

2장 프로그래밍 실습_파이썬 코드

[프로그래밍 실습 1]

```
def main() :
    a = int(input("자연수 a를 입력하라.: ")) # 사용자로부터 자연수를 입력받음
    print("자연수 a의 약수들은 다음과 같다.")

    # 해당 for 문은 a의 약수들을 찾아서 출력
    for i in range(1, a + 1):
        # a를 i로 나눌 때 나머지가 0이면 i가 a의 약수임
        if a % i == 0:
            print(i, end=" ")
    print()

if __name__ == "__main__":
    main()
```

[프로그래밍 실습 2]

```
import math # ceil과 sqrt 함수를 포함한 패키지
# isPrime 함수가 주어진 number가 소수인지 확인
def isPrime(number) -> bool:

    if number < 2: # 정수 1은 소수도 아니고 합성수도 아님
        return False
    if number == 2: # 정수 2는 소수
        return True
    if number % 2 == 0: # 정수 2를 제외한 모든 짝수가 소수가 아님
        return False
    # 3부터 sqrt(number)까지의 정수들로 number를 나누면서 소수인지 확인
    for i in range(3, math.ceil((math.sqrt(number))), 2):
        if number % i == 0: # number의 약수가 있으면 number는 소수가 아님
            return False
    return True # number는 소수

def main():
    number = int(input("정수를 입력하라. : ")) # 사용자로부터 정수를 입력받음
    # isPrime 함수를 통해 number가 소수인지 아닌지 결정
    if isPrime(number):
        print(number, "는 소수이다.")
    else:
        print(number, "는 소수가 아니다.")

if __name__ == "__main__":
    main()
```

[프로그래밍 실습 3]

```
def main():
    # 사용자로부터 정수를 입력받음
    number = int(input("어떤 정수까지의 소수들을 알고 싶은가? "))
    # 소수값들을 저장하기 위한 배열 정의
    array = list(range(2, number + 1))

    for i in range(2, number):
        if array[i - 2]:
            if array[i - 2] ** 2 > number:
                break
            else:
                factor = array[i - 2]
                # factor의 배수들을 array에서 제거
                while factor * i <= number:
                    if array[(factor * i) - 2]:
                        array[(factor * i) - 2] = 0
                    factor += 1

    # 결과 출력
    print(number, "까지의 소수들은 다음과 같다.: ", end = "")

    for i in range(2, number):
        if array[i - 2]:
            print(array[i - 2], " ", end = "")
    print()

if __name__ == "__main__":
    main()
```

[프로그래밍 실습 4]

```
# matrixout 함수가 정방행렬 mx를 출력
def matrixout(mx, size):
    print("\u250c          \u2510")
    for i in range(0, size):
        print("| ", end = "")
        for j in range(0, size):
            print("%2d" % mx[i][j], end = " ")
        print(" |")
    print("\u2514          \u2518")
a = [[2,0,3],[2,5,3],[1,0,8]]      # 정방행렬 A
b = [[2,4,3],[2,5,6],[1,0,5]]      # 정방행렬 B
c = [[0,0,0],[0,0,0],[0,0,0]]      # 정방행렬 C
d = [[0,0,0],[0,0,0],[0,0,0]]      # 정방행렬 D
print("행렬 A : ")
matrixout(a, 3)                    # 정방행렬 A를 출력
```

```

print()
print("행렬 B : ")          # 정방행렬 B를 출력
matrixout(b, 3)
print()
print("A + B: ")
# 정방행렬 A와 B의 합을 행렬 C에 저장하고 C를 출력
for i in range(0,3):
    for j in range(0,3):
        c[i][j] = a[i][j] + b[i][j]
matrixout(c, 3)

# 정방행렬 A와 B의 곱을 계산해서 행렬 D에 대입
for i in range(0,3):
    for j in range(0,3):
        for k in range(0,3):
            d[i][j] += a[i][k] * b[k][j];
print()
print("A x B: ")
matrixout(d, 3)          # 정방행렬 D를 출력

```

[프로그래밍 실습 5]

```

from __future__ import print_function
from sys import stdin

def printf(str, *args):
    print(str % args, end='')

# matrixout 함수가 정방행렬 mx를 출력
def matrixout(mx, size):
    print("\u250c          \u2510")
    for i in range(0, size):
        print("| ", end = "")
        for j in range(0, size):
            print("%3d" % mx[i][j], end = " ")
        print(" |")
    print("\u2514          \u2518")

arr_1 = [[1,2,3,4],[4,3,2,1],[2,5,7,9],[6,3,2,1]] # 정방행렬 arr_1
arr_2 = [[1,5,6,7],[8,3,1,7],[6,2,8,3],[9,2,1,2]] # 정방행렬 arr_2

# 정방행렬 arr_1을 출력
printf("행렬 A : \n")
matrixout(arr_1, 4)

printf("\n\n")

# 정방행렬 arr_2를 출력
printf("행렬 B : \n")

```

```

matrixout(arr_2, 4)

printf("\n\n")

# 정방행렬 res를 0으로 초기화
res = [[0 for _ in range(4)] for _ in range(4)]

# 삼중 for 문은 (i, j, k 순으로) 정방행렬 arr_1과 arr_2를 곱함
printf("A x B (i,j,k 순으로) : \n")
for i in range(0, 4):
    for j in range(0, 4):
        for k in range(0, 4):
            res[i][j] += arr_1[i][k] * arr_2[k][j]

# 정방행렬 res를 출력
matrixout(res, 4)

# 정방행렬 res를 0으로 재초기화
res = [[0 for _ in range(4)] for _ in range(4)]

# 삼중 for 문은 (i, k, j 순으로) 정방행렬 arr_1과 arr_2를 곱함
printf("\n\nA x B (i,k,j 순으로) : \n")
for i in range(0, 4):
    for k in range(0, 4):
        for j in range(0, 4):
            res[i][j] += arr_1[i][k] * arr_2[k][j]

# 정방행렬 res를 출력
matrixout(res, 4)

printf("\n\n결과는 같다. \n")
stdin.readline()

```

[프로그래밍 실습 6]

```

from __future__ import print_function
from sys import stdin

def printf(str, *args):
    print(str % args, end='')

# print_matrix 함수가 부울행렬 x를 출력
def print_matrix(x):
    print("\u250c      \u2510")
    for i in range(0, maxvalue):
        print("|", end = "")
        for j in range(0, maxvalue):
            printf("%2d" % x[i][j])

```

```

        print(" |",end = "")
        printf("\n")
    print("\u2514      \u2518")

# mat_compute 함수가 a와 b에 대해 접합, 교합, 부울곱 계산
def mat_compute(a, b):
# 접합, 교합, 부울곱 계산을 위해 삼중 for 문 사용
    for i in range(0, maxvalue):
        for j in range(0, maxvalue):
            for k in range(0, maxvalue):
                c[i][j] = a[i][j] and b[i][j]      # 접합
                d[i][j] = a[i][j] or b[i][j]        # 교합
                # 부울곱
                if k == 0:
                    e[i][j] = a[i][k] * b[k][j]
                else:
                    e[i][j] = e[i][j] or (a[i][k] and b[k][j])

maxvalue = 3      # 정방행렬의 크기를 위한 변수

# mat_compute 함수에서 보조 공간으로 쓰일 부울행렬 정의 및 초기화
c = [[False for _ in range(maxvalue)] for _ in range(maxvalue)]
d = [[False for _ in range(maxvalue)] for _ in range(maxvalue)]
e = [[False for _ in range(maxvalue)] for _ in range(maxvalue)]

# 사용자가 입력할 부울행렬 2개 정의
a = [[False for _ in range(maxvalue)] for _ in range(maxvalue)]
b = [[False for _ in range(maxvalue)] for _ in range(maxvalue)]

# 사용자를 위한 안내 출력
printf("\n")
printf("3x3인 부울행렬 A를 입력하라(0과 1만 사용).\n")
printf("\n[예]\t0 1 0   \n\t0 0 0   \n\t1 0 1\n\n")

list = []      # 사용자가 입력할 부울행렬을 문자열로 받아주는 list 정의

# 해당 while 문을 통해 사용자로부터 부울행렬을 줄별로 입력
while len(list) < maxvalue * maxvalue:
    list += stdin.readline().strip("\n").split()

# list에 있는 값들을 a에 대입
for i in range(0, maxvalue):
    for j in range(0, maxvalue):
        a[i][j] = bool(int(list[i*maxvalue + j]))

printf("\n3x3인 부울행렬 B를 입력하라(0과 1만 사용).\n")

list = []      # 사용자가 입력할 부울행렬을 문자열로 받아주는 list 재정의

```

```

# 해당 while 문을 통해 사용자로부터 부울행렬을 줄별로 입력
while len(list) < maxvalue * maxvalue:
    list += stdin.readline().strip("\n").split()

# list에 있는 값들을 b에 대입
for i in range(0, maxvalue):
    for j in range(0, maxvalue):
        b[i][j] = bool(int(list[i*maxvalue + j]))

# 부울행렬 a와 b에 대해 집합, 교합, 부울곱을 계산하는 함수 호출
mat_compute(a,b)
printf("\n")

printf("A \u2227 B (집합)\n")
print_matrix(c)          # 집합 결과를 출력
printf("\n")

printf("A \u2228 B (교합)\n")
print_matrix(d)          # 교합 결과를 출력
printf("\n")

printf("A \u2299 B (부울곱)\n")
print_matrix(e)          # 부울곱 결과를 출력

stdin.readline()

```

[프로그래밍 실습 7]

```

from __future__ import print_function
from sys import stdin

def printf(str, *args):
    print(str % args, end='')

# matrixout 함수가 정방행렬 mx를 출력
def matrixout(mx, size):
    blanks = size * 7 + 2

    print("\u250c", end = "")
    for i in range(0,blanks):
        print(" ", end = "")
    print("\u2510")

    for i in range(0, size):
        print("| ", end = "")
        for j in range(0, size):
            print("% 1.3f" % mx[i][j], end = " ")
        print(" |")

```

```

print("\u2514", end = "")
for i in range(0,blanks):
    print(" ", end = "")
print("\u2518")

# transposeMatrix 함수가 정방행렬 m의 전치행렬을 반환
def transposeMatrix(m):
    result = [[m[j][i] for j in range (len (m))] for i in range (len (m[0]))]
    return result

# getMatrixMinor 함수가 성분 (i,j)에 대한 소행렬식을 반환
def getMatrixMinor(m,i,j):
    minor_matrix = [row[j+1:] + row[:j] for row in (m[:i]+m[i+1:])]
    return minor_matrix

# getMatrixDeterminant 재귀함수가 m에 대한 행렬식을 반환
def getMatrixDeterminant(m):
    # 2x2 행렬일 경우(기초 단계) :
    if len(m) == 2:
        return m[0][0]*m[1][1]-m[0][1]*m[1][0]
    determinant = 0
    for c in range(len(m)):
        determinant += ((-1)**c)*m[0][c]*getMatrixDeterminant(getMatrixMinor(m,0,c))
    return determinant

# is_zero 함수가 행렬 m의 행렬식이 0인지 아닌지 알려줌
def is_zero(m, k):
    determinant = getMatrixDeterminant(m) # 행렬식을 계산하는 함수 호출
    if(determinant==0): # 행렬식이 0일 때 안내를 출력하고 프로그램을 멈춤
        print('오류! 해당 행렬의 행렬식은 0이다.')
        print('역행렬 계산은 불가능하다!')
        return 0

# getMatrixInverse 함수가 정방행렬 m의 역행렬을 반환
def getMatrixInverse(m):
    determinant = getMatrixDeterminant(m) # 행렬식을 계산하는 함수 호출
    # 차수가 2인 정방행렬의 역행렬을 계산해서 반환
    if len(m) == 2:
        return [[m[1][1]/determinant, -1*m[0][1]/determinant],
                [-1*m[1][0]/determinant, m[0][0]/determinant]]

    # m의 각 성분에 대한 여인수를 계산
    cofactors = [] # 여인수들을 저장한 임시 행렬
    for r in range(len(m)):
        cofactorRow = []
        for c in range(len(m)):
            minor = getMatrixMinor(m,r,c)
            cofactorRow.append(((((-1)**(r+c)) * getMatrixDeterminant(minor)))
        cofactors.append(cofactorRow)

```

```

# 여인수들로 이루어진 cofactors 정방행렬의 전치행렬을 Disjoint에 대입
Disjoint = transposeMatrix(cofactors)

# Disjoint의 각 성분의 값을 행렬식 값으로 나눔
for r in range(len(Disjoint)):
    for c in range(len(Disjoint)):
        Disjoint[r][c] = Disjoint[r][c]/determinant

return Disjoint # 역행렬 반환

# 행렬의 차수에 입력 요구
k = int(input("정방행렬의 차수를 입력하라.: "))

# 차수가 k인 정방행렬을 정의하고 False 값들로 초기화
a = [[False for _ in range(k)] for _ in range(k)]

printf("\n")
printf("%dx%d인 정방행렬 A를 입력하라.\n", k , k)

list = [] # 사용자가 입력할 정방행렬을 문자열로 받아주는 list 정의

# 해당 while 문을 통해 사용자로부터 부울행렬을 줄별로 입력받음
while len(list) < k * k:
    list += stdin.readline().strip("\n").split()

# list에 있는 값들을 a에 대입
for i in range(0, k):
    for j in range(0, k):
        a[i][j] = int(list[i*k + j])

# 행렬 a의 행렬식이 0이 아니면 a에 대한 역행렬을 출력
if is_zero(a, k) != 0:
    print("\n해당 행렬의 역행렬은 다음과 같다.")
    matrixout(getMatrixInverse(a),k)
print()

```

[프로그래밍 실습 8]

```

# matrixout 함수가 차수가 size인 정방행렬 mx를 출력
def matrixout(mx, size, is_inverse):
    blanks = size * 7 + 2

    print("\u250c", end = "")
    for i in range(0,blanks):
        print(" ", end = "")
    print("\u2510")

    if(is_inverse == True): # 역행렬 부분만 출력

```



```

        for i in range(0, size):
            print("| ", end = "")
            for j in range(0, size):
                print("% 1.3f" % mx[i][j + 3], end = " ")
            print(" |")
    else:
        # 원본 행렬 부분만 출력
        for i in range(0, size):
            print("| ", end = "")
            for j in range(0, size):
                print("% 1.3f" % mx[i][j], end = " ")
            print(" |")

    print("\u2514", end = "")
    for i in range(0, blanks):
        print(" ", end = "")
    print("\u2518")

```

```

# 차수가 k인 정방행렬을 정의하고 정숫값으로 초기화
# 최종적으로 a를 단위행렬로 만들
a = [[2,0,3,1,0,0],[2,5,3,0,1,0],[1,0,8,0,0,1]]
n = 3 # 행렬의 차수

```

```

# 초기화된 행렬 a를 출력
print("행렬 A : \n")
matrixout(a, n, False)

```

```

# 역행렬을 반복문으로 계산
for i in range(n):
    if a[i][i] == 0.0:
        sys.exit('Divide by zero detected!')

    for j in range(n):
        if i != j:
            ratio = a[j][i]/a[i][i]
            for k in range(2*n):
                a[j][k] = a[j][k] - ratio * a[i][k]

```

```

# 주대각 성분을 1로 만들기 위한 행 연산
for i in range(n):
    divisor = a[i][i]
    for j in range(2*n):
        a[i][j] = a[i][j]/divisor

```

```

# 결과인 역행렬을 출력
print("\nA의 역행렬 : \n");
matrixout(a, n, True)

```

[프로그래밍 실습 9]

```
# 다음과 같은 연립일차방정식에 대해 확대행렬을 a로 정의하고 초기화
#  $2x_1 + 3x_2 + 5x_3 = 7$ 
#  $4x_1 + 5x_2 + 6x_3 = 6$ 
#  $8x_1 + 7x_2 + 8x_3 = 5$ 
a = [[2,3,5,7],[4,5,6,6],[8,7,8,5]]
x = ["x\u2081","x\u2082","x\u2083"]
n = 3 # 행렬의 차수

# 연립일차방정식을 출력
print("연립일차방정식 : ")
for i in range(0,3):
    print(' ', a[i][0],x[0], " + ", a[i][1], x[1], " + ", a[i][2], x[2], " = ", a[i][3], sep
    = '')

# 연립일차방정식을 행렬 형태로 출력
print()
print("연립일차방정식의 행렬 표현 : ")
print(" \u250c          \u2510")
for i in range(0,3):
    print(" |", end = "")
    for j in range(0,3):
        print("%3.f" % a[i][j], end = " ")
    print(" | ", a[i][3], " | ")
print(" \u2514          \u2518")

# 연립일차방정식을 반복문으로 계산
for i in range(0,3):
    if a[i][i] == 0: # 0으로 나누기 오류
        print("\nMathematical Error!\a\n\n")
        exit(0)
    for j in range(0,3):
        if i != j:
            ratio = a[j][i]/a[i][i]
            for k in range(0, 4):
                a[j][k] = a[j][k] - ratio*a[i][k]

for i in range(0,3):
    for j in range(3,4):
        a[i][j] = a[i][j] / a[i][i]

# 연립일차방정식의 해를 출력
print("\n 위의 연립일차방정식의 해:")
for i in range(0,3):
    for j in range(3,4):
        print(' ',x[i], ' = %3.3f ' % a[i][j], end = "", sep='')
    print()
print()
```