Actividades Parte VII: Extras

Reporte de temperaturas

Las temperaturas mínimas y máximas de algunas ciudades de la región están guardadas en un diccionario cuyas llaves son las ciudades y cuyos valores son tuplas `(mínima, máxima)`

Se desea generar un archivo cuyo contenido sea un reporte como el del ejemplo de más abajo. Los nombres de las ciudades en las que hubo más de 25 grados deben aparecer en mayúsculas. El nombre del archivo debe incluir la fecha. El orden en que aparecen las ciudades dentro del archivo no importa.

Escriba la función `crear_reporte(fecha, temperaturas)`, cuyos parámetros son la fecha (una tupla (año, mes, día)) y el diccionario de temperaturas, y que genere el archivo de texto con el formato del ejemplo.

La función `crear_reporte` no debe retornar nada. Recuerde que `s.upper()` entrega el string s en mayúsculas.

```
temp = {
  'Vina del Mar':  ( 9, 26),
  'Valparaiso':    (10, 24),
  'Quilpue' :      ( 7, 30),
  'Olmue':         ( 5, 29),
  'Limache':       ( 9, 23),
  'Villa Alemana': ( 9, 22),
}
crear_reporte((2011, 5, 14), temp)
```

Archivo `reporte-2011-5-14.txt`:

```
QUILPUE: max 30, min 7
Valparaiso: max 24, min 10
VINA DEL MAR: max 26, min 9
Villa Alemana: max 22, min 9
Limache: max 23, min 9
OLMUE: max 29, min 5
```

Fookbace

La red social Fookbace almacena la información de sus usuarios en un diccionario. Las llaves son un código numérico entero asignado a cada usuario, y los valores son tuplas con el nombre, la ciudad y la fecha de nacimiento del usuario. La fecha de nacimiento es una tupla `(año, mes, día)`:

```
usuarios = {
    522514: ('Jean Dupont',        'Marseille',  (1989, 11, 21)),
    587125: ('Perico Los Palotes', 'Valparaiso', (1990,  4, 12)),
    189471: ('Jan Kowalski',       'Krakow',     (1994,  4, 22)),
    914210: ('Antonio Nobel',      'Valparaiso', (1983,  7,  1)),
```

```
    # ...
}
```

1. Escriba la función `misma_ciudad(u1, u2)`, que indique si los usuarios con códigos u1 y u2 viven en la misma ciudad:

```
>>> misma_ciudad(914210, 587125)
True
>>> misma_ciudad(522514, 189471)
False
```

2. Escriba la función `diferencia_edad(u1, u2)`, que retorne la diferencia de edad entre los usuarios cuyos códigos son u1 y u2. (Utilice sólo el año de nacimiento de los usuarios para calcular la diferencia, no el mes ni el día).

```
>>> diferencia_edad(914210, 587125)
7
```

3. Para guardar la información sobre cuáles de sus usuarios son amigos entre sí, Fookbace utiliza el conjunto amistades, que contiene tuplas con los códigos de dos usuarios. Si la tupla (a, b) está dentro del conjunto, significa que los usuarios con códigos a y b son amigos. En todas las tuplas se cumple que a < b.

```
amistades = {
    (198471, 289142), (138555, 429900), (349123, 781118), # ...
}
```

    1. Escriba la función `obtener_amigos(u)`, que retorne el conjunto de los códigos de los amigos de u.
    2. Escriba la función `recomendar_amigos(u)`, que retorne el conjunto de los códigos de los usuarios que cumplen todas estas condiciones a la vez:
        - son amigos de un amigo de u,
        - no son amigos de u,
        - viven en la misma ciudad que u, y
        - tienen menos de diez años de diferencia con u.

En ambas funciones, el parámetro u es el código de un usuario, y el valor de retorno es un conjunto de códigos de usuarios. Recuerde que `c.add(x)` agrega el valor x al conjunto c.

ADN

Debido a la gran cantidad de crímenes y casos sin resolver, la policía local ha decidido implementar su propio sistema de reconocimiento de sospechosos con la técnica basada en el uso del DNA.

Para esto la policía mantiene dos listas de información; la primera contiene el nombre de las personas registradas en la región (nombre y apellido), la otra, un cromosoma representativo del DNA de cada una de esas personas.

Por simplicidad, un cromosoma se considera como una cadena de 0 (ceros) y 1 (unos), de largo 20.

El método para determinar el sospechoso, es el siguiente:

- Se obtiene una muestra del cromosoma del autor del delito (20 caracteres)
- Se busca en la lista de cromosomas, aquel cromosoma que es *más parecido* a la muestra. El más parecido se define como el cromosoma que tiene más genes (caracteres) iguales a la muestra.
- Al terminar la búsqueda, se muestra el nombre de la persona cuyo cromosoma es sospechoso, con el porcentaje de parentesco.

La información inicial del programa se debe ingresar directamente en el código, es decir, nombres y cromosomas, en cambio la secuencia encontrada en la escena del crimen debe ser ingresada por el usuario.

Desarrolle un programa que lleve a cabo la búsqueda descrita a partir de una muestra de entrada.

Recuerde que como se trata de dos listas, la información del *nombre* como la de los cromosomas, deben estar en la misma posición en ambas listas.

Consideremos, personas como Pedro, Juan y Diego. Sus secuencias serán

```
00000101010101010101
00101010101101110111
00100010010000001001
```

```
Ingrese secuencia: 01010101000011001100
El culpable es Pedro con un parentezco de 60%
```

El juego del ahorcado

Desarrolle un programa para jugar al popular juego *el ahorcado*, el cual consiste en un personaje, el cual está a punto de ser ejecutado.

Para salvarlo es necesario adivinar una palabra, de la cual sólo se conoce su longitud. El jugador debe ir eligiendo letra por letra, de modo de ir completando la palabra.

Si el jugador se equivoca en una letra, es decir, la letra seleccionada no pertenece a la palabra a adivinar, el personaje pierde alguna parte de su cuerpo (un brazo, una pierna, el tronco, etc).

Se puede jugar hasta que el personaje pierda la cabeza, el último resto de su trágica vida.

Lea cada letra de la palabra que debe adivinarse en elementos sucesivos de un string llamado *palabra*. El jugador debe adivinar las letras que pertenecen a la *palabra* y el programa debe terminar cuando todas las letras se hayan adivinado, es decir, ganar el juego, o bien se haya cometido un número establecido de desaciertos, es decir, gana el computador.

Observaciones:

- Utilice un string para anotar las soluciones.

- Asigne el caracter “_” a cada elemento del string, y cada vez que se adivine una letra, substituya el caracter por la letra correspondiente.
- El programa debe realizarse utilzando funcionesfunciones
- Considere las siguientes partes del cuerpo del ahorcadoahorcado:
- pierna derecha
- pierna izquierda
- brazo derecho
- brazo izquierdo
- tronco
- cabeza

```
Ingrese la palabra: caramelo
Comienza el juego!
" _ _ _ _ _ _ _ _ "
Ingrese letra: a
" _ a _ a _ _ _ _ "
Ingrese letra: e
" _ a _ a _ e _ _ "
Ingrese letra: i
Pierde "pierna derecha"
Ingrese letra: o
" _ a _ a _ e _ o "
Ingrese letra: b
Pierde "pierna izquierda"
Ingrese letra: c
" c a _ a _ e _ o "
Ingrese letra: d
Pierde "brazo derecho"
Ingrese letra: f
Pierde "brazo izquierdo"
Ingrese letra: g
Pierde "tronco"
Ingrese letra: h
Pierde "cabeza"

Haz perdido el juego!
```

```
Ingrese la palabra: ola
Comienza el juego!
" _ _ _"
Ingrese letra: a
" _ _ a_"
Ingrese letra: a
Ya ingresaste esa letra!
Ingrese letra: l
" _ l a_"
Ingrese letra: o
" o l a "

Haz ganado!
```

Asignación de salas

El edificio de una escuela secundaria tiene tres pisos, cada uno con cinco salas de clases de tamaños diversos como se muestra en la siguiente tabla:

	Número de sala				
Piso	01	02	03	04	05
1	30	30	15	30	40
2	25	30	25	10	110
3	62	30	40	40	30

Cada semestre, la escuela imparte 15 cursos que deben distribuirse en las salas.

Desarrolle un programa que intente hacer una asignación de salas satisfactoria que acomode 15 clases, teniendo ya los datos de las capacidades de salas y también el tamaño de cada grupo de los cursos.

Para los grupos que no puedan ser adecuadamente ubicados, el programa mostrará el mensaje:

```
No hay sala disponible
```

además, el programa indicará el número de asientos vacíos en cada sala y en todo el edificio.

Para realizar la asignación utilice la estrategia del *mejor ajuste*: para una demanda dada se asigna la sala disponible cuyo exceso de capacidad sea mínimo.

Horario de clases

Desarrolle un programa para manipular la información relativa al horario de clases.

Esta información puede ser almacenada en un arreglo bidimensional donde los índices representan la hora y el día de la semana.

Los elementos del arreglo representan su carga académica. Para cada clase almacene el nombre de la asignatura (o su sigla) y el tipo de la sesión (cátedra, ayudantía, laboratorio, etc).

El programa debe:

- Llenar el horario con sus clases.
- Determinar el día en que las clases comienzan más temprano.
- Determinar el día en que tiene más clases.
- Dado el nombre o sigla de la asignatura mostrar el horario de clases y el horario de ayudantía y laboratorio si es que posee.

Listas ordenadas

Desarrolle un programa que sea capaz de realizar las siguientes operaciones, considerando dos listas A y B con 10 elementos cada una.

- Ingrese los 10 valores de cada lista.
- Ordene ascendentemente A y B
- Mezcle ordenadamente los elementos de A y B en una lista C, el cual no debe tener elementos repetidos.

```
Ingrese valores de A:
2
4
1
10
3
9
5
7
8
6
Ingrese valores de B:
2
8
11
6
1
7
4
17
10
3
A ordenado: 1 2 3 4 5 6 7 8 9 10
B ordenado: 1 2 3 4 6 7 8 10 11 17
C: 1 2 3 4 5 6 7 8 9 10 11 17
```

Maquinaria

Una empresa tiene 15 máquinas las cuales trabajan los 7 días de la semana. Las horas trabajadas por día en cada máquina se guardan en un arreglo bidimensional de 15×715×7 elementos con el fin de realizar las siguientes estadísticas:

- Los promedios de horas trabajabas cada día de la semana por el total de máquinas.
- Los promedios de horas trabajadas por cada máquina en la semana.
- Indicar cuáles fueron las máquinas que trabajaron un mayor y un menor número de horas durante la semana.
- Indicar cuáles fueron los días con el mayor y el menor número de horas trabajadas.

Desarrolle un programa que lea esta información, la almacene en un arreglo bidimensional y calcule las estadísticas pedidas, las que deben ser mostradas por el programa.

No es necesario que el usuario deba ingresar todos los datos, puede escribirlos en el código.

Notas de certámenes

Desarrolle un programa para poder manipular las notas de tres certámenes de los 10 alumnos de un curso:

C1	C2	C3
58	65	53
44	84	60
...	...	...
65	60	80

Para almacenar la información se puede utilizar una matriz de 10 filas y 3 columnas, donde en cada columna para una determinada fila, se almacenará la nota que el alumno, representado por la fila, obtuvo en uno de los tres certámenes (la fila *i* representa al alumno *i* y la nota del certamen 1 está en la columna 1, la nota del certamen 2 en la columna 2, etc).

Además, utilice un arreglo unidimensional de largo 10 para almacenar los nombres de los alumnos. Así en la posición *i* del arreglo estará el nombre del alumno *i*. El programa debe:

- Ingresar el nombre y las notas para todos los alumnos del curso.
- Calcular el promedio de cada alumno.
- Calcular el promedio del curso en cada certamen.
- Calcular el promedio general del curso.
- Calcular la cantidad de alumnos aprobados (promedio >= 55) y reprobados (promedio < 55
- Ordenar alumnos según promedio
- Mostrar el nombre, notas de certámenes y promedio final para cada alumno ordenado.

```
Nombre 1: Manuel Pavez
C1: 55
C2: 60
C3: 75
Nombre 2: Guillermo Fuenzalida
C1: 60
C2: 60
C3: 20
...
Nombre 10: Cristian Perez
C1: 100
C2: 0
C3: 55
Promedio 1: 63.3
Promedio 2: 46.6
...
Promedio 10: 51.6
Promedio del curso C1: 56
Promedio del curso C2: 30
Promedio del curso C3: 60
Promedio Final Curso: 55
```

```
Aprobados: 4
Reprobados: 6
...
Guillermo Fuenzalida 46.6
...
Cristian Perez 51.6
...
Manuel Pavez 63.3
...
```

Contraseñas

Actualmente un método muy utilizado para escoger una contraseña es cambiar ciertas letras de una determinada palabra por números, por ejemplo:

```
me gusta el futbol.
m3 gust4 3l futb0l!
```

Por lo tanto, para poder hacer más fácil esta tarea, realice una función que utilizando:

- una frase.
- un diccionario con los caracteres a reemplazar.

pueda entregar la contraseña con los caracteres reemplazados.

Recuerde la función `replace()`

```
frase = "quiero mi password segura."
diccionario = {'a':4,'e':3,'i':1,'o':0,'.':'?'}
cambia(frase,diccionaio)
"qu13r0 m1 p4ssw0rd s3gur4?"
frase = "gatito bonito"
diccionario = {'t':'n','o':''}
cambia(frase,diccionario)
"ganin bonin"
```

Además, como siempre se desea una contraseña mucho más segura modifique la función anterior, para poder cambiar los caracteres sólo una cantidad *n* de veces.

Por ejemplo:

```
frase = "pablito clavo un clavito"
diccionario = {'a':4,'o':0,'i':1}
cambia(frase,diccionario,1)
"p4bl1t0 clavo un clavito"
cambia(frase,diccionario,3)
"p4bl1t0 cl4v0 un cl4v1t0"
```

Láminas

El módulo random provee funciones para entregar valores al azar. En computación, los números creados al azar se llaman **números aleatorios**.

La función `randrange(n, m)` entrega un número aleatorio en el rango entre n y m:

```
>>> from random import randrange
>>> randrange(5, 15)
14
>>> randrange(5, 15)
6
>>> randrange(5, 15)
14
>>> randrange(5, 15)
5
>>> randrange(5, 15)
12
>>> randrange(5, 15)
9
```

(Recuerde que los rangos incluyen el primer valor pero no el último, así que la función retorna números entre 5 y 14).

1. Suponga que Pepito colecciona un álbum de láminas. Las láminas están numeradas desde 1 hasta 640, y se compran en sobres de cinco láminas al azar.

Escriba la función `nuevo_sobre()` que entregue una lista con las láminas que vienen en un sobre recién comprado:

```
>>> nuevo_sobre()
[27, 31, 207, 455, 529]
>>> nuevo_sobre()
[66, 577, 481, 171, 602]
>>> nuevo_sobre()
[275, 493, 167, 25, 592]
>>> nuevo_sobre()
[113, 35, 592, 560, 244]
```

2. Pepito lleva un registro de sus láminas en una lista llamada `laminas_pepito`. Cada ciertos días, Pepito va al quiosco y compra algunos sobres, y los agrega a su lista.

Escriba la función `agregar_laminas(lista_laminas, m)`, que agregue las láminas de m nuevos sobres a la `lista_laminas`:

```
>>> laminas_pepito = []
>>> agregar_laminas(laminas_pepito, 1)
>>> laminas_pepito
[190, 130, 53, 537, 167]
>>> agregar_laminas(laminas_pepito, 2)
>>> laminas_pepito
[190, 130, 53, 537, 167, 572, 537, 375, 496, 469, 249, 545, 95, 279, 131]
>>> agregar_laminas(laminas_pepito, 14)
>>> len(laminas_pepito)
85
```

Note que la función no retorna nada. Sólo modifica la lista que recibe como parámetro.

3. Escriba la `función faltantes(lista_laminas)` que entregue el conjunto de las láminas que faltan para completar el álbum:

```
>>> laminas_pepito = []
>>> agregar_laminas(laminas_pepito, 128)
>>> faltantes(laminas_pepito)
{514, 3, 5, 7, 10, 523, 12, 525, 14, 16, 529, ...}
```

Note que Pepito compró 128 sobres, que en total tienen el mismo número de láminas del álbum, pero como hay muchas láminas repetidas y otras que no salen, no es suficiente para completar el álbum.

4. Escriba la función `cuenta(lista_laminas)` que entregue un diccionario que asocie a cada lámina el número de veces que está en la lista de láminas:

```
>>> laminas_pepito = [4, 6, 9, 12, 9, 9, 6, 12, 2]
>>> cuenta(laminas_pepito)
{9: 3, 2: 1, 4: 1, 6: 2, 12: 2}
```

5. Pepito intercambia láminas con Yayita, que también colecciona el álbum. A Pepito le interesa obtener las láminas que Yayita tiene repetidas y que a él le faltan, y viceversa.

Escriba la función `cuales_me_sirven(lista_quiere, lista_tiene)` que entregue el conjunto de las láminas que le faltan a `lista_quiere` y que `lista_tiene` tiene repetidas:

```
>>> laminas_pepito = [4, 6, 9, 12, 9, 9, 6, 12, 2]
>>> laminas_yayita = [4, 9, 7, 7, 4, 4, 8]
>>> cuales_me_sirven(laminas_pepito, laminas_yayita)
{7}
>>> cuales_me_sirven(laminas_yayita, laminas_pepito)
{12, 6}
```

A Pepito le falta la lámina 7, que Yayita tiene repetida. También le falta la 8, pero ella no la tiene repetida, así que no le sirve. Yayita tiene repetida la 4, pero Pepito ya la tiene, así que tampoco le sirve.

6. El sobre de láminas vale $250. Pepito quiere saber cuánto va a gastar en láminas para completar el álbum.

Escriba la función `costo_laminas()` que vaya comprando sobres hasta completar las 640 láminas distintas, y que retorne cuál fue el gasto total:

Si no sale ninguna repetida, el resultado será:

```
>>> costo_laminas()
32000
```

Si salen algunas repetidas:

```
>>> costo_laminas()
54250
```

Muy mala suerte:

```
>>> costo_laminas()
241750
```

7. Vladimiro es un fanfarrón: él desea sacar pica a Yayita por las láminas que él tiene y que ella no.

Escriba la función `tengo_y_tu_no(mis_laminas, tus_laminas)` que entregue el conjunto de láminas que están en `mis_laminas` y no en `tus_laminas`:

```
>>> laminas_vladimiro = [6, 1, 3, 3, 4, 7]
>>> laminas_yayita = [8, 4, 9, 12, 2, 11, 4, 6, 13, 14]
>> tengo_y_tu_no(laminas_vladimiro, laminas_yayita)
{1, 3, 7}
```

Mayúsculas y minúsculas

Cuando debemos trabajar con distintos textos, nos encontramos con situaciones donde las mayúsculas y las minúsculas no son siempre bien utilizadas.

Por lo general, cuando tenemos un párrafo de texto las reglas son que la primera letra debe ser mayúscula y si es que hay algún punto seguido, también.

Por otro lado, cuando nos referimos a nombres de personas o instituciones, estas deben utilizar la primera letra de cada palabra, o sin son siglas deben ir todas en mayúsculas.

Utilice *conjuntos* para poder almacenar las siglas, nombres e instituciones, y corrija el siguiente texto:

```
La universidad se DEBE fundamentalmente a federico santa maría carrera,
Quien con una impORtante Donación la hizo pSIble y a Agustín Edwards Mc
Clure,
Albacea de Santa María, Y, por ENde, ejecutor de su volunTAD Testamentaria
de dotar a Valparaíso de un centro de estudio compuesto de una Escuela de
ArteS y Oficios y un ColeGIO de ingenieros.
consiDERAndo las SUGerencias hechas por EDWARDS, en el sentido de
asignarle
colaboradores para esa tarea, santa maría otorgó un testamento, cerrado en
París,
con fecha 6 de eNero dE 1920.Reducidos a escRItura pública los estatutos
de
la Fundación fedeRIco sANta maRía constituida por LOS albaceas, Fueron
Aprobados
POR Decreto Supremo del 27 de abril de 1926.

eXtracto de LA reseña histórica, utfsm.
```

Considere las funciones *upper()*, *lower()* y *swapcase()*.

Secuencia ADN

Escriba una función con la que se pueda buscar una subsecuencia determinada en una gran secuencia de ADN, entregada por el usuario.

Considere la siguiente secuencia:

```
gtgggggggtttatgcctttagaacagcagactactgataactccaatcctgggttgaaa
atgccaagggcgccagagagccaaacgatgagcgttggaccacaaacgataaaaactcac
tttctccgtggggtgaaagcgattctttctggcccgtatccgccagcacttaaagttgca
ttcggcgcggccctaccgctgctaattggggtaattgtcctaggattgtacgtaacgctt
ggcgggcacagccgcaagaaagcccacgcagccgcgatagatgctttggtcgagaagcac
gaagcatgctacaaggtccaagcaaagattgcacacggcaggcttgccttacagtccgct
gtggtgtctgttgcggatgccagcatgcaacaactccagttcgtgcagcaaggaattctc
atgtgtgtcggagagctcgacgatatgcagaagttccggacccgactggataatgaaatc
agtgccatcaaccagcgaattcccagcattgtcgaggaggtaagaaaacacaccgacgat
gcgcttgagtggaatcttgctagaaccaagaacattttagagggcactgaagagcgcctg
aaggatatgggcaatgagttggtgcgctacctagacgatgctcgcgccctcattgaaaat
gcacgtatagctgcaggatcaatgcaacacctcgttggtgatgaggtgagaaagcagctt
gctgaggttctagtaaaagttgcagaagtaagtaatggctttattgcgcttaagaagagt
gtatctggctatttggaaaaaagcagtggacttgttgctagggaagttagggcaatcctg
gatgaccgcatgcgaagcctgcggaccatgtacaaaatgtgggatgcagaacaaaactcc
gtagtcagcgtgtgtaccacgctccaaaaggcaagcatggaggctgccgcggtagcaagt

>>> esta_en_adn("ttt", secuencia)
True
>>> esta_en_adn("aaaaaa", secuencia)
True
>>> esta_en_adn("hola", secuencia)
False
```

Binarios

El sistema binario es un sistema de numeración en los cuales los números se representan sólo utilizando 0 y 1.

Realice una función llamada binario(n), la cual entrada el número binario correspondiente para un número n entero.

Para transformar un número decimal a binario se debe dividir el número por 2 y almacenar el resto, proceso que se repite hasta que el resultado de dicha división sea 0. Finalmente, el número deseado es el recorrido en orden inverso de los restos almacenados.

Por ejemplo, al transformar el número decimal 3 al binario, se obtiene:

```
3 : 2 = 1 con resto = 1
1 : 2 = 0 con resto = 1
```

Por lo tanto, 3 en binario es 11

Para el número 7, por ejemplo:

```
7 : 2 = 3 con resto = 1
3 : 2 = 1 con resto = 1
1 : 2 = 0 con resto = 1
```

Por lo tanto, 7 en binario es 111

De la misma forma, para poder pasar un número binario a entero decimal, se logra entender el procedimiento.

Primero se invierte el número y se comienza a multiplicar cada elemento por las potencia de 2:

$$Nrooriginal: 1101 Nroinvertido: 1011 Mult: 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^3 + 1 \times 2^4 : 25$$

Desarrolle una función decimal(n) la cual haga el trabajo de transformar el binario a una representación de entero decimal:

```
>>> binario(10)
1010
>>> binario(123)
1111011
>>> decimal(10111)
23
>>> decimal(1101111)
111
```

Número de letras

Desarrolle un programa que lea una cantidad de palabras y calcule el largo de cada una, pero sin considerar las letras repetidas, determinando la más larga y más corta.

Por ejemplo, la palabra «casas» es más corta que la palabra «reno», ya que tiene 3 letras distintas (*c*, *a* y *s*), mientras que «reno» tiene 5.

```
ingrese n: 4
palabra 1: mascotas
palabra 2: dinosaurio
palabra 3: camarote
palabra 4: siniestro

La palabra mas larga es: dinosaurio
La palabra mas corta es: mascotas
ingrese n: 3
palabra 1: sabanas 4
palabra 2: canales 6
palabra 3: cojines 7

La palabra mas larga es: cojines
La palabra mas corta es: sabanas
```

Distancias

La siguiente tabla indica las distancias entre algunas ciudades chilenas:

Distancia	Arica	Santiago	Valparaíso	San Fernando	Temuco
Arica	0	2050	2015	2190	2725
Santiago	2050	0	119	140	675
Valparaíso	2015	119	0	255	790
San Fernando	2190	140	255	0	535
Temuco	2725	675	790	535	0

En un programa, esta información puede representarse mediante un arreglo de ciudades y un arreglo bidimensional de distancias.

```
Escriba la función que pregunte al usuario la cantidad de ciudades y los
nombres de las ciudades:
 ¿Cuántas ciudades?
5
 Ingrese los nombres:
Arica
Santiago
Valparaíso
San Fernando
Temuco
Escriba la función que pregunte al usuario las distancias entre cada par
de ciudades:
 ¿Distancia Arica-Santiago?
2050
 ¿Distancia Arica-Valparaíso?
2015
...
 ¿Distancia Temuco-San Fernando?
535
```

Tener la precaución de preguntar sólo una vez cada par de ciudades: si se pregunta Arica-Santiago, no debe preguntarse Santiago-Arica. Tampoco debe preguntarse la distancia desde una ciudad a sí misma.

```
Escriba la función que pida al usuario que ingrese una lista de ciudades,
que representan un itinerario por realizar:
 ¿Cuántas ciudades tiene el itinerario?
4
 Ingrese las ciudades del itinerario:
Temuco
Santiago
San Fernando
Arica
```

Escriba la función kms que entregue como resultado los kilómetros que hay que recorrer para visitar las ciudades en orden. Por ejemplo, para el itinerario de arriba, debe entregar como resultado 30053005.