1. 자연수 N을 입력 받아, N개의 1 이상 5 이하의 난수로 이루어진 리스트 a를 생성한다. 이렇게 생성된 리스트의 원소들을 3개씩 묶어서 합을 구한 다음 새로운 리스트 b에 삽입하는 함수 sumThree(a, n)을 사용하여 프로그램을 작성하라. 함수 sumThree(a, n)은 리스트 b를 반환한다.
   - 이 문제에서 순서도는 sumThree(a, n)에 대해서 그린다.
   - 만약 N이 3의 배수가 아닌 경우에는 다음과 같이 처리한다. N을 3으로 나눈 나머지가 1일 때는 리스트 a에 두 개의 0을 추가하고, N을 3으로 나눈 나머지가 2일 때는 리스트 a에 하나의 0을 추가하여 리스트 a의 길이를 3의 배수로 맞추어 준다.

```
>>>
N = 6
a = [4, 2, 4, 5, 3, 4]
b = [10, 12]
>>>
N = 7
a = [5, 1, 1, 1, 4, 1, 2]
b = [7, 6, 2]
>>>
N = 8
a = [2, 4, 1, 3, 2, 1, 3, 2]
b = [7, 6, 5]
>>>
```

```
def sumThree(a, n):
    while n % 3 != 0:
        a.append(0)
        n += 1
    b = []
    i = 0
    while i < n:
        s = a[i] + a[i+1] + a[i+2]
        b.append(s)
        i += 3
    return b


import random
N = int(input('N = '))
a = []
for i in range(N):
    a.append(random.randint(1, 5))
print('a =', a)
print('b =', sumThree(a, N))
```

2. 중복되는 데이터가 없이 정수를 입력 받아서 리스트로 만든 다음, 리스트의 원소 중 비소수의 합 p_sum과 소수의 합 n_sum을 출력하는 함수 sumPrime(a)를 사용하여 프로그램을 작성하라. 함수 sumPrime(a)는 p_sum과 n_sum을 반환한다.
   - 이 문제에서 순서도는 sumPrime(a)에 대해서 그린다.
   - 프로그램을 실행할 때 2보다 작은 수를 입력하면, '2 이상의 수만 입력하세요.'라는 메시지를 출력한다.

```
>>>
정수 입력(종료시는 999) : 1
2 이상의 수만 입력하세요.
정수 입력(종료시는 999) : 3
정수 입력(종료시는 999) : 2
정수 입력(종료시는 999) : 4
정수 입력(종료시는 999) : 5
정수 입력(종료시는 999) : 6
정수 입력(종료시는 999) : 1
2 이상의 수만 입력하세요.
정수 입력(종료시는 999) : 3
정수 입력(종료시는 999) : 4
정수 입력(종료시는 999) : 7
정수 입력(종료시는 999) : 999
생성된 리스트 : [3, 2, 4, 5, 6, 7]
소수의 합 :  17
비소수의 합 :  10
>>>
```

소수일 때 True를 반환하는 함수 isPrime(x)는 다음과 같다.

```
def isPrime(x):
    for i in range(2, int(x/2)+1):
        if x % i == 0:
            return False
    return True
```

```
def isPrime(x):
    for i in range(2, int(x/2)+1):
        if x % i == 0:
            return False
    return True

def sumPrime(a):
    n = len(a)
    p_sum = 0
    n_sum = 0
    for i in range(n):
        if isPrime(a[i]):
            p_sum += a[i]
        else:
            n_sum += a[i]
    return (p_sum, n_sum)


data = int(input('정수 입력(종료시는 999) : '))
nums = []
while data != 999:
    if data <= 1:
        print('2 이상의 수만 입력하세요.')
    elif nums.count(data) == 0:
        nums.append(data)
    data = int(input('정수 입력(종료시는 999) : '))
print('생성된 리스트 :', nums)
result = sumPrime(nums)
print('소수의 합 : ', result[0])
print('비소수의 합 : ', result[1])
```

3. 다음과 같이 0 이상 3 이하의 정수로 이루어진 정방행렬에서 i를 행, j를 열이라고 할 때, 상위 삼각 행렬(i < j인 원소)에 있는 원소의 합 s1과 하위 삼각 행렬(i > j인 원소)에 있는 원소의 합 s2를 구한 다음, s1에서 s2를 뺀 결과를 출력하는 함수 subMatrix(a)를 사용하여 프로그램을 작성하라. 함수 subMatrix(a)는 s1과 s2를 반환한다.
   - 이 문제에서 순서도는 subMatrix(a)에 대해 그린다.
   - 사용자로부터 정방행렬의 크기 M을 입력 받고, 0이상 3 이하의 정수는 난수를 발생시켜서 입력한다. 주대각선(i = j)에 있는 원소에는 0을 입력한다.

```
>>>
M = 4

0 1 0 1
3 0 2 1
1 3 0 0
1 0 2 0

결과 : 5 - 10 = -5
>>>
```

행렬을 만들어 주는 함수 makeMatrix(m)와 행렬을 출력해 주는 함수 printMatrix(a)는 다음과 같다.

```
def makeMatrix(m):
    a = []
    for i in range(m):
        b = []
        for j in range(m):
            if i == j:
                b.append(0)
            else:
                b.append(random.randint(0, 3))
        a.append(b)
    return a
```

```
def printMatix(a):
    n = len(a)
    for i in range(n):
        for j in range(n):
            print(a[i][j], end=' ')
        print()
```

```
def makeMatrix(m):
    a = []
    for i in range(m):
        b = []
        for j in range(m):
            if i == j:
                b.append(0)
            else:
                b.append(random.randint(0, 3))
        a.append(b)
    return a

def printMatix(a):
    n = len(a)
    for i in range(n):
        for j in range(n):
            print(a[i][j], end=' ')
        print()

def subMatrix(a):
    s1 = 0
    s2 = 0
    n = len(a)
    for i in range(n):
        for j in range(n):
            if i < j:
                s1 += a[i][j]
            elif i > j:
                s2 += a[i][j]
    return (s1, s2)

import random
M = int(input('M = '))
A = makeMatrix(M)
print()
printMatix(A)
print()
```

```
result = subMatrix(A)
print('결과 : %d - %d = %d'%(result[0], result[1], result[0] - result[1]))
```

---

4. 영문 소문자 a를 0, b는 1, c는 2, …, i는 8, j는 9와 같이 숫자를 나타낸다고 하자. 다음과 같이 두 개의 문자열 S1과 S2를 입력 받은 다음, 입력된 문자열을 정수로 만드는 함수 makeNumber(s)를 사용하여 S1과 S2의 합을 출력하는 프로그램을 작성하라.
   - 이 문제에서 순서도는 makeNumber(s)에 대해 그린다.
   - a부터 j 사이의 문자가 아닌 경우 'a부터 j 사이의 영문 소문자만 입력 가능합니다.'라는 오류 메시지를 출력한다.

```
>>>
S1 = xyz
a 부터 j 사이의 영문 소문자만 입력 가능합니다.
S1 = bca
S2 = efg
120 + 456 = 576
>
```

---

a부터 j 사이의 영문 소문자인 경우 True를 반환하는 함수 isAtoj(s)는 다음과 같다.

```
def isAtoj(s):
    for i in range(len(s)):
        t = ord(s[i])
        if t < 97 or t > 106:
            return False
    return True
```

```
def isAtoj(s):
    for i in range(len(s)):
        t = ord(s[i])
        if t < 97 or t > 106:
            return False
    return True

def makeNumber(s):
    k = 1
    a = 0
    for i in range(len(s)-1, -1, -1):
        t = ord(s[i]) - 97
        a += t * k
        k *= 10
    return a

S1 = input('S1 = ')
while not isAtoj(S1):
    print('a 부터 j 사이의 영문 소문자만 입력 가능합니다.')
    S1 = input('S1 = ')
S2 = input('S2 = ')
while not isAtoj(S2):
    print('a 부터 j 사이의 영문 소문자만 입력 가능합니다.')
    S2 = input('S2 = ')
s1 = makeNumber(S1)
s2 = makeNumber(S2)

print('%d + %d = %d'%(s1, s2, s1+s2))
```