

บทที่ 1

ทฤษฎีที่เกี่ยวข้อง

บทนี้จะเป็นรายละเอียดเกี่ยวกับทฤษฎีและงานวิจัยที่เกี่ยวข้องกับการพัฒนาโปรแกรมในครั้งนี้ โดยที่แต่ละหัวข้อจะมีความสัมพันธ์กันเป็นลำดับ โดยหัวข้อที่หนึ่งจะแนะนำความรู้เรื่อง xxx เพื่อให้เข้าใจพื้นฐานเบื้องต้นเกี่ยวกับที่มาของโครงงาน หัวข้อที่สองที่สามจะช่วยเตรียมให้ผู้อ่านเข้าใจเทคโนโลยีที่ใช้ในการออกแบบและพัฒนา ส่วน ...

1.1 ความรู้พื้นฐาน

คำนำเรื่องความรู้พื้นฐาน

1.2 การเขียนคำสั่ง

คำนำเรื่องการเขียนคำสั่ง

1.2.1 การเขียนคำสั่งโดยแทรกในเอกสาร

```
1 import numpy as np
2
3 def gaussian_elimination(A, b):
4     # 1. ขั้นตอนการตัดทอน - Elimination Step
5     n = len(b)
6     for j in range(0, n-1):
7         # 1.1 เลือกสมการตั้งต้น - pivot equation สมการที่ j
8         # 1.2 ลดรูปสมการที่ i โดยที่ i เริ่มจาก j+1
9         for i in range(j+1, n):
10            # 1.2.1 คำนวณหาค่าตัวคูณ lam
11            lam = A[i, j]/A[j, j]
12            # 1.2.2 ลบออกจากด้านซ้ายของสมการ
13            A[i, j:n] = A[i, j:n] - lam*A[j, j:n]
14            # 1.2.3 ลบออกจากด้านขวาของสมการ
```

```

15         b[i] = b[i] - lam*b[j]
16     # 2. แทนค่าของตัวแปรที่หาค่าได้จากสมการล่างสุด แทนไปสมการบนไล่ไปเรื่อย
17     #     ขั้นตอนนี้ทั้งหมดเรียกว่า back substitution
18     x = b.copy()
19     for k in range(n-1, -1, -1):
20         x[k] = (b[k] - np.dot(A[k,k+1:n], x[k+1:n]))/A[k,k]
21
22     # 3. แสดงผลลัพธ์
23     #     ค่าของตัวแปร x ทุกตัว
24     print(x)
25
26 A = np.array([
27     [4, -2, 1],
28     [-2, 4, -2],
29     [1, -2, 4]
30 ], float)
31 b = np.array([11,-16,17], float)
32 gaussian_elimination(A, b)

```

ตัวอย่างที่ไม่

```

import numpy as np

def gaussian_elimination(A, b):
    # 1. ขั้นตอนการตัดทอน - Elimination Step
    n = len(b)

    for j in range(0, n-1):
        # 1.1 เลือกสมการตั้งต้น - pivot equation สมการที่ j
        # 1.2 ลดรูปสมการที่ i โดยที่ i เริ่มจาก j+1

        for i in range(j+1, n):
            # 1.2.1 คำนวณหาค่าตัวคูณ lam

```

```

    lam = A[i, j] / A[j, j]
    # 1.2.2 ลบออกจากด้านซ้ายของสมการ
    A[i, j:n] = A[i, j:n] - lam * A[j, j:n]
    # 1.2.3 ลบออกจากด้านขวาของสมการ
    b[i] = b[i] - lam * b[j]

# 2. แทนค่าของตัวแปรที่หาค่าได้จากสมการล่างสุด แทนไปสมการบนไล่ไปเรื่อย
#   ขั้นตอนนี้ทั้งหมดเรียกว่า back substitution
x = b.copy()
for k in range(n-1, -1, -1):
    x[k] = (b[k] - np.dot(A[k, k+1:n], x[k+1:n])) / A[k, k]

# 3. แสดงผลลัพธ์
#   ค่าของตัวแปร x ทุกตัว
print(x)

A = np.array([
    [4, -2, 1],
    [-2, 4, -2],
    [1, -2, 4]
], float)
b = np.array([11, -16, 17], float)
gaussian_elimination(A, b)

```