Open Source Practice HW #2

Department of Software
32153180
Lee Sang Min

Problem 1

A DNA sequence is a string made up of the letters A, T, G, and C. To find the complement of a DNA sequence, As are replaced by Ts by As, Gs by Cs, and Cs by Gs. For example, the complement of AATTGCCGT is TTAACGGCA.

1. Write a pseudo code in English of the algorithm that takes a DNA sequence and return its complement.
2. Test your algorithm

- ttcccatcaa gccctagggc tcctcgtggc tgctgggagt tgtagtctga acgcttctat
- cttggcgaga agcgcctacg ctccccctac cgagtcccgc ggtaattctt aaagcacctg
- caccgccccc ccgccgcctg cagagggcgc agcaggtctt gcacctcttc tgcatctcat
- tctccaggct tcagacctgt ctccctcatt caaaaaatat ttattatcga gctcttactt

1. Write a function named **complement** that takes a DNA sequence and returns the complement of it. Here, we can get some examples from the https://www.ncbi.nlm.nih.gov (https://www.ncbi.nlm.nih.gov). For example, **p53.rtf** is given.

pseudo code

```
Input:
  DNA sequence

Algorithm:
  Function Complement(string)
    for str in length(string)
      if str is 'T' then str is 'A'
      else if str is 'A' then str is 'T'
      else if str is 'G' then str is 'C'
      else if str is 'C' then str is 'G'
      else str is ' ' (this case is empty space)
    end

Output:
  complement of a DNA sequence
```

In [1]:

```
def complement(str):
    answer = []
    for i in range(len(str)):
        if str[i] == 'a':
            answer.append('t')
        elif str[i] == 't':
            answer.append('a')
        elif str[i] == 'c':
            answer.append('g')
        elif str[i] == 'g':
            answer.append('c')
        else:
            answer.append(' ')

    return "".join(answer)


DNA = input('DNA sequence : ')
print('The complement is :', complement(DNA))
```

```
DNA sequence : ttcccatcaa gccctagggc tcctcgtggc tgctgggagt tgtagtctga acgcttctat
The complement is : aagggtagtt cgggatcccg aggagcaccg acgaccctca acatcagact tgcgaag
ata
```

Problem 2

Develop a function that finds the minimum or maximum value in a list, depending on the caller’s request.

1. Write a loop (including initialization) to find both the minimum value in a list and that value’s index in one pass through the list.
2. Write a function named min index that takes a list and returns a tuple containing the minimum value in the list and that value’s index in the list.
3. Write a function named max index that takes a list and returns a tuple containing the maximum value in the list and that value’s index in the list.

In [2]:

```
value = list(map(int, input('input values : ').split()))


def loop(value):
    min_idx = 0
    min_value = value[0]

    for i in range(1, len(value)):
        if value[i] < min_value:
            min_value = value[i]
            min_idx = i

    return min_value, min_idx


def min_index(value):
    return min(value), value.index(min(value))


def max_index(value):
    return max(value), value.index(max(value))


check = int(input('   - If you check the loop function, input a number 1₩n'
                  '   - If you check the minimum function, input a number 2₩n'
                  '   - If you check the maximum function, input a number 3₩n>> '))

if check == 1:
    print("(minimum value, value's index) :", loop(value))
elif check == 2:
    print("(minimum value, value's index) :", min_index(value))
else:
    print("(maximum value, value's index) :", max_index(value))
```

```
input values : 16 24 37 9 5 2 15 38 22 60 45 55 71
   - If you check the loop function, input a number 1
   - If you check the minimum function, input a number 2
   - If you check the maximum function, input a number 3
>> 2
(minimum value, value's index) : (2, 5)
```

Problem 3

Design and implement a class **Country** that stores the information on countries such as nation name, capital city, population, and area. Then write a program that reads in a set of countries and prints

1. the country with the largest area.
2. the country with the largest population.
3. the country with the largest population density.
4. the country with its capital city.

In [3]:

```python
class Country:
    def __init__(self, nation_name, capital_city, population, area):
        self.nation_name = nation_name;
        self.capital_city = capital_city;
        self.population = population;
        self.area = area;
        self.density = round(self.area / self.population, 2);


nation_name = ['Korea', 'USA', 'Japan', 'China', 'Tailand']
capital_city = ['Seoul', 'Washington, D.C', 'Tokyo', 'Beijing', 'Bangkok']
population = [51164435, 326766748, 127185332, 1415045928, 69183173]
area = [100210, 9372610, 377930, 9706961, 513120]

country = []
for i in range(5):
    country.append(Country(nation_name[i], capital_city[i], population[i], area[i]))


def largestArea(country):
    area = []
    for i in range(len(country)):
        area.append(country[i].area)
    large_area = max(area)
    large_area_idx = area.index(large_area)

    print('========== Largest Area Information ==========')
    print('{}, Area is {}₩n'.format(
        country[large_area_idx].nation_name, country[large_area_idx].area
    ))


def largestPopulation(country):
    population = []
    for i in range(len(country)):
        population.append(country[i].population)
    large_pop = max(population)
    large_pop_idx = population.index(large_pop)

    print('======= Largest Population Information =======')
    print('{}, Population is {}₩n'.format(
        country[large_pop_idx].nation_name, country[large_pop_idx].population
    ))


def largestDensity(country):
    density = []
    for i in range(len(country)):
        density.append(country[i].density)
    large_density = max(density)
    large_density_idx = density.index(large_density)

    print('=== Largest Population Density Information ===')
    print('{}, Population Density is {}₩n'.format(
        country[large_density_idx].nation_name, country[large_density_idx].density
    ))


def capitalCity(country):
    print('====== Country Capital City Information ======')

    for i in range(len(country)):
        print('{}, capital city is {}'.format(
            country[i].nation_name, country[i].capital_city
        ))


largestArea(country)
largestPopulation(country)
largestDensity(country)
capitalCity(country)
```

```
========== Largest Area Information ==========
China, Area is 9706961

======= Largest Population Information =======
China, Population is 1415045928

=== Largest Population Density Information ===
USA, Population Density is 0.03

====== Country Capital City Information ======
Korea, capital city is Seoul
USA, capital city is Washington, D.C
Japan, capital city is Tokyo
China, capital city is Beijing
Tailand, capital city is Bangkok
```

Problem 4

Based on object-oriented programming, design and implement each class for geometry objects on the next page.

1. Implement and test the class on each object
2. Place those classes into a **geometry** module. Then write a program that prints a result for the chosen object depending on a user's values.

In [4]:

```
# geometry module
from math import pi
from math import sqrt

class SQUARE:
    def __init__(self, shape, type):
        self.shape = shape
        self.type = type

    def square(self, s):
        parameter = 4*s
        area = s**2
        return [parameter, area]

    def rectangle(self, a, b):
        parameter = 2*a + 2*b
        area = a*b
        return [parameter, area]

    def parallelogram(self, a, b, h):
        parameter = 2*a + 2*b
        area = b*h
        return [parameter, area]

    def trapezoid(self, a, b, c, d, h):
        parameter = a + b + c + d
        area = h*(a+b)/2
        return [parameter, area]

    def rectangularBox(self, a, b, c):
        area = 2*a*b + 2*a*c + 2*b*c
        volume = a*b*c
        return [area, volume]

    def cube(self, l):
        area = 6*l*l
        volume = l**3
        return [area, volume]


class CIRCLE:
    def __init__(self, shape, type):
        self.shape = shape
        self.type = type

    def circle(self, r):
        parameter = 2*pi*r
        area = round(pi*r*r, 2)
        return [parameter, area]

    def circularSector(self, r, seta):
        length = round(pi*r*seta/180, 2)
        area = round(pi*r*r*seta/360, 2)
        return [length, area]

    def circularRing(self, r, R):
        area = round(pi * (R**2 - r**2), 2)
        return area
    def sphere(self, r):
        surface = round(4*pi*r**2, 2)
        volume = round(4*pi*r**3/3, 2)
        return [surface, volume]

    def cylinder(self, r, h):
        area = round(2*pi*r * (r+h), 2)
        volume = round(pi*r*r*h, 2)
        return [area, volume]

class TRIANGLE:
    def __init__(self, shape, type):
        self.shape = shape
        self.type = type

    def triangle(self, a, b, c, h):
        parameter = a + b + c
        area = 0.5*b*h
        return [parameter, area]

    def pythagorean(self, a, b):
        theorem = round(sqrt(a**2 + b**2), 2)
        return theorem

    def rightCircularCone(self, r, h, s):
        area = round(pi*r*r + pi*r*s, 2)
        surface = round(sqrt(r**2 + h**2), 2)
        volume = round(1/3 * pi*r*r*h, 2)
        return [area, surface, volume]

    def frustumCone(self, r, h, R):
        volume = round(1/3 * pi*h * (r**2 + r*R + R**2), 2)
        return volume
```

In [5]:

```
import geometry as geo

print('''
========== Geometry Calculation ===================
=== < shape > ========== < type > =================
=== square ============= parameter / area =========
=== rectangle ========== parameter / area =========
=== circle ============= parameter / area =========
=== triangle =========== parameter / area =========
=== parallelogram ====== parameter / area =========
=== circularSector ===== length / area ============
=== pythagorean ======== theorem ==================
=== circularRing ======= area =====================
=== sphere ============= surface / volume =========
=== trapezoid ========== parameter / area =========
=== rectangularBox ===== area / volume ============
=== cube =============== area / volume ============
=== cylinder =========== area / volume ============
=== rightCircularCone == area / surface / volume ==
=== frustumCone ======== volume ===================
''')

shape, type = map(str, input('input shape and type : ').split())

if shape == "square":
    s = int(input('input s(side) : '))
    if type == "parameter":
        answer = geo.SQUARE(shape, type).square(s)[0]
    elif type == "area":
        answer = geo.SQUARE(shape, type).square(s)[1]

elif shape == "rectangle":
    a, b = map(int, input('input a(width), b(height) : ').split())
    if type == "parameter":
        answer = geo.SQUARE(shape, type).rectangle(a, b)[0]
    elif type == "area":
        answer = geo.SQUARE(shape, type).rectangle(a, b)[1]

elif shape == "parallelogram":
    a, b, h = map(int, input('input a, b, h(height) : ').split())
    if type == "parameter":
        answer = geo.SQUARE(shape, type).parallelogram(a, b, h)[0]
    elif type == "area":
        answer = geo.SQUARE(shape, type).parallelogram(a, b, h)[1]

elif shape == "trapezoid":
    a, b, c, d, h = map(int, input('input a, b, c, d, h(height) : ').split())
    if type == "parameter":
        answer = geo.SQUARE(shape, type).trapezoid(a, b, c, d, h)[0]
    elif type == "area":
        answer = geo.SQUARE(shape, type).trapezoid(a, b, c, d, h)[1]

elif shape == "rectangularBox":
    a, b, c = map(int, input('input a, b, c : ').split())
    if type == "area":
        answer = geo.SQUARE(shape, type).rectangularBox(a, b, c)[0]
    elif type == "volume":
        answer = geo.SQUARE(shape, type).rectangularBox(a, b, c)[1]
```

```python
elif shape == "cube":
    l = int(input('input l(length) : '))
    if type == "area":
        answer = geo.SQUARE(shape, type).cube(l)[0]
    elif type == "volume":
        answer = geo.SQUARE(shape, type).cube(l)[1]

elif shape == "circle":
    r = int(input('input r(radius) : '))
    if type == "parameter":
        answer = geo.CIRCLE(shape, type).circle(r)[0]
    elif type == "area":
        answer = geo.CIRCLE(shape, type).circle(r)[1]

elif shape == "circularSector":
    r, seta = map(int, input('input r, seta : ').split())
    if type == "length":
        answer = geo.CIRCLE(shape, type).circularSector(r, seta)[0]
    elif type == "area":
        answer = geo.CIRCLE(shape, type).circularSector(r, seta)[1]

elif shape == "circularRing":
    r, R = map(int, input('input r(inner), R(outer) : ').split())
    if type == "area":
        answer = geo.CIRCLE(shape, type).circularRing(r, R)

elif shape == "sphere":
    r = int(input('input r : '))
    if type == "surface":
        answer = geo.CIRCLE(shape, type).sphere(r)[0]
    elif type == "volume":
        answer = geo.CIRCLE(shape, type).sphere(r)[1]

elif shape == "cylinder":
    r, h = map(int, input('input r, h(height) : ').split())
    if type == "area":
        answer = geo.CIRCLE(shape, type).cylinder(r, h)[0]
    elif type == "volume":
        answer = geo.CIRCLE(shape, type).cylinder(r, h)[1]

elif shape == "triangle":
    a, b, c, h = map(int, input('input a, b, c, h(height) : ').split())
    if type == "parameter":
        answer = geo.TRIANGLE(shape, type).triangle(a, b, c, h)[0]
    elif type == "area":
        answer = geo.TRIANGLE(shape, type).triangle(a, b, c, h)[1]

elif shape == "pythagorean":
    a, b = map(int, input('input a, b : ').split())
    if type == "theorem":
        answer = geo.TRIANGLE(shape, type).pythagorean(a, b)

elif shape == "rightCircularCone":
    r, h, s = map(int, input('input r, h(height), s(slant height) : ').split())
    if type == "area":
        answer = geo.TRIANGLE(shape, type).rightCircularCone(r, h, s)[0]
    elif type == "surface":
        answer = geo.TRIANGLE(shape, type).rightCircularCone(r, h, s)[1]
    elif type == "volume":
        answer = geo.TRIANGLE(shape, type).rightCircularCone(r, h, s)[2]
```

```
elif shape == "frustumCone":
    r, h, R = map(int, input('input r(small), h, R(large) : ').split())
    if type == "volume":
        answer = geo.TRIANGLE(shape, type).frustumCone(r, h, R)


print('the result is', answer)
```

```
========== Geometry Calculation ===================
=== < shape > ========== < type > =================
=== square ============= parameter / area =========
=== rectangle ========== parameter / area =========
=== circle ============= parameter / area =========
=== triangle =========== parameter / area =========
=== parallelogram ====== parameter / area =========
=== circularSector ===== length / area ============
=== pythagorean ======== theorem ==================
=== circularRing ======= area =====================
=== sphere ============= surface / volume =========
=== trapezoid ========== parameter / area =========
=== rectangularBox ===== area / volume ============
=== cube =============== area / volume ============
=== cylinder =========== area / volume ============
=== rightCircularCone == area / surface / volume ==
=== frustumCone ======== volume ===================

input shape and type : rightCircularCone surface
input r, h(height), s(slant height) : 3 8 10
the result is 8.54
```

Problem 5

Design a class **Msg** that models an e-mail message. A message has a recipient, a sender,and a message text. Support the following methods:

- A constructor that takes the sender and recipient
- A method append that appends a line of text to the message body
- A method str that returns the whole string like this:

From G. D. Hong
To: G. I. Dong
Content: Dear friend, I would like to ....

In [6]:

```
class Msg:
    def __init__(self, sender, recipient, text):
        self.sender = sender
        self.recipient = recipient
        self.text = text

    def append(self, new_text):
        self.text += '₩n₩t' + new_text

    def __str__(self):
        return "From {}₩nTo: {}₩nContent: {}".format(self.sender, self.recipient, self.text)


if __name__ == '__main__':
    sender = input('input a sender : ')
    recipient = input('input a recipient : ')
    text = input('input a text (endpoint is .) : ')

    message = Msg(sender, recipient, text)
    while True:
        new_text = input()
        message.append(new_text)

        if new_text == ".":
            break

    print(message)
```

```
input a sender : G. D. Hong
input a recipient : G. I. Dong
input a text (endpoint is .) : Dear friend,
I would like to
...
bye bye
.
From G. D. Hong
To: G. I. Dong
Content: Dear friend,
        I would like to
        ...
        bye bye
        .
```

Problem 7

Design and implement a class **Letter** for authoring a simple letter. In the constructor, supply the names of the sender and the recipient:

def _ _ init _ _ (self, letterfrom, letterto)

Supply a method

def addLine(self, line)

to add a line of text to the body of the letter. Supply a method

def get_text(self)

that returns the entire text of the letter. The text has the form:

Dear recipient name:
first line of the body
second line of the body
...
last line of the body
Sincerely,
sender name

In [7]:

```
class Letter:
    def __init__(self, letterfrom, letterto):
        self.letterfrom = letterfrom
        self.letterto = letterto
        self.text = ""

    def addLine(self, line):
        self.text = self.text + line + '₩n'

    def get_text(self):
        return "Dear {}: ₩n₩n{}₩nSincerely,₩n₩n{}".format(self.letterto, self.text, self.lett
erfrom)


if __name__=='__main__':
    letterfrom = input("From : ")
    letterto = input("To : ")
    letter = Letter(letterfrom, letterto)
    start = input('Text (endpoint is .) : ')
    letter.addLine(start)

    while True:
        text = input()
        letter.addLine(text)

        if text == '.':
            break

    print(letter.get_text())
```

```
From : Lee Sang Min
To : Park Dong Hak
Text (endpoint is .) : first line of the body
second line of the body
...
last line of the body
.
Dear Park Dong Hak:

first line of the body
second line of the body
...
last line of the body
.

Sincerely,

Lee Sang Min
```

Problem 6

Design and implement functions that perform subtraction, multiplication, and elementwise division by extending the Gobhagi project.

Element-wise division is an operation that performs division between values at the same matrix position. Subtraction, multiplication, and element-wise division operations are used only on equally sized matrices.

Test your program by splitting more than 6 files and write a **Makefile** to generate an execution file.

subtraction.h

```
#include <stdio.h>
#include <stdlib.h>

// extern : global variable of another source file
extern int rowA, colA, rowB, colB;
void subtraction(int **a, int **b);
```

subtraction.c

```
#include "subtraction.h"

void subtraction(int **a, int **b) {
        int **c = malloc(sizeof(int *) * rowA);
        for (int i = 0; i < rowA; i++)
                c[i] = malloc(sizeof(int *) * colA);
        for (int i = 0; i < rowA; i++) {
                for (int j = 0; j < colA; j++) {
                        c[i][j] = a[i][j] - b[i][j];
                }
        }

        printf("result₩n");
        for (int i = 0; i < rowA; i++) {
                for (int j = 0; j < colA; j++) {
                        printf("%d₩t", c[i][j]);
                }
                printf("₩n");
        }
        for (int i = 0; i < rowA; i++)
                free(c[i]);
        free(c);
}
```

multiplication.h

```c
#include <stdio.h>
#include <stdlib.h>

extern int rowA, colA, rowB, colB;
void multiplication(int **a, int **b);
```

multiplication.h

```c
#include "multiplication.h"

void multiplication(int **a, int **b) {
	int **c = malloc(sizeof(int *) * rowA);
	for (int i = 0; i < rowA; i++)
		c[i] = malloc(sizeof(int) * colB);
	int sum;
	for (int i = 0; i < rowA; i++) {
		for (int j = 0; j < colB; j++) {
			sum = 0;
			for (int k = 0; k < colA; k++)
				sum += a[i][k] * b[k][j];
			c[i][j] = sum;
		}
	}

	printf("result₩n");
	for (int i = 0; i < rowA; i++) {
		for (int j = 0; j < colB; j++)
			printf("%d₩t", c[i][j]);
		printf("₩n");
	}
	for (int i = 0; i < rowA; i++)
		free(c[i]);
	free(c);
}
```

division.h

```
#include <stdio.h>
#include <stdlib.h>

extern int rowA, colA, rowB, colB;
void division(int **a, int **b);
```

division.c

```
#include "division.h"

void division(int **a, int **b) {
        int n;
        int idx = 0;
        int **c = malloc(sizeof(int *) * rowA);

        for (int i = 0; i < rowA; i++) {
                c[i] = malloc(sizeof(int) * colA);
                for (int j = 0; j < colA; j++)
                        c[i][j] = 0;
        }

        for (int i = 0; i < rowA; i++) {
                for (int j = 0; j < colA; j++) {
                        if (b[i][j] == 0) {
                                printf("Error : Can't divide by zero");
                                idx = 1;
                                break;
                        }
                        else
                                c[i][j] = a[i][j] / b[i][j];
                }
        }

        if (idx == 0) {
                printf("result₩n");
                for (int i = 0; i < rowA; i++) {
                        for (int j = 0; j < colA; j++)
                                printf("%d₩t", c[i][j]);
                        printf("₩n");
                }
                for (int i = 0; i < rowA; i++)
                        free(c[i]);
                free(c);
        }
        else {
                for (int i = 0; i < rowA; i++)
                        free(c[i]);
                free(c);
        }
}
```

main.c

```
#include <stdio.h>
#include <stdlib.h>
#include "subtraction.h"
#include "multiplication.h"
#include "division.h"

int rowA, colA, rowB, colB;

int main() {
        int num;

        printf("A 행렬의 행(row) : ");
        scanf("%d", &rowA);
        printf("A 행렬의 열(column) : ");
        scanf("%d", &colA);

        int **a = malloc(sizeof(int *) * rowA);
```

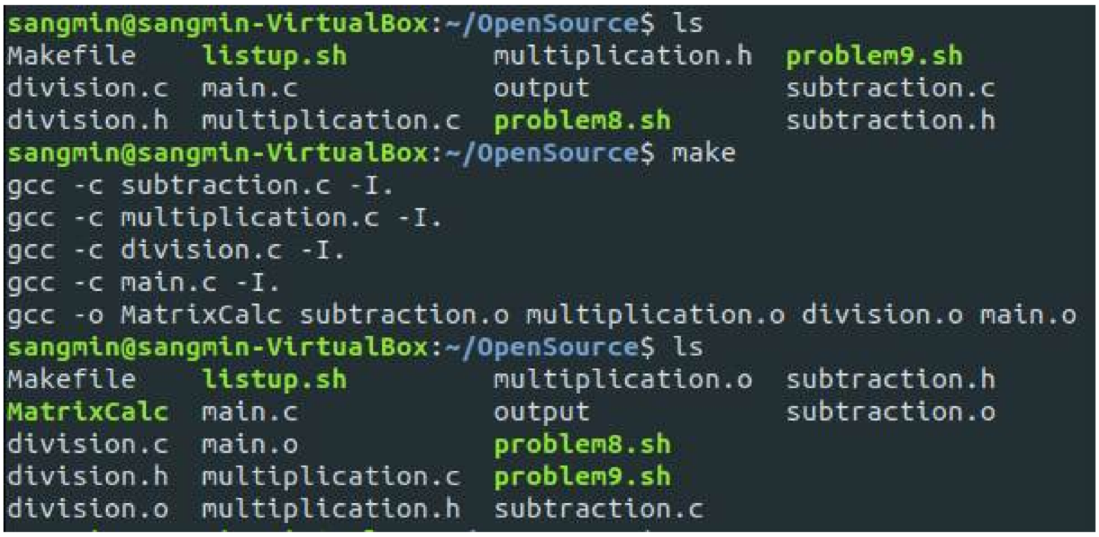
```
for (int i = 0; i < rowA; i++)
        a[i] = malloc(sizeof(int) * colA);

printf(">> A 행렬 입력₩n");
for (int i = 0; i < rowA; i++) {
        printf(">> ");
        for (int j = 0; j < colA; j++) {
                scanf("%d", &num);
                a[i][j] = num;
        }
}
printf("₩n");

printf("B 행렬의 행(row) : ");
scanf("%d", &rowB);
printf("B 행렬의 열(column) : ");
scanf("%d", &colB);

int **b = malloc(sizeof(int *) * rowB);
for (int i = 0; i < rowB; i++)
        b[i] = malloc(sizeof(int) * colB);

printf(">> B 행렬 입력₩n");
for (int i = 0; i < rowB; i++) {
        printf(">> ");
        for (int j = 0; j < colB; j++) {
                scanf("%d", &num);
                b[i][j] = num;
        }
}
printf("₩n");
```

```
	while(1) {
		printf("1. Subtraction / 2. Multiplication / 3. Element-wise
Division / 0. Quit₩n>> ");
		scanf("%d", &num);
		if (num == 1) {
			subtraction(a, b);
		} else if (num == 2) {
			if (colA == rowB)
				multiplication(a, b);
			else
				printf("Check two matrices row and
column");
		} else if (num == 3) {
			if (rowA == rowB && colA == colB)
				division(a, b);
			else
				printf("Check two matrices row and
column");
		} else {
			for (int i = 0; i < rowA; i++)
				free(a[i]);
			free(a);
			for (int i = 0; i < rowB; i++)
				free(b[i]);
			free(b);
			break;
		}
	}

	return 0;
}
```

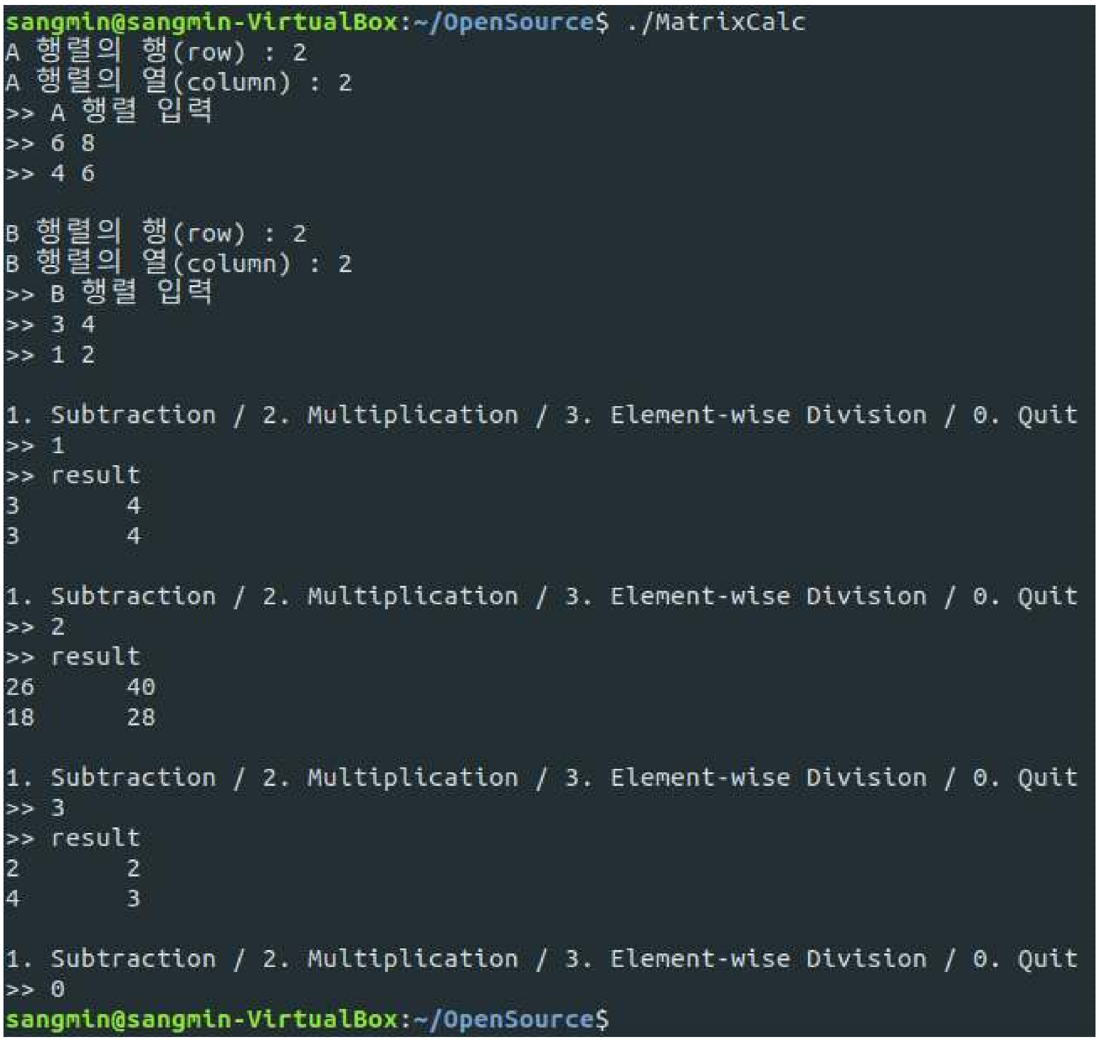
Makefile

```
CC = gcc
CFLAG = -I.

MatrixCalc: subtraction.o multiplication.o division.o main.o
	$(CC) -o MatrixCalc subtraction.o multiplication.o division.o main.o

subtraction.o: subtraction.c subtraction.h
	$(CC) -c subtraction.c $(CFLAG)

multiplication.o: multiplication.c multiplication.h
	$(CC) -c multiplication.c $(CFLAG)

division.o: division.c division.h
	$(CC) -c division.c $(CFLAG)

main.o: main.c subtraction.h multiplication.h division.h
	$(CC) -c main.c $(CFLAG)

clean:
	rm -f *.o MatrixCalc
```

Result



```
sangmin@sangmin-VirtualBox:~/OpenSource$ ls
Makefile    listup.sh          multiplication.h  problem9.sh
division.c  main.c             output            subtraction.c
division.h  multiplication.c   problem8.sh       subtraction.h
sangmin@sangmin-VirtualBox:~/OpenSource$ make
gcc -c subtraction.c -I.
gcc -c multiplication.c -I.
gcc -c division.c -I.
gcc -c main.c -I.
gcc -o MatrixCalc subtraction.o multiplication.o division.o main.o
sangmin@sangmin-VirtualBox:~/OpenSource$ ls
Makefile    listup.sh          multiplication.o  subtraction.h
MatrixCalc  main.c             output            subtraction.o
division.c  main.o             problem8.sh
division.h  multiplication.c   problem9.sh
division.o  multiplication.h   subtraction.c
```



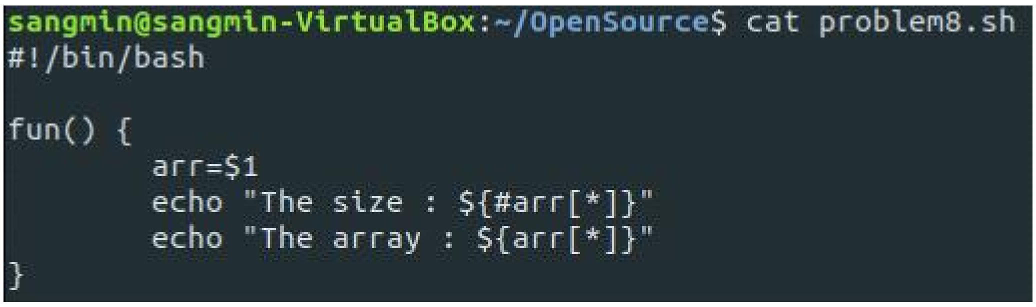
```
sangmin@sangmin-VirtualBox:~/OpenSource$ ./MatrixCalc
A 행렬의 행(row) : 2
A 행렬의 열(column) : 2
>> A 행렬 입력
>> 6 8
>> 4 6

B 행렬의 행(row) : 2
B 행렬의 열(column) : 2
>> B 행렬 입력
>> 3 4
>> 1 2

1. Subtraction / 2. Multiplication / 3. Element-wise Division / 0. Quit
>> 1
>> result
3       4
3       4

1. Subtraction / 2. Multiplication / 3. Element-wise Division / 0. Quit
>> 2
>> result
26      40
18      28

1. Subtraction / 2. Multiplication / 3. Element-wise Division / 0. Quit
>> 3
>> result
2       2
4       3

1. Subtraction / 2. Multiplication / 3. Element-wise Division / 0. Quit
>> 0
sangmin@sangmin-VirtualBox:~/OpenSource$
```

Problem 8

What is the printed value of the following script?
Explain why the results come out.

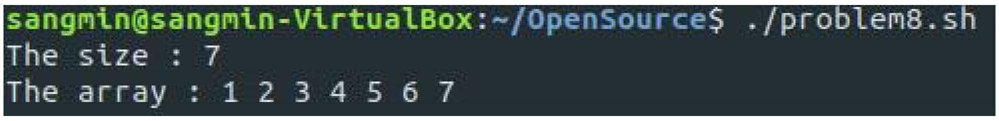
#!/bin/bash
fun(){<br>
arr=$1<br>
echo "The size : ${#arr[*]}"<br>
echo "The array : ${arr[*]}"<br>
}
arr=(1 2 3 4 5 6 7)
fun ${arr[*]}

Code



```
sangmin@sangmin-VirtualBox:~/OpenSource$ cat problem8.sh
#!/bin/bash

fun() {
        arr=$1
        echo "The size : ${#arr[*]}"
        echo "The array : ${arr[*]}"
}
```

Result



```
sangmin@sangmin-VirtualBox:~/OpenSource$ ./problem8.sh
The size : 7
The array : 1 2 3 4 5 6 7
```

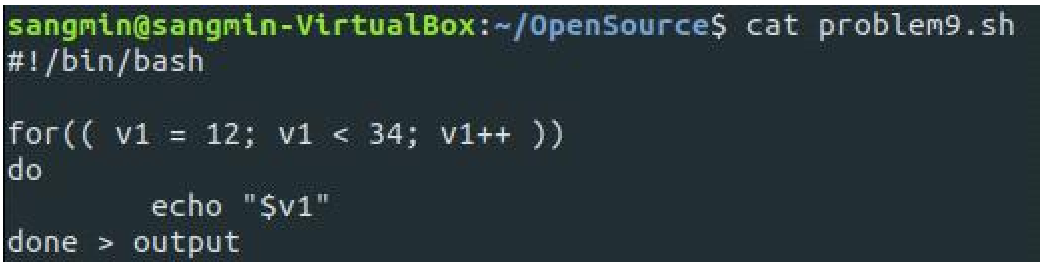
arr 배열이 fun() 함수 안에 인자로 전달된다.
${#arr[*]} : arr 배열의 모든 원소의 개수 (배열의 크기)
${arr[*]} : arr 배열의 모든 원소

Problem 9

How many lines will be printed on screen from the following script?

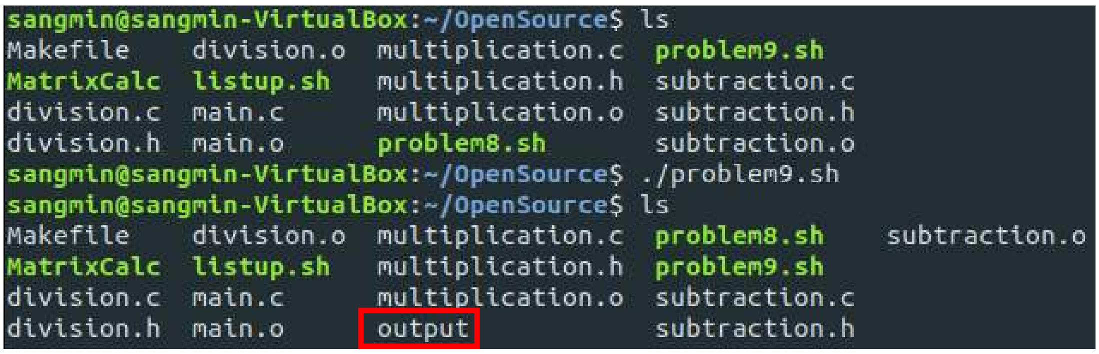
```
#!/bin/bash
for(( v1 = 12; v1 < 34; v1++))
do<br>
        echo "$v1"
done > output
```

Code



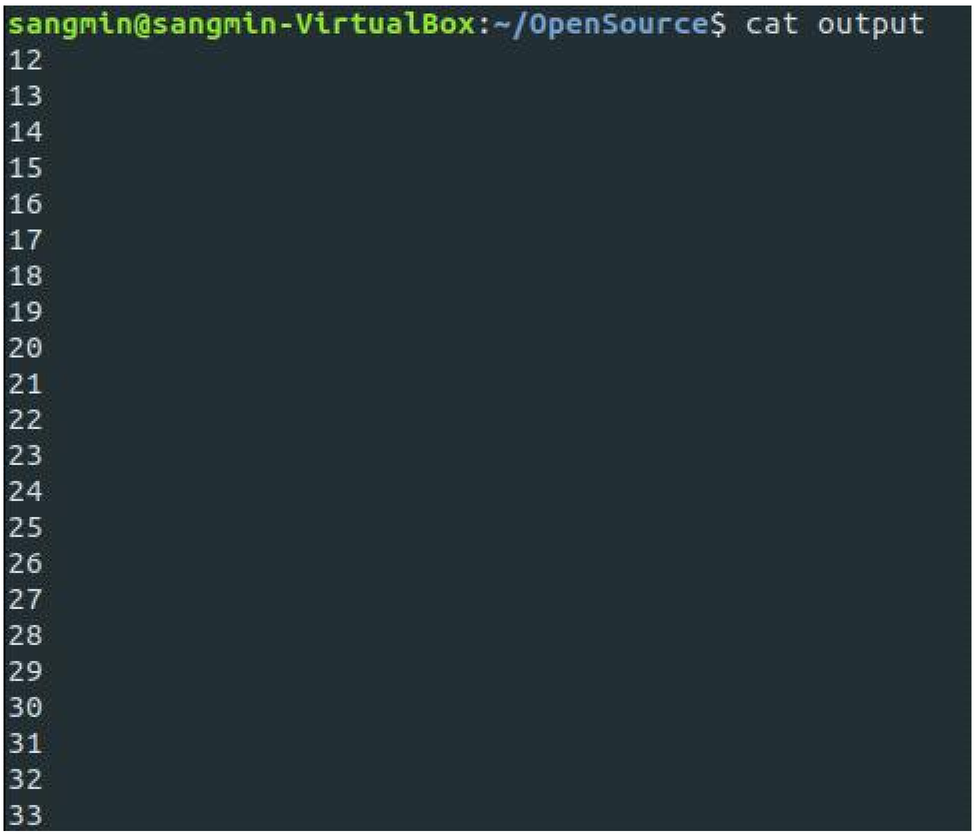
```
sangmin@sangmin-VirtualBox:~/OpenSource$ cat problem9.sh
#!/bin/bash

for(( v1 = 12; v1 < 34; v1++ ))
do
        echo "$v1"
done > output
```

Result



```
sangmin@sangmin-VirtualBox:~/OpenSource$ ls
Makefile    division.o  multiplication.c  problem9.sh
MatrixCalc  listup.sh   multiplication.h  subtraction.c
division.c  main.c      multiplication.o  subtraction.h
division.h  main.o      problem8.sh       subtraction.o
sangmin@sangmin-VirtualBox:~/OpenSource$ ./problem9.sh
sangmin@sangmin-VirtualBox:~/OpenSource$ ls
Makefile    division.o  multiplication.c  problem8.sh    subtraction.o
MatrixCalc  listup.sh   multiplication.h  problem9.sh
division.c  main.c      multiplication.o  subtraction.c
division.h  main.o      output            subtraction.h
```

‘ > ’ 리다이렉션을 통해 터미널에 바로 출력되지 않고 output 결과가 output 파일에 생성된다.



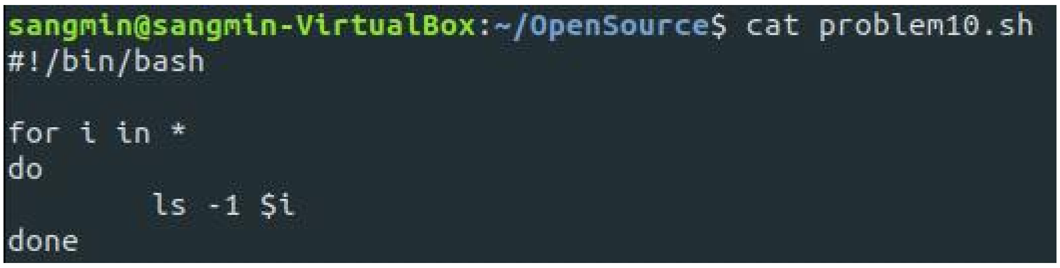
```
sangmin@sangmin-VirtualBox:~/OpenSource$ cat output
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
```

output 파일에 12부터 33까지의 수가 저장된 것을 확인할 수 있다.

Problem 10

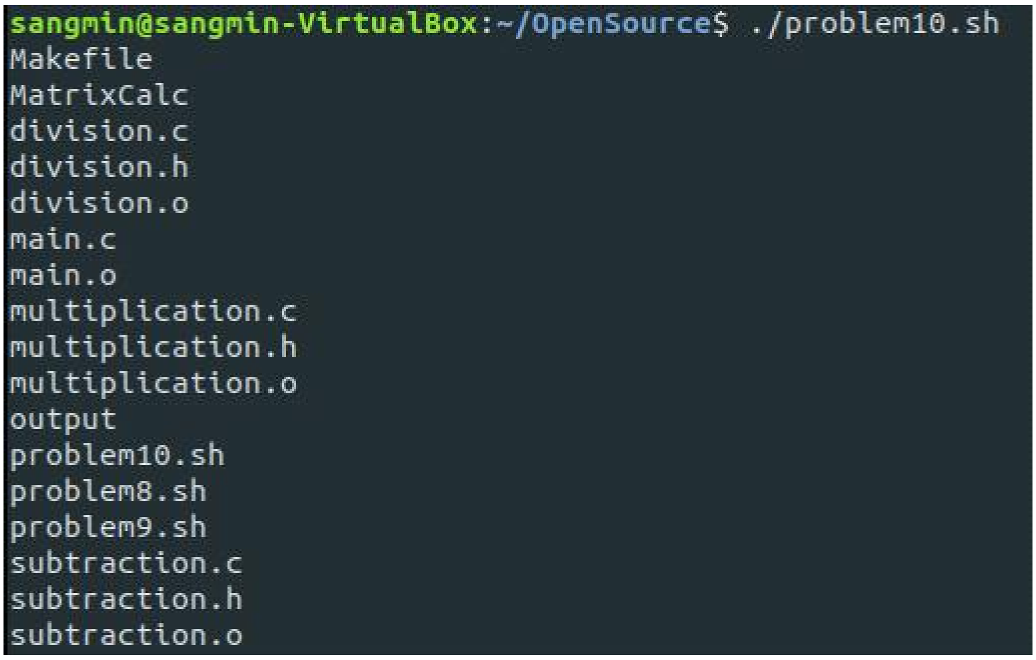
Write a shell script to output the list of files in the current directory. Only file names are printed one per line.

Code



```
sangmin@sangmin-VirtualBox:~/OpenSource$ cat problem10.sh
#!/bin/bash

for i in *
do
        ls -1 $i
done
```

Result



```
sangmin@sangmin-VirtualBox:~/OpenSource$ ./problem10.sh
Makefile
MatrixCalc
division.c
division.h
division.o
main.c
main.o
multiplication.c
multiplication.h
multiplication.o
output
problem10.sh
problem8.sh
problem9.sh
subtraction.c
subtraction.h
subtraction.o
```

실행 결과 현재 디렉토리에 있는 모든 파일의 이름을 확인할 수 있다.