

Taller de conocimientos previos

Listas

Ejercicio #1

Invitados a la reunión

Suponga que usted está con un amigo en una reunión y deciden hacer una apuesta la cual consiste en adivinar la primera letra del nombre de cada persona que vaya entrando a la reunión. Para ello cada uno debe proponer una lista de letras (puede ser de 3 a 6 letras) antes de empezar la apuesta. Cada vez que entre una persona se mira quién de los dos tiene la letra inicial del nombre de la persona y se sacan las cuentas. Si usted lleva la ventaja entonces se anota un 1, si su amigo lleva la ventaja entonces se anota un 2, si van empatados entonces se coloca un asterisco.

Realizar un programa que pida las letras que usted propone y las letras que propone su amigo (como cadena de caracteres) y que también pida una cadena de caracteres con la primera letra del nombre de las diferentes personas que entran a la reunión. Finalmente, se debe mostrar lo que se anota a medida que van entrando las personas.

Entrada

La primera entrada es la lista de letras que usted propone. La segunda entrada es la lista de letras que su amigo propone. La tercera entrada es la lista de iniciales de los nombres de personas que llegan a la reunión.

Salida

La salida es la lista de lo que van anotando a medida que van entrando las personas.

Entrada	Salida
QCWKFI	111**22*1*****1*22
LGECUO	2
KATOAETFQUAPMVKLGM T	
MHJW	22222**1*****1****
PMTZ	
PYAFKHRWZOUYMJPMQM	

Diccionarios

Ejercicio #2

Codifica un programa en python que nos permita guardar los nombres de los alumnos de una clase y las notas que han obtenido. Cada alumno puede tener distinta cantidad de notas. Guarda la información en

un diccionario cuya claves serán los nombres de los alumnos y los valores serán listas con las notas de cada alumno.

El programa pedirá el número de alumnos que vamos a introducir, pedirá su nombre e irá pidiendo sus notas hasta que introduzcamos un número negativo. Al final el programa nos mostrará la lista de alumnos y la nota media obtenida por cada uno de ellos. Nota: si se introduce el nombre de un alumno que ya existe el programa nos dará un error.

Ejercicio #3

Escribir un programa que implemente una agenda. En la agenda se podrán guardar nombres y números de teléfono. El programa nos dará el siguiente menú:

- Añadir/modificar: Nos pide un nombre. Si el nombre se encuentra en la agenda, debe mostrar el teléfono y, opcionalmente, permitir modificarlo si no es correcto. Si el nombre no se encuentra, debe permitir ingresar el teléfono correspondiente.
- Buscar: Nos pide una cadena de caracteres, y nos muestras todos los contactos cuyos nombres comiencen por dicha cadena.
- Borrar: Nos pide un nombre y si existe nos preguntará si queremos borrarlo de la agenda.
- Listar: Nos muestra todos los contactos de la agenda.

Implementar el programa con un diccionario.

Ejercicio #4

Escribir un programa que cree un diccionario simulando una cesta de la compra. El programa debe preguntar el artículo y su precio y añadir el par al diccionario, hasta que el usuario decida terminar. Después se debe mostrar por pantalla la lista de la compra y el coste total, con el siguiente formato

Lista de la Compra	
Artículo 1	Precio
Artículo 2	Precio
Artículo 3	Precio
Artículo 4	Precio
TOTAL	Costo total

Funciones

Ejercicio #5

Aplicativo para un parqueadero

El parqueadero Autoseguro necesita crear un programa para que sus empleados puedan calcular el valor a pagar por el uso del parqueadero. Solamente se reciben 3 tipos de vehículos: automóviles, camionetas y motos.

El valor estipulado para cada uno de estos vehículos por minuto es de 2000, 3000 y \$1000 respectivamente. El parqueadero sólo funciona de 7:00 de la mañana a las 7:00 de la noche. El programa debe capturar la placa del vehículo, el tipo de vehículo (se debe presentar un menú con los 3 tipos y el valor por minuto), la hora de entrada y la hora de salida (se necesita utilizar la hora militar, es decir de 24 horas). Como salida se debe mostrar la información capturada, la cantidad de minutos que duró el vehículo estacionado y el valor a pagar. Debe utilizar funciones para: capturar la información, calcular la cantidad de minutos y el valor a pagar por el parqueadero, y mostrar la información. Todos los datos ingresados por el usuario deben ser validados. El programa debe funcionar mientras el usuario necesite ingresar más registros.

Ejercicio #6

Escribir una función `sum()` y una función `multip()` que sumen y multipliquen respectivamente todos los números de una lista. Por ejemplo: `sum([1,2,3,4])` debería devolver 10 y `multip([1,2,3,4])` debería devolver 24

Ejercicio #7

Crear dos funciones que permitan dibujar un triángulo con el carácter `*`, de manera que el dibujo lo realicen dos funciones que reciban un parámetro (el ancho), una que dibuje la primera mitad del triángulo y la otra que dibuje la segunda mitad. El dibujo debe ser similar al siguiente.



Ejercicio #7

Aplicativo para un parqueadero

El parqueadero Autoseguro necesita crear un programa para que sus empleados puedan calcular el valor a pagar por el uso del parqueadero. Solamente se reciben 3 tipos de vehículos: automóviles, camionetas y motos.

El valor estipulado para cada uno de estos vehículos por minuto es de 2000, 3000 y \$1000 respectivamente. El parqueadero sólo funciona de 7:00 de la mañana a las 7:00 de la noche. El programa debe capturar la placa del vehículo, el tipo de vehículo (se debe presentar un menú con los 3 tipos y el valor por minuto), la hora de entrada y la hora de salida (se necesita utilizar la hora militar, es decir de 24 horas). Como salida se debe mostrar la información capturada, la cantidad de minutos que duró el vehículo estacionado y el valor a pagar. Debe utilizar funciones para: capturar la información, calcular la cantidad de minutos y el valor a pagar por el parqueadero, y mostrar la información.

Todos los datos ingresados por el usuario deben ser validados. El programa debe funcionar mientras el usuario necesite ingresar más registros