใช้วัตถุดิบ ดังนี้

วัตถุดิบ/ ส่วนผสม	ราคาวัตถุดิบ
ข้าวหอมมะลิ 2 กิโลกรัม	38 บาท/ กิโลกรัม
เนื้อไก่ 1 กิโลกรัม	70 บาท/ กิโลกรัม
ใบกะเพรา 200 กรัม	20 บาท/ กิโลกรัม
พริก 200 กรัม	45 บาท/ กิโลกรัม
กระเทียม 20 กรัม	80 บาท/ กิโลกรัม
ไข่ไก่ 15 ฟอง	3.50 บาท/ ฟอง

หมายเหตุ เครื่องปรุงรส (ซอส ซีอิ้ว น้ำตาล) เตา น้ำ ไฟ และภาชนะในการทำอาหารโรงเรียนจัดเตรียมไว้ให้ โดยไม่คิดเป็นค่าใช้จ่าย