Programación PYTHON
Hasta módulo 06

Consideraciones y buenas prácticas que adoptamos para el desarrollo de programas

- Usar nombre de variables estilo camelCase (ejemplos: fechaDeNacimiento, deudaOriginalActualizada, etc.)
- No utilizar acentos ni eñes en los nombres de variables, funciones ni programas
- Documentar el código con comentarios (#comentario """comentario""")
- Estructurar el código en bloques (ejemplo: inicializaciones > pedidos de datos > operaciones > resultados)
- Indentar (tabular) el código correctamente (utilizar la tecla TAB)
- Dejar espacio entre operadores y expresiones (ejemplo: precio=neto+iva debería codificarse precio = neto + iva)
- Usar funciones cuando se pida, y luego, aunque no se pida expresamente utilizarlas para modularizar el código
- Dar nombre a los proyectos y programas con la codificación TPXX-YY (XX = número de TP; YY número de ejercicio)

INDICE

TRABAJO PRÁCTICO 00 | PROGRAMAS BÁSICOS

TP00-01 | AUTONOMÍA DE VEHÍCULO

Desarrollar un programa que calcule cuántos kilómetros podrá recorrer un auto de acuerdo con la cantidad de litros de combustible ingresados y al tipo de camino indicado (ruta o ciudad).

Rendimiento del vehículo:
14.1 km por litro en ruta
10.3 km por litro en ciudad

TP00-02 | ASIENTOS DE CONFERENCIA

Realizar un programa que permita ingresar la cantidad de inscriptos a una conferencia y la cantidad de asientos disponibles en el auditorio. Se debe indicar si alcanzan los asientos. Si los asientos no alcanzan, indicar cuantos faltan para que todos los inscriptos puedan sentarse.

TP00-03 | COBRO Y VUELTO

Escribir un programa básico de caja, donde se ingrese el precio total de la compra, luego se ingrese el monto con el cual el cliente abona la compra, y finalmente informe con un mensaje si no es suficiente con lo que abonó o, caso contrario, informe el vuelto que se le debe dar al cliente.

TP00-04 | AUMENTO DE LÍMITES DE TARJETAS

Un banco necesita establecer los nuevos límites de crédito de sus tarjetas. Las de tipo 1 aumentarán un 25%; las de tipo 2 aumentarán un 35%; las de tipo 3 aumentarán un 40%, y las de cualquier otro tipo aumentarán un 50%. Desarrollar un algoritmo para calcular el nuevo límite según el límite actual y el tipo de tarjeta del cliente.

TP00-05 | COMPRA TOTAL Y CANTIDAD

En un mercado los clientes pueden comprar sólo una unidad de cada producto. Realizar un programa que pida uno por uno los precios de los productos comprados por el cliente, y que al ingresar un precio igual a cero muestre el total que debe abonar por la compra y la cantidad de productos comprados.

TP00-06 | PROMEDIO DE CURSO

Realizar un programa donde se vayan ingresando las calificaciones de los alumnos de un curso. Luego de ingresar la calificación del último alumno, se ingresará un -1 para terminar la carga. El programa informará entonces la calificación promedio del curso.

TP00-07 | FORMAS DE PAGO

Escribir un programa que, ingresado el precio de lista de un producto, muestre cuanto le costará al cliente según todas las opciones de pago disponibles (si es en cuotas además del precio final debe mostrar el valor de cada cuota). Los descuentos o recargos según las formas de pago son los siguientes:

En efectivo aplicar 10% de descuento
Tarjeta 1 pago mantener el precio de lista
Tarjeta 3 pagos recargar 5%
Tarjeta 6 pagos recargar 10%
Tarjeta 12 pagos recargar 15%

Una vez mostrados los valores, el algoritmo debe esperar un nuevo ingreso, y sólo debe finalizar si se ingresa un precio de 0 pesos (en dicho caso debe terminar sin calcular nada). Se pide usar un tipo de bucle que evite tener que escribir el input dos veces.

TP00-08 | PROMEDIO DE TEMPERATURAS

Realizar un programa que solicite la carga de las temperaturas de todos los días de enero y al finalizar devuelva la temperatura promedio, máxima y mínima del mes.

TRABAJO PRÁCTICO 01 | FUNCIONES

TP01-01 | MAYOR ENTRE TRES NÚMEROS

Desarrollar una función que reciba tres números positivos y devuelva el mayor de los tres, sólo si éste es único. En caso de no existir un valor mayor único entonces devolver -1. No utilizar operadores lógicos (and, or, not). Desarrollar también un programa para ingresar los tres valores, invocar a la función y mostrar el máximo hallado, o un mensaje informativo si éste no existe.

TP01-02 | FECHA VÁLIDA

Desarrollar una función que reciba tres números enteros positivos correspondientes al día, mes y año de una fecha, y verifique si corresponden a una fecha válida. Debe tenerse en cuenta la cantidad de días de cada mes, incluyendo los años bisiestos. La función debe devolver True o False según la fecha sea correcta o no. Realizar también un programa para verificar el comportamiento de la función.

TP01-03 | GASTO DE TRANSPORTE SUBTE

Una persona desea llevar el control de los gastos realizados al viajar en el subterráneo dentro de un mes. Sabiendo que dicho medio de transporte utiliza un esquema de tarifas decrecientes (detalladas en la tabla de abajo) se solicita desarrollar una función que reciba como parámetro la cantidad de viajes realizados en un determinado mes y devuelva el total gastado en viajes. Realizar también un programa para verificar el comportamiento de la función.

Cantidad de viajes	Valor de 1 pasaje
1 a 20	Averiguar en internet el valor actualizado
21 a 30	20% de descuento
31 a 40	30% de descuento
41 o más	40% de descuento

TP01-04 | BILLETES SEGÚN VUELTO

Un comercio de electrodomésticos necesita para su línea de cajas un programa que le indique al cajero el cambio que debe entregarle al cliente. Para eso se ingresan dos números enteros, correspondientes al total de la compra y al dinero recibido. Informar cuántos billetes de cada denominación deben ser entregados al cliente como vuelto, de tal forma que se minimice la cantidad de billetes. Considerar que existen billetes de $5000, $1000, $500, $200, $100, $50 y $10. Emitir un mensaje de error si el dinero recibido fuera insuficiente. Ejemplo: Si la compra es de $3170 y se abona con $5000, el vuelto debe contener 1 billete de $1000, 1 billete de $500, 1 billete de $200, 1 billete de $100 y 3 billetes de $10.

TP01-05 | OBLONGOS Y TRIANGULARES

Escribir dos funciones separadas que reciban un número natural y devuelvan verdadero o falso según el número sea de alguna de las siguientes categorías:

Función oblongo(): Informa si un número es oblongo. Se dice que un número es oblongo cuando se puede obtener multiplicando dos números naturales consecutivos. Por ejemplo 6 es oblongo porque resulta de multiplicar 2 * 3.

Función triangular(): Informa si un número es triangular. Un número se define como triangular si puede expresarse como la suma de un grupo de números naturales consecutivos comenzando desde 1. Por ejemplo 10 es un número triangular porque se obtiene sumando 1+2+3+4.

Opcional: Desarrollar estas funciones pero bajo la forma de funciones lambda.

TP01-06 | CONCATENAR BÁSICO

Desarrollar una función que reciba como parámetros dos números enteros positivos y devuelva el número que resulte de concatenar ambos valores. Por ejemplo, si recibe 1234 y 567 debe devolver 1234567. No se permite utilizar facilidades de Python no vistas en clase, ni tampoco concatenar strings mediante la conversión de número a cadena.

TP01-07 | DÍA SIGUIENTE

Escribir una función diaSiguiente() que reciba como parámetro una fecha cualquiera expresada por tres enteros (correspondientes al día, mes y año) y calcule y devuelva tres enteros correspondientes el día siguiente al dado. Utilizando esta misma función, sin modificaciones ni agregados, desarrollar programas que permitan:

- Programa TP01-07A: Sumar N días a una fecha.
- Programa TP01-07B: Calcular la cantidad de días existentes entre dos fechas cualesquiera.

TP01-08 | DÍA DE LA SEMANA

La siguiente función permite averiguar el día de la semana para una fecha determinada. La fecha se suministra en forma de tres parámetros enteros y la función devuelve 0 para domingo, 1 para lunes, 2 para martes, etc. Escribir un programa para imprimir por pantalla el calendario de un mes completo, correspondiente a un mes y año cualquiera basándose en la función suministrada. Considerar que la semana comienza en domingo.

```
def diaDeLaSemana(dia, mes, anio):
    """
    Función para calcular a que día de la semana corresponde una fecha (0,1,2,3,4,5,6).
    PARÁMETROS:
        dia, mes, anio: fecha para la cual obtener el día de la semana.
    SALIDA:
        Entero indicando el día de la semana (0,1,2,3,4,5,6) (0 = domingo)
    """
    if mes < 3:
        mes = mes + 10
        anio = anio - 1
    else:
        mes = mes - 2
    siglo = anio // 100
    anio2 = anio % 100
    diaSem = (((26 * mes - 2) // 10) + dia + anio2 + (anio2 // 4) + (siglo // 4) - (2 * siglo)) % 7
    if diaSem < 0:
        diaSem = diaSem + 7
    return diaSem
```

TP01-09 | CAJONES DE NARANJAS

Resolver el siguiente problema utilizando funciones: Un productor frutihortícola desea contabilizar sus cajones de naranjas según el peso para poder cargar el camión de reparto. La empresa cuenta con N camiones, y cada uno puede transportar hasta media tonelada (500 kilogramos). En un cajón caben 100 naranjas con un peso entre 200 y 300 gramos cada una. Si el peso de alguna naranja se encuentra fuera del rango indicado, se clasifica para procesar como jugo. Se solicita desarrollar un programa para ingresar la cantidad de naranjas cosechadas e informar cuántos cajones se pueden llenar, cuántas naranjas son para jugo y si hay algún sobrante de naranjas que deba considerarse para el siguiente reparto. Simular el peso de cada unidad generando un número entero al azar entre 150 y 350.

Además, se desea saber cuántos camiones se necesitan para transportar la cosecha, considerando que la ocupación del camión no debe ser inferior al 80%; en caso contrario el camión no será despachado por su alto costo.

TRABAJO PRÁCTICO 02 | LISTAS

TP02-01 | MATERIAS CURSANDO

Escribí un programa donde se declare dentro del mismo código una lista con todas las materias que estás cursado en la facultad (no es necesario cargarla con input). A continuación, programar un bucle para listar por pantalla dichas materias, cada materia en una línea.

TP02-02 | MES NÚMERO A TEXTO

Escribir una función que reciba un número de mes y devuelva una cadena con el nombre del mes.

Probar la función desde un programa principal con un input para la entrada del número de mes, luego la llamada a la función con dicho número como argumento, y finalmente un print de lo que la función devuelve.

TP02-03 | ELEMENTOS NUMÉRICOS

Desarrollar cada una de las siguientes funciones y escribir un programa que permita verificar su funcionamiento, imprimiendo lo que devuelve cada función luego de invocar a cada una de ellas:

a. Cargar una lista con números al azar de cuatro dígitos. La cantidad de elementos también será un número al azar de dos dígitos.
b. Calcular y devolver el producto de todos los elementos de la lista anterior.
c. Eliminar todas las apariciones de un valor en la lista anterior. El valor a eliminar se ingresa desde el teclado y la función lo recibe como parámetro. No utilizar listas auxiliares.
d. Determinar si el contenido de una lista cualquiera es capicúa, sin usar listas auxiliares. Un ejemplo de lista capicúa es [50, 17, 91, 17, 50].

TP02-04 | ELEMENTOS REPETIDOS

Escribir funciones para:

a. Generar una lista de N números aleatorios del 1 al 100. El valor de N se ingresa a través del teclado.
b. Recibir una lista como parámetro y devolver True si la misma contiene algún elemento repetido. La función no debe modificar la lista.
c. Recibir una lista como parámetro y devolver una nueva lista con los elementos únicos de la lista original, sin importar el orden.

Combinar estas tres funciones en un mismo programa.

TP02-05 | ELEMENTOS AL CUADRADO

Crear una lista con los cuadrados de los números entre 1 y N (ambos incluidos), donde N se ingresa desde el teclado. Luego se solicita imprimir los últimos 10 valores de la lista.

TP02-06 | ELEMENTOS ELIMINADOS

Eliminar de una lista de números enteros aquellos valores que se encuentren en una segunda lista. Imprimir la lista original, la lista de valores a eliminar y la lista resultante. La función deben modificar la lista original sin crear una copia modificada.

TP02-07 | VERIFICAR ORDEN DE ELEMENTOS

Escribir una función que reciba una lista como parámetro y devuelva True si la lista está ordenada en forma ascendente o False en caso contrario. Por ejemplo, ordenada([1, 2, 3]) retorna True y ordenada(['b', 'a']) retorna False. Desarrollar además un programa para verificar el comportamiento de la función.

TP02-08 | LISTA NORMALIZADA

Escribir una función que reciba una lista de números enteros como parámetro y la normalice, es decir que todos sus elementos deben sumar 1.0, respetando las proporciones relativas que cada elemento tiene en la lista original. Desarrollar también un programa que permita verificar el comportamiento de la función. Por ejemplo, normalizar([1, 1, 2]) debe devolver [0.25, 0.25, 0.50].

TP02-09 | INTERCALACIÓN DE ELEMENTOS

Intercalar los elementos de una lista entre los elementos de otra. La intercalación podrá realizarse utilizando el método insert o mediante la técnica de rebanadas (slicing), y nunca se creará una lista nueva, sino que se modificará la primera. Por ejemplo, si lista1 = [8, 1, 3] y lista2 = [5, 9, 7], lista1 deberá quedar como [8, 5, 1, 9, 3, 7]. Las listas pueden tener distintas longitudes.

TP02-10 | ELEMENTOS IMPARES

Utilizar la técnica de listas por comprensión para construir una lista con todos los números impares comprendidos entre 100 y 200.

TP02-11 | AZAR Y ELEMENTOS IMPARES

Generar una lista con números al azar entre 1 y 100 y crear una nueva lista con los elementos de la primera que sean impares. Imprimir las dos listas por pantalla.

- Programa TP02-11A: El proceso deberá realizarse utilizando listas por comprensión.
- Programa TP02-11B: El proceso deberá realizarse utilizando la función incorporada filter(). (investigarla)

TP02-12 | ATENCIÓN DE PACIENTES EN CLÍNICA

Resolver el siguiente problema, diseñando las funciones a utilizar:

Una clínica necesita un programa para atender a sus pacientes. Cada paciente que ingresa se anuncia en la recepción indicando su número de afiliado (número entero de 4 dígitos) y además indica si viene por una urgencia (ingresando un 0) o con turno (ingresando un 1). Para finalizar se ingresa -1 como número de socio. Luego se solicita:

a. Mostrar un listado de los pacientes atendidos por urgencia y un listado de los pacientes atendidos por turno en el orden que llegaron a la clínica.
b. Realizar la búsqueda de un número de afiliado e informar cuántas veces fue atendido por turno y cuántas por urgencia. Repetir esta búsqueda hasta que se ingrese -1 como número de afiliado.

TP02-13 | REGISTRO DE VISITAS DE SOCIOS

Resolver el siguiente problema, utilizando funciones:

Se desea llevar un registro de los socios que visitan un club cada día. Para ello, se ingresa el número de socio de cinco dígitos hasta ingresar un cero como fin de carga.

Se solicita:

a. Informar para cada socio, cuántas veces ingresó al club (cada socio debe aparecer una sola vez en el informe).
b. Solicitar un número de socio que se dio de baja del club y eliminar todos sus ingresos. Mostrar los registros de entrada al club antes y después de eliminarlo. Informar cuántos ingresos se eliminaron.

TRABAJO PRÁCTICO 03 | MATRICES

TP03-01 | FUNCIONES CON MATRICES Y MENÚ

Desarrollar un programa que presente el siguiente menú de opciones:

```
SELECCIONE LA OPCIÓN DEL MENÚ
1 – Generar matriz
2 – Ordenar matriz
3 – Intercambiar dos filas
4 – Intercambiar dos columnas
5 – Transponer matriz
6 – Promedio de fila
7 – Porcentaje de impares de columna
8 – Verificación de simetría diagonal principal.
9 – Verificación de simetría diagonal secundaria.
0 - Salir del programa

opción?:
```

Cada opción llamará a una función a desarrollar según las siguientes funcionalidades:

1. Cargar números enteros aleatorios de 0 a 99 en una matriz de N x N, ingresando la medida desde el teclado.
2. Ordenar en forma ascendente cada una de las filas de la matriz.
3. Intercambiar dos filas, cuyos números se reciben como parámetro.
4. Intercambiar dos columnas dadas, cuyos números se reciben como parámetro.
5. Trasponer la matriz sobre sí misma. (intercambiar cada elemento Aij por Aji)
6. Calcular el promedio de los elementos de una fila, cuyo número se recibe como parámetro.
7. Calcular el porcentaje de elementos con valor impar en una columna, cuyo número se recibe como parámetro.
8. Determinar si la matriz es simétrica con respecto a su diagonal principal.
9. Determinar si la matriz es simétrica con respecto a su diagonal secundaria.
0. Salir del programa usando exit()

Para operar el programa siempre primero se elegirá la opción 1 que llamará a una función para generar la matriz de trabajo. La matriz de trabajo debe quedar en el ámbito global del programa para que pueda servir de argumento para otras funciones. En el ámbito global también se debe crear una copia de la matriz de trabajo para mantener la matriz original sin alteraciones. Al elegir la opción 1 se debe mostrar por pantalla la matriz generada.

Luego, al elegir cualquiera de las demás opciones del menú, se solicitarán datos de ser necesario, y luego se llamará a la correspondiente función, para finalmente presentar la matriz original junto al resultado de la función invocada para poder comprobar los cambios. Se deberá esperar a presionar ENTER para que vuelva a aparecer el menú de opciones, codificando para esto input("Presione ENTER para continuar.")

NOTA: No incluir instrucciones “input” ni “print” dentro de las funciones.

TP03-02 | GENERACIÓN DE MATRICES CON PATRONES

Para cada una de las siguientes matrices, desarrollar una función que la genere.

Escribir un programa con un menú que invoque a cada una de ellas y muestre por pantalla la matriz obtenida. El tamaño de las matrices debe establecerse como N x N solicitando el valor por teclado, y las funciones deben servir para cualquier valor ingresado. Antes de volver al menú detener el programa y continuar con ENTER.

a:

1	0	0	0
0	3	0	0
0	0	5	0
0	0	0	7

b:

0	0	0	27
0	0	9	0
0	3	0	0
1	0	0	0

c:

4	0	0	0
3	3	0	0
2	2	2	0
1	1	1	1

d:

8	8	8	8
4	4	4	4
2	2	2	2
1	1	1	1

e:

0	1	0	2
3	0	4	0
0	5	0	6
7	0	8	0

f:

0	0	0	1
0	0	3	2
0	6	5	4
10	9	8	7

g:

1	2	3	4
12	13	14	5
11	16	15	6
10	9	8	7

h:

1	2	4	7
3	5	8	11
6	9	12	14
10	13	15	16

i:

1	2	6	7
3	5	8	13
4	9	12	14
10	11	15	16

TP03-03 | MATRIZ ALEATORIA SIN REPETICIONES

Desarrollar un programa para rellenar una matriz de N x N con números enteros al azar comprendidos en el intervalo $[0,N^2)$, de tal forma que ningún número se repita. Imprimir la matriz por pantalla.

TP03-04 | FÁBRICA DE BICICLETAS

Una fábrica de bicicletas guarda en una matriz la cantidad de unidades producidas en cada una de sus plantas durante una semana. De este modo, cada columna representa el día de la semana (Lunes columna 0, Martes columna 1, etc.) y cada fila representa a una de sus fábricas. Ejemplo:

	(Lunes) 0	(Martes) 1	(Miércoles) 2	(Jueves) 3	(Viernes) 4	(Sábado) 5
(Fábrica 1) 0	23	150	20	120	25	150
(Fábrica 2) 1	40	75	80	0	80	35
(. . .)	. . .	. . .	. . .	. . .	. . .	. . .
(Fábrica n) 3	80	80	80	80	80	80

Se solicita:

A. Crear una matriz con datos generados al azar de N fábricas durante una semana, considerando que la capacidad máxima de fabricación es de 150 unidades por día y puede suceder que en ciertos días no se fabrique ninguna. Mostrar la matriz por pantalla.
B. Mostrar la cantidad total de bicicletas fabricadas por cada fábrica.
C. Mostrar cuál es la fábrica que más produjo en un solo día (detallar día y fábrica).
D. Mostrar cuál es el día más productivo, considerando todas las fábricas combinadas.
E. Crear una lista por comprensión que contenga la menor cantidad fabricada por cada fábrica. Mostrarla.

TP03-05 | CINE DE BARRIO

Desarrollar un programa con menú que permita realizar reservas en una sala de cine de barrio de N filas con M butacas por cada fila. Desarrollar las siguientes funciones y utilizarlas en el mismo programa:

Función cargarSala: recibirá la matriz como parámetro y la cargará con valores aleatorios para simular una sala con butacas ya reservadas.

Función mostrarButacas: Mostrará por pantalla el estado de cada una de las butacas del cine. Esta función deberá ser invocada antes de que el usuario realice la reserva, y se volverá a invocar luego de la misma con los estados actualizados.

Función reservar: Deberá recibir la matriz y la butaca seleccionada, y actualizará la matriz en caso de estar disponible dicha butaca. La función devolverá True/False si logró o no reservar la butaca.

Función butacasLibres: Recibirá como parámetro la matriz y retornará cuántas butacas desocupadas hay en la sala.

Función butacasContiguas: Buscará la secuencia más larga de butacas libres contiguas en una misma fila y devolverá las coordenadas de inicio de la misma.

Al elegir cada opción de menú mostrar la sala o el dato calculado según corresponda. Antes de volver al menú detener el programa y continuar con ENTER.

TRABAJO PRÁCTICO 04 | CADENAS de CARACTERES

TP04-01 | CADENA CAPICÚA

Desarrollar una función que determine si una cadena de caracteres es capicúa, sin utilizar cadenas auxiliares ni rebanadas. Escribir además un programa que permita verificar su funcionamiento.

TP04-02 | TEXTO CENTRADO EN PANTALLA

Leer una cadena de caracteres e imprimirla centrada rellenando a izquierda y derecha con guiones para cubrir toda la pantalla. Suponer que la pantalla tiene 80 columnas. En el mismo programa hacer 3 versiones: una sin utilizar facilidades de Python, otra usando la facilidad de función o método incorporado, y otra usando la facilidad de f-strings.

TP04-03 | CONTRASEÑAS INTERCALADAS

Los números de claves de dos cajas fuertes están intercalados dentro de un número entero llamado "clave maestra", cuya longitud no se conoce. Realizar un programa para obtener ambas claves, donde la primera se construye con los dígitos ubicados en posiciones impares de la clave maestra y la segunda con los dígitos ubicados en posiciones pares. Los dígitos se numeran desde la izquierda. Ejemplo: Si clave maestra fuera 18293, la clave 1 sería 123 y la clave 2 sería 89.

TP04-04 | RELOJ

Desarrollar un programa que pida un valor de hora, un valor de minuto, y un valor de segundo. A partir de esos valores mostrar un reloj digital en formato de display HH:MM:SS (cada valor siempre en 2 dígitos). El display deberá avanzar cada 1 segundo como cualquier reloj digital (es decir que cuando los segundos superen los 59 volverán a 00 y se agregará un minuto, etc. Y lo mismo entre los minutos y las horas)

TP04-05 | TEMPORIZADOR

Desarrollar un programa que pida un valor de minuto, y un valor de segundo. A partir de esos valores mostrar un reloj digital en formato de display MM:SS (cada valor siempre en 2 dígitos). El display deberá ir en cuenta regresiva cada 1 segundo hasta llegar a 00:00. Cuando llegue a cero deberá detenerse y mostrar el mensaje “<<<< TIEMPO >>>>”

TP04-06 | FILTRADO DE PALABRAS

Escribir una función filtrarPalabras() que reciba una cadena de caracteres conteniendo una frase y un entero N, y devuelva otra cadena con las palabras que tengan N o más caracteres de la cadena original. Escribir también un programa para verificar el comportamiento de la misma. Hacer tres versiones de la función, para cada uno de los siguientes casos:

A. Utilizando ciclos normales y slicing. Sin utilizar el método split()
B. Utilizando el método split() y ciclos normales

C. Utilizando el método split() y listas por comprensión

TP04-07 | PUNTO CADA 3 DÍGITOS

Escribir una función que reciba una cadena que contiene un número entero de muchos dígitos y devuelva una cadena con el mismo número, pero con los puntos de las separaciones de miles. Por ejemplo, si recibe 1234567890, debe devolver 1.234.567.890

TP04-08 | FUNCIÓN TOMAR SUBCADENA

Desarrollar una función que extraiga una subcadena de una cadena de caracteres, indicando la posición y la cantidad de caracteres deseada. Devolver la subcadena como valor de retorno. Escribir también un programa para verificar el comportamiento de la misma. Ejemplo, dada la cadena "El número de teléfono es 4356-7890" extraer la subcadena que comienza en la posición 25 y tiene 9 caracteres, resultando la subcadena "4356-7890". Escribir una función para cada uno de los siguientes casos:

A. Utilizando rebanadas
B. Sin utilizar rebanadas

TP04-09 | FUNCIÓN ELIMINAR SUBCADENA

Escribir una función para eliminar una subcadena de una cadena de caracteres, a partir de una posición y cantidad de caracteres dadas, devolviendo la cadena resultante. Escribir también un programa para verificar el comportamiento de la misma. Escribir una función para cada uno de los siguientes casos:

A. Utilizando rebanadas
B. Sin utilizar rebanadas

TP04-10 | PALABRAS DE FRASE ORDENADAS

Escribir una función que reciba como parámetro una cadena de caracteres en la que las palabras se encuentran separadas por uno o más espacios. Devolver otra cadena con las palabras ordenadas alfabéticamente, dejando un espacio entre cada una.

TP04-11 | REEMPLAZO DE PALABRA

Desarrollar una función para reemplazar todas las apariciones de una palabra por otra en una cadena de caracteres y devolver la cadena obtenida y un entero con la cantidad de reemplazos realizados. Tener en cuenta que sólo deben reemplazarse palabras completas, y no fragmentos de palabras. Escribir también un programa para verificar el comportamiento de la función.

TP04-12 | VOCALES ARRIBA

Se está desarrollando una importante app para tratamiento de texto y nos piden que desarrollemos una función para una de las opciones de la app. La función consiste en poner en mayúscula todas las vocales de una frase, por ejemplo, si la función recibe el texto “frase de prueba para el nuevo programa de tratamiento de texto” debe devolver “frAsE dE prUEbA pArA El nUEvO prOgrAmA dE trAtAmIEntO dE tExtO”. Probar la función desde un programa principal

TP04-13 | BARAJA ESPAÑOLA

Escribir un programa para crear mediante listas por comprensión los naipes de la baraja española de 48 cartas. La lista debe contener cadenas de caracteres. Ejemplo: ["1 de Oros", "2 de Oros"...]. Imprimir la lista por pantalla. (investigar en internet el tema “python listas por comprensión producto cartesiano de dos listas”)

TP04-14 | VERIFICACIÓN DE DIRECCIÓN DE EMAIL

Se solicita crear un programa para leer direcciones de correo electrónico y verificar si representan una dirección válida. Por ejemplo usuario@dominio.com.ar. Para que una dirección sea considerada válida el nombre de usuario debe poseer

solamente caracteres alfanuméricos, la dirección contener un solo carácter @, el dominio debe tener al menos un carácter y tiene que finalizar con ".com.ar"

Repetir el proceso de validación hasta ingresar una cadena vacía. Al finalizar mostrar un listado de todos los dominios, sin repetirlos y ordenados alfabéticamente, recordando que las direcciones de mail no son case sensitive.

TP04-15 | NUMERO A LETRAS

Muchas aplicaciones financieras requieren que los números sean expresados también en letras. Por ejemplo, el número 2153 puede escribirse como "dos mil ciento cincuenta y tres". Nos piden un programa con una función para convertir a texto un número entero entre 0 y casi mil millones (es decir, cifras hasta 999.999.999).

Revisando módulos desarrollados en el pasado por nuestro equipo de programadores encontramos una función para convertir números a texto, pero preparada para valores entre 0 y 999.999 (el profesor entregará esta función). Analizar la función recuperada y realizarle los cambios necesarios para que funcione hasta 999.999.999

TRABAJO PRÁCTICO 05 | EXCEPCIONES

TP05-01 | CADENAS A REALES CON MANEJADOR DE EXCEPCIONES

Realizar una función que reciba como parámetros dos cadenas de caracteres conteniendo números reales, sume ambos valores y devuelva el resultado como un número real. Devolver -1 si alguna de las cadenas no contiene un número válido, utilizando manejo de excepciones para detectar el error.

TP05-02 | MES NÚMERO A TEXTO CON MANEJADOR DE EXCEPCIONES

Desarrollar una función que devuelva una cadena de caracteres con el nombre del mes cuyo número se recibe como parámetro. Los nombres de los meses deberán obtenerse de una lista de cadenas de caracteres inicializada dentro de la función. Devolver una cadena vacía si el número de mes es inválido. La detección de meses inválidos deberá realizarse a través de excepciones.

TP05-03 | STOP DE CÓDIGO CON MANEJADOR DE EXCEPCIONES

Todo programa Python es susceptible de ser interrumpido mediante la pulsación de las teclas *Ctrl-C*, lo que genera una excepción del tipo *KeyboardInterrupt*. Realizar un programa para imprimir los números enteros entre 1 y 100000, y que solicite confirmación al usuario antes de detenerse cuando se presione Ctrl-C.

TP05-04 | RAIZ CUADRADA CON MANEJADOR DE EXCEPCIONES

La raíz cuadrada de un número puede obtenerse mediante la función *sqrt()* del módulo *math*. Escribir un programa que utilice esta función para calcular la raíz cuadrada de un número cualquiera ingresado a través del teclado. El programa debe utilizar manejo de excepciones para evitar errores si se ingresa un número negativo.

TP05-05 | INDEX CON MANEJADOR DE EXCEPCIONES

El método *index* permite buscar un elemento dentro de una lista, devolviendo la posición que éste ocupa. Sin embargo, si el elemento no pertenece a la lista se produce una excepción de tipo *ValueError*. Desarrollar un programa que cargue una lista con números enteros ingresados a través del teclado (terminando con -1) y permita que el usuario ingrese el valor de algunos elementos para visualizar la posición que ocupan, utilizando el método *index*. Si el número no pertenece a la lista se imprimirá un mensaje de error y se solicitará otro para buscar. Abortar el proceso al tercer error detectado. No utilizar el operador *in* durante la búsqueda.

TP05-06 | ADIVINAR NÚMERO CON MANEJADOR DE EXCEPCIONES

Escribir un programa que juegue con el usuario a adivinar un número. El programa debe generar un número al azar entre 1 y 500 y el usuario debe adivinarlo. Para eso, cada vez que se introduce un valor se muestra un mensaje indicando si el número que tiene que adivinar es mayor o menor que el ingresado. Cuando consiga adivinarlo, se debe imprimir en

pantalla la cantidad de intentos que le tomó hallar el número. Si el usuario introduce algo que no sea un número se mostrará un mensaje en pantalla y se lo contará como un intento más.

TRABAJO PRÁCTICO 06 | ARCHIVOS

TP06-01 | APELLIDOS POR PAÍS

Escribir un programa que lea un archivo de texto conteniendo un conjunto de apellidos y nombres en formato "Apellido, Nombre" y guarde en el archivo ARMENIA.TXT los nombres de aquellas personas cuyo apellido terminan con la cadena "IAN", en el archivo ITALIA.TXT los terminados en "INI" y en el archivo ESPAÑA.TXT los terminados en "EZ". Descartar el resto. Ejemplo:

Arslanian, Gustavo –> ARMENIA.TXT
Rossini, Giuseppe –> ITALIA.TXT
Pérez, Juan –> ESPAÑA.TXT
Smith, John –> descartar
El archivo de entrada puede ser creado mediante el Block de Notas o Notepad++. No escribir un programa para generarlo.

TP06-02 | INSCRIBIR DEPORTISTAS

Una institución deportiva necesita clasificar a sus atletas para inscribirlos en los próximos Juegos Panamericanos. Para eso encargó la realización de un programa que incluya las siguientes funciones:

GrabarRangoAlturas() Graba en un archivo las alturas de los atletas de distintas disciplinas, los que se ingresan desde el teclado. Cada dato se debe grabar en una línea distinta. Ejemplo:

<Deporte 1>
<altura del atleta 1>
<altura del atleta 2>
< . . . >
<Deporte 2>
<altura del atleta 1>
<altura del atleta 2>
< . . . >

GrabarPromedio() Graba en un archivo los promedios de las alturas de los atletas presentes en el archivo generado en el paso anterior. La disciplina y el promedio deben grabarse en líneas diferentes. Ejemplo:

<Deporte 1>
<Promedio de alturas deporte 1>
<Deporte 2>
<Promedio de alturas deporte 2>
< . . . >

MostrarMasAltos() Muestra por pantalla las disciplinas deportivas cuyos atletas superan la estatura promedio general. Obtener todos los datos del archivo de promedios.

TP06-03 | ELIMINAR COMENTARIOS

Desarrollar un programa para eliminar todos los comentarios de un programa escrito en lenguaje Python. Tener en cuenta que los comentarios comienzan con el signo # (siempre que éste no se encuentre encerrado entre comillas simples o dobles) y que también se considera comentario a las cadenas de documentación (docstrings).

TP06-04 | REGISTRO DE PRECIPITACIONES

Escribir un programa que permita grabar un archivo los datos de lluvia caída durante un año. Cada línea del archivo se grabará con el siguiente formato:

<día>;<mes>;<lluvia caída en mm> por ejemplo 25;5;319

Los datos se generarán mediante números al azar, asegurándose que las fechas sean válidas. La cantidad de registros también será un número al azar entre 50 y 200. Por último, se solicita leer el archivo generado e imprimir un informe en formato matricial donde cada columna represente a un mes y cada fila corresponda a los días del mes. Imprimir además el total de lluvia caída en todo el año.

TP06-05 | HOTEL Y RESERVAS

Un hotel necesita un programa para gestionar la operación de sus habitaciones. El hotel cuenta con 10 pisos y 6 habitaciones por piso. Por cada huésped o grupo familiar que se aloja en el mismo se registra la siguiente información:

- DNI del cliente (número entero)
- Apellido y Nombre
- Fecha de ingreso (DDMMAAAA)
- Fecha de egreso (DDMMAAAA)
- Cantidad de ocupantes

Se solicita desarrollar un programa para realizar las siguientes tareas:

- Registrar el ingreso de huéspedes al hotel, hasta que se ingrese un número de DNI -1. Esta información deberá grabarse en un archivo CSV donde cada registro incluirá todos los campos indicados más arriba. Tener en cuenta que los números de DNI no pueden repetirse y que la fecha de salida debe ser mayor a la de entrada.
- Finalizado el ingreso de huéspedes se solicita:
    - Leer el archivo de huéspedes y asignar las habitaciones a cada uno. El piso y habitación son asignados arbitrariamente, y no puede asignarse una habitación ya otorgada.
    - Mostrar el piso con mayor cantidad de habitaciones ocupadas.
    - Mostrar cuántas habitaciones vacías hay en todo el hotel.
    - Mostrar el piso con mayor cantidad de personas.
    - Mostrar cuál será la próxima habitación en desocuparse. La fecha actual se ingresa por teclado. Mostrar todas las que correspondan.
    - Mostrar un listado de todos los huéspedes registrados en el hotel, ordenado por cantidad de días de alojamiento.

TP06-06 | NÓMINA DE EMPLEADOS

Se dispone de dos formatos diferentes de archivos de texto en los que se almacenan datos de empleados, detallados más abajo. Desarrollar un programa para cada uno de los formatos suministrados, que permitan leer cada uno de los archivos y grabar los datos obtenidos en otro archivo de texto con formato CSV.

Los archivos de entrada pueden ser creados mediante el Block de Notas o Notepad++. Ambos archivos tienen tres campos por registro: Apellido y Nombre, Fecha de alta y Domicilio.

```
Formato 1: Los campos tienen longitud fija con un espacio de separación
entre ellos.

(Regla)   1         2         3         4         5         6
01234567890123456789012345678901234567890123456789012345678901234567890123

Pérez Juan       20080211 Corrientes 348
González Ana M   20080115 Juan de Garay 1111 3er piso dto A

Formato 2: Todos los campos de este archivo están precedidos por un
número de dos dígitos que indica la longitud del campo que sigue.

10Pérez Juan0820080211 14Corrientes 348
14González Ana M0820080115 33Juan de Garay 1111 3er piso dto A
```

NOTA 1: Los espacios que se encuentren al final de las cadenas en el archivo 1 deberán ser eliminados.

NOTA 2: El formato 2 debe generalizarse para que soporte registros con cualquier cantidad de campos.