```python
# 12-1
print("stack")
n = 0
stack = []
while True:
    print("------------------")
    print("stack operation")
    print("option 1: Push")
    print("option 2: Pop")
    print("option 3: Display")
    print("option 4: Quit")
    print("------------------")

    choice = eval(input("Input your choice: "))
    if choice == 1:
        key = eval(input("Input your key: "))
        stack.append(key)
        n += 1
    elif choice == 2:
        if n > 0:
            key = stack.pop()
            print("Pop key: ", key)
            n -= 1
        else:
            print("Stack is empty")
    elif choice == 3:
        print("Stack: ", stack)
    else:
        break
```

```python
# 12-2
print("Queue")
n = 0
queue = []
while True:
    print("------------------")
    print("Queue operation")
    print("option 1: Enqueue")
    print("option 2: dequeue")
    print("option 3: Display")
    print("option 4: Quit")
    print("------------------")

    choice = eval(input("Input your choice: "))
    if choice == 1:
        key = eval(input("Input your key: "))
        queue.append(key)
        n += 1
    elif choice == 2:
        if n > 0:
            key = queue.pop(0)
            print("Pop key: ", key)
            n -= 1
        else:
            print("Queue is empty")
    elif choice == 3:
        print("Queue: ", queue)
    else:
        break
```

```python
# 12-3
a = [1, 2, 3, 4]
b = [2, 4, 3, 1]

c = [0 for i in range(4)]
for i in range(4):
    c[i] = a[i] + b[i]
print("向量加法")
print(c)

scalar = 3
for i in range(4):
    c[i] = scalar * a[i]
print("向量乘法")
print(c)

# 12-4
def Matrix_Add(A, B):  1 usage  new *
    n = len(A)
    c = [[0 for j in range(n)] for i in range(n)]
    for i in range(n):
        for j in range(n):
            c[i][j] = A[i][j] + B[i][j]
    return c
def Matrix_Multiply(A, B):  1 usage  new *
    n = len(A)
    c = [[0 for j in range(n)] for i in range(n)]
    for i in range(n):
        for j in range(n):
```

```python
def Matrix_Add(A, B):  1 usage  new *
    c = [[0 for j in range(n)] for i in range(n)]
    for i in range(n):
        for j in range(n):
            c[i][j] = A[i][j] + B[i][j]
    return c
def Matrix_Multiply(A, B):  1 usage  new *
    n = len(A)
    c = [[0 for j in range(n)] for i in range(n)]
    for i in range(n):
        for j in range(n):
            for k in range(n):
                c[i][j] = c[i][j] + A[i][k] * B[k][j]

    return c

A = [[1, 2], [3, 4]]
B = [[2, 4], [3, 1]]

c = Matrix_Add(A, B)
print("矩陣乘法")
print(c)

c = Matrix_Multiply(A, B)
print("矩陣乘法")
print(c)
```

```
/usr/local/bin/python3.12 /Users/pengyenjia/Desktop/運算思維與程式
stack
------------------
stack operation
option 1: Push
option 2: Pop
option 3: Display
option 4: Quit
------------------
Input your choice: 1
Input your key: 1
------------------
stack operation
option 1: Push
option 2: Pop
option 3: Display
option 4: Quit
------------------
Input your choice: 1
Input your key: 2
------------------
stack operation
option 1: Push
option 2: Pop
option 3: Display
option 4: Quit
------------------
Input your choice: 1
Input your key: 3
------------------
stack operation
option 1: Push
option 2: Pop
```

```
------------------
Input your choice: 2
Pop key:  3
------------------
stack operation
option 1: Push
option 2: Pop
option 3: Display
option 4: Quit
------------------
Input your choice: 3
Stack:  [1, 2]
------------------
stack operation
option 1: Push
option 2: Pop
option 3: Display
option 4: Quit
------------------
Input your choice: 4
Queue
------------------
Queue operation
option 1: Enqueue
option 2: dequeue
option 3: Display
option 4: Quit
------------------
Input your choice: 1
Input your key: 1
------------------
Queue operation
option 1: Enqueue
```

```
Input your choice: 1
Input your key: 1
------------------
Queue operation
option 1: Enqueue
option 2: dequeue
option 3: Display
option 4: Quit
------------------
Input your choice: 1
Input your key: 2
------------------
Queue operation
option 1: Enqueue
option 2: dequeue
option 3: Display
option 4: Quit
------------------
Input your choice: 1
Input your key: 3
------------------
Queue operation
option 1: Enqueue
option 2: dequeue
option 3: Display
option 4: Quit
------------------
Input your choice: 2
Pop key:  1
------------------
Queue operation
option 1: Enqueue
option 2: dequeue
```

```
option 3: Display
option 4: Quit
------------------
Input your choice: 2
Pop key:  1
------------------
Queue operation
option 1: Enqueue
option 2: dequeue
option 3: Display
option 4: Quit
------------------
Input your choice: 3
Queue:  [2, 3]
------------------
Queue operation
option 1: Enqueue
option 2: dequeue
option 3: Display
option 4: Quit
------------------
Input your choice: 4
向量加法
[3, 6, 6, 5]
向量乘法
[3, 6, 9, 12]
矩陣乘法
[[3, 6], [6, 5]]
矩陣乘法
[[8, 6], [18, 16]]

Process finished with exit code 0
```