FACULTAD DE INGENIERIA Y CIENCIAS EXACTAS
Departamento de Tecnología Informática

FUNDAMENTOS DE INFORMÁTICA

Guía de Trabajos Prácticos

v12.5 – 24.05.22

Objetivos: Que el alumno pueda:

- Entender y saber construir un programa
- Expresar un programa en términos estructurados
- Diferenciar conceptualmente los distintos tipos de estructuras de control
- Plantear estrategias de resolución de problemas
- Identificar módulos que puedan resolverse mediante procedimientos
- Utilizar estructuras de datos básicas

Bibliografía sugerida:
Básica

- **Joyanes Aguilar, Luis:** Fundamentos de Programación. Editorial Mc.Graw-Hill. ISBN 978-84-481-6111-8. EAN 9788448161118.

Complementaria

- **Pilgrin, Mark:** Inmersión en Python 3 [en línea] Traducido por José Miguel González Aguilera. Disponible gratuitamente (bajo licencia Creative Commons 3.0). [Revisado 06/07/21] http://www.jmgaguilera.com/libro/python/traducci%C3%B3n/latex/2016/08/19/inmersion-python.html
- **Marzal Varó, Andrés - Gracia Luengo, Isabel, García Sevilla, Pedro:** Introducción a la programación con Python 3 [En línea] Disponible gratuitamente bajo licencia Creative Commons. [Revisado: 06/07/21] https://openlibra.com/es/book/introduccion-a-la-programacion-con-python-3
- The **Python Language Reference.** [Revisado: 06/07/21]. Español: https://docs.python.org/es/3/tutorial/index.html
- **Documentos varios** en Python.org.ar [Revisado: 06/07/21]. Inglés / Español: https://wiki.python.org.ar/aprendiendopython/

Autores:
Ricardo Thompson, Patricia Mazzitelli, Felipe D'Aquino, Verónica Galati
Revisión y actualizaciones:
Equipo docente de la cátedra

Trabajo Práctico 1: Algoritmos elementales

Objetivos:

- Resolver problemas creando algoritmos.
- Definir la estructura básica de un algoritmo: Entrada-Proceso-Salida.
- Conocer el modelo Top-Down.

Ejercicio 1: Describir la secuencia de acciones necesarias para realizar el objetivo planteado. Identificar el estado inicial y el estado final en cada situación:

a. Enviar un mensaje de voz por Whatsapp.
b. Cruzar una calle en una esquina con semáforos..
c. Lavarse las manos.
d. Preparar un mate.
e. Realizar una publicación en Instagram.
f. Destapar una botella de cerveza.
g. Poner una alarma en el reloj del celular.
h. Resolver la siguiente operación: 4 + 2 * 3.
i. Pintar una pared.
j. Comprar una entrada de cine por Internet.
k. Separar el cuatro de copas de un conjunto de naipes.
l. Buscar la menor carta de un mazo de naipes, que puede no estar completo.
m. Ordenar un conjunto de naipes.

Trabajo Práctico 2: Estructura secuencial

Objetivos:

- Conocer los distintos tipos de datos y su almacenamiento en variables.
- Identificar los operadores aritméticos y el orden de precedencia al realizar cálculos básicos.
- Crear los primeros algoritmos en pseudocódigo o diagrama de flujo con estructura secuencial.
- Codificar el algoritmo en lenguaje Python.

Ejercicio 1: Calcular el valor de las siguientes expresiones, respetando el orden de operaciones establecido.

a. 12 * 4 + 4 * 5
b. (12 * (1 - 5) + 4) * 3
c. 12 * 1 - 5 + 4 * 3
d. (17 - 2) / 5
e. 3 + 2 * 2 - (8 * 4 + 1 / 2.0) * 3
f. 5 * 4 / 2
g. 5 * (4 / 2)
h. 24 / 2 ** 2
i. (24 / 2) ** 2
j. 3 + 4 * 6 / 2 – 5
k. 3 + 4 * 6 / (2 – 5)
l. (- 0.1) * 3
m. – 9 ** 2
n. (– 9) ** 2
o. 10 / 3
p. 10 // 3
q. 12 % 5

Ejercicio 2: Desarrollar un programa que permita ingresar dos números enteros A y B a través del teclado. Imprimir su suma y su diferencia.

Ejercicio 3: Calcular el promedio de las notas que obtiene un alumno al rendir los dos parciales.

Ejercicio 4: Escribir un programa que permita ingresar la edad de una persona en años y la convierta a días, imprimiendo el resultado. Considerar que todos los años tienen 365 días.

Ejercicio 5: Tres personas invierten dinero para fundar una empresa (no necesariamente en partes iguales). Calcular qué porcentaje invirtió cada una.

Ejercicio 6: Ingresar tres números enteros, calcular su promedio y mostrarlo por pantalla.

Ejercicio 7: Una persona invierte su capital en un banco y desea saber cuánto dinero ganará en un mes, teniendo en cuenta que el banco paga 2% mensual. ¿Cuánto ganará en seis meses si deja su dinero invertido?

Ejercicio 8: Leer una medida en metros e imprimir esta medida expresada en centímetros, pulgadas, pies y yardas. Los factores de conversión son:

- 1 pie = 12 pulgadas
- 1 yarda = 3 pies
- 1 pulgada = 2,54 cm.
- 1 metro = 100 cm.

Ejercicio 9: Una inmobiliaria paga a sus vendedores un salario de $50000, más una comisión de $5000 por cada venta realizada, más el 5% del valor de las ventas. Realizar un programa que imprima el número del vendedor y el salario que le corresponde en un determinado mes. Se leen el número del vendedor, la cantidad de ventas que realizó y el valor total de las mismas.

Ejercicio 10: Leer un período en segundos e imprimirlo expresado en días, horas, minutos y segundos. Por ejemplo, 200000 segundos equivalen a 2 días, 7 horas, 33 minutos y 20 segundos.

Ejercicio 11: Un banco necesita para sus cajeros automáticos un programa que lea una cantidad de dinero e imprima a cuántos billetes equivale, considerando que existen billetes de $1000, $500, $100, $50, $10, $5 y $1. Desarrollar dicho programa de tal forma que se minimice la cantidad de billetes entregados por el cajero.

Ejercicio 12: Escribir un programa para convertir un número binario de 4 cifras en un número decimal. El número binario se ingresa como un solo número entero de cuatro dígitos.

Procedimiento: Para convertir un número binario a decimal es necesario multiplicar el valor de cada dígito por el número 2 elevado a un exponente. Este exponente se obtiene de la posición que ocupa el dígito dentro del número, comenzando desde la derecha con la posición 0. Todos estos resultados se suman para obtener el valor final. Ejemplo: Convertir 1011 a decimal:

$$1_3\ 0_2\ 1_1\ 1_0 = 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0 = 11$$

Trabajo Práctico 3: Estructura condicional

Objetivos:

- Crear algoritmos en pseudocódigo o diagrama de flujo combinando las estructuras secuencial y condicional.
- Codificar los algoritmos en lenguaje Python.

Ejercicio 1: Ingresar dos números enteros e indicar si son iguales o distintos.

Ejercicio 2: Leer un número entero e imprimir un mensaje indicando si es par o impar.

Ejercicio 3: Desarrollar un programa que solicite un número de mes (por ejemplo 4) y escriba el nombre del mes en letras ("abril"). Verificar que el mes sea válido y mostrar un mensaje de error en caso de que no lo sea.

Ejercicio 4: Ingresar las notas de los dos parciales de un alumno e indicar si promocionó, aprobó o si debe recuperar. Informar un error si el valor de alguna nota no está entre 0 y 10.

- Se promociona cuando las notas de ambos parciales son mayores o iguales a 7.
- Se aprueba cuando las notas de ambos parciales son mayores o iguales a 4.
- Se debe recuperar cuando al menos una de las dos notas es menor a 4.

Ejercicio 5: Una editorial determina el precio de un libro según la cantidad de páginas que contiene. El costo básico del libro es de $500, más $3,20 por página con encuadernación rústica. Si el número de páginas supera las 300 la encuadernación debe ser en tela, lo que incrementa el costo en $200. Además, si el número de páginas sobrepasa las 600 se hace necesario un procedimiento especial de encuadernación que incrementa el costo en otros $336. Desarrollar un programa que calcule el costo de un libro dado el número de páginas.

Ejercicio 6: Una remisería requiere un programa que calcule el precio de un viaje a partir de la cantidad de kilómetros que desea recorrer el usuario. Para eso cuenta con la siguiente tarifa:

- Viaje mínimo $250. Sólo se cobra cuando el importe por kilómetro no alcanza este mínimo.
- Si recorre entre 0 y 10 km: $30 por km
- Si recorre 10 km o más: $20 por km

Ejercicio 7: Leer un número correspondiente a un año e imprimir un mensaje indicando si es bisiesto o no. Un año es bisiesto cuando es divisible por 4. Sin embargo, aquellos años que sean divisibles por 4 y también por 100 no son bisiestos, a menos que también sean divisibles por 400. Por ejemplo, 1900 no fue bisiesto pero sí el 2000.

Ejercicio 8: Leer tres números correspondientes al día, mes y año de una fecha e imprimir un mensaje indicando si la fecha es válida o no.

Ejercicio 9: Diseñar un programa que calcule y muestre el sueldo neto de un empleado en base a su sueldo básico y su antigüedad en años. Si es soltero se le incrementa el sueldo en 5% del salario bruto por cada año de antigüedad, mientras que si es casado se le incrementa el sueldo en 7% del bruto por cada año de antigüedad. También se le realizan los siguientes descuentos: Jubilación: 11%, Obra Social: 3%, Sindicato: 3%

Como datos de entrada se ingresa por teclado el sueldo básico, antigüedad y estado civil (1 si es soltero o 2 si es casado). Se debe informar: (reemplazar los 9 por los valores que correspondan)

Estado Civil: Soltero/Casado

Sueldo básico	Antigüedad	Descuentos	Importe
$ 999.99	99 años		+ 999.99
		Jubilación - 999,99	
		Obra Social - 999,99	
		Sindicato - 999,99	

		Sueldo Neto	999,99

Trabajo Práctico 4: Estructura iterativa

Objetivos:

- Crear algoritmos en pseudocódigo o diagrama de flujo combinando las estructuras secuencial, condicional e iterativa.
- Codificar los algoritmos en lenguaje Python.

Ejercicio 1: Realizar un programa para ingresar desde el teclado un conjunto de números. Al finalizar mostrar por pantalla el primer y último valor ingresado. Finalizar la lectura con -1.

Ejercicio 2: Realizar un programa para ingresar desde el teclado un conjunto de números e informar si la cantidad de elementos es impar o par, sin utilizar contadores. Finalizar la lectura de datos con -1.

Ejercicio 3: Realizar un programa para ingresar desde el teclado un conjunto de números y mostrar por pantalla el menor y el mayor de ellos. Finalizar la lectura de datos con un valor -1.

Ejercicio 4: Desarrollar un programa que imprima la suma de los números impares comprendidos entre 42 y 176.

Ejercicio 5: Desarrollar un programa que imprima los números naturales comprendidos entre 1 y N. El valor de N se ingresa desde el teclado.

Ejercicio 6: Mostrar la tabla de multiplicar (entre 1 y 12) del número 4. ¿Cómo cambiaría el algoritmo para que el usuario pueda decidir la tabla de multiplicar a mostrar?

Ejercicio 7: Leer 10 números enteros e imprimir su promedio, el mayor valor leído y en qué posición se encontraba. Si se ingresó más de una vez sólo debe informar la primera.

Ejercicio 8: Ingresar números, hasta que la suma de los números pares supere 100. Mostrar cuántos números se ingresaron en total.

Ejercicio 9: Se desea analizar cuántos autos circulan con patente con numeración par y cuántos con numeración impar en un día. Escribir un programa que permita ingresar la terminación de la patente (entre 0 y 9) hasta ingresar -1 e informe cuántos vehículos pasaron con numeración par y cuántos con numeración impar.

Ejercicio 10: El factorial de un número entero N mayor que cero se define como el producto de todos los enteros X tales que 0 < X <= N. Desarrollar un programa para calcular el factorial de un número dado. Deberán rechazarse las entradas inválidas (menores que 0).

Ejercicio 11: Realizar un programa que lea un número natural H e imprima un mensaje indicando si H es primo o no. Se dice que un número es primo cuando sólo es divisible por sí mismo y por la unidad.

Ejercicio 12: La sucesión de Fibonacci es una sucesión de números enteros donde cada término se obtiene como suma de los dos anteriores, siendo los dos primeros 0 y 1. Por lo tanto, Fib=0, 1, 1, 2, 3, 5, 8, 13, 21.... Realizar un programa que lea N e imprima los N primeros términos de esta sucesión, como así también la suma de los mismos.

Trabajo Práctico 5: Ejercicios integradores

Objetivos:

- Plantear estrategias de resolución de problemas creando algoritmos.
- Entender y lograr construir un programa estructurado.
- Diferenciar los distintos tipos de estructuras de control

En los ejercicios siguientes es necesario probar el programa con al menos tres conjuntos de datos de entrada, detallando los resultados obtenidos en cada caso.

Ejercicio 1: Leer números que representan edades de un grupo de personas, finalizando la lectura cuando se ingrese el número -1. Imprimir cuántos son menores de 18 años, cuántos tienen 18 años o más y el promedio de edad de ambos grupos. Descartar valores que no representan una edad válida. (Se considera válida una edad entre 0 y 100).

Ejercicio 2: Escribir un programa que permita ingresar los números de legajo de los alumnos de un curso y su nota de examen final. El fin de la carga se determina ingresando un -1 en el legajo. Informar para cada alumno si aprobó o no el examen considerando que se aprueba con nota mayor o igual a 4. Se debe validar que la nota ingresada sea entre 1 y 10. Terminada la carga de datos, informar:

- Cantidad de alumnos que aprobaron con nota mayor o igual a 4.
- Cantidad de alumnos que desaprobaron el examen con nota menor a 4.
- Porcentaje de alumnos que están aplazados (tienen 1 en el examen).

Ejercicio 3: Una empresa aplica el siguiente procedimiento en la comercialización de sus productos:

- Aplica el precio base a la primera docena de unidades.
- Aplica un 10% de descuento a todas las unidades entre 13 y 100.
- Aplica un 25% de descuento a todas las unidades por encima de las 100.

Por ejemplo, supongamos que vende 230 unidades de un producto cuyo precio base es 100. El cálculo resultante sería:

100 * 12 + 90 * 88 + 75 * 130 = 18870
y el precio promedio es de 18870/230 = 82.04

Escribir un programa que lea la cantidad solicitada de un producto y su precio base, y muestre el valor total de la venta y el precio promedio por unidad. El fin de carga se determina ingresando -1 como cantidad solicitada. Al finalizar informar:

- Cantidad de ventas realizadas total.
- Cantidad de ventas en las que se aplicó un 10% de descuento.
- Cantidad de ventas en las que SOLO se aplicó el precio base, es decir que no se le realizaron descuentos.

Ejercicio 4: Una empresa factura a sus clientes el último día de cada mes. Si el cliente paga su factura dentro de los primeros 10 días del mes siguiente, tiene un descuento de $200 o del 2% de la factura, lo que resulte más conveniente para el cliente. Si paga en los siguientes 10 días del mes deberá pagar el importe original de la factura, mientras que si paga después del día 20 deberá abonar una multa de $350 o del 10% de su factura, lo que resulte mayor. Escribir un programa que lea el número del cliente y el total de la factura, y emita un informe donde conste el número del cliente y los tres importes que podría abonar según la fecha de pago.

Ejercicio 5: Leer tres números D, M y A correspondientes al día, mes y año de una fecha, y un número entero N que representa una cantidad de días. Realizar un programa que imprima la nueva fecha que resulta de sumar N días a la fecha dada. Tener en cuenta los años bisiestos tal como se detalla en el ejercicio 7 de la práctica 3.

Ejercicio 6: Leer un número entero y mostrar un mensaje informando cuántos dígitos contiene. Tener en cuenta que el número puede ser negativo. Ejemplo: Si se ingresa 12345 se debe imprimir 5.

Ejercicio 7: Leer un número entero e invertir las cifras que contiene. Imprimir por pantalla el número invertido. Tener en cuenta que el número puede ser negativo. Por ejemplo, si se ingresa 1234 debe mostrar 4321.

Ejercicio 8: Una empresa cuenta con N empleados, divididos en tres categorías (1, 2 y 3). Por cada empleado se lee su legajo, categoría y salario. Se solicita elaborar un informe que contenga:

- Importe total de salarios pagados por la empresa.
- Cantidad de empleados que ganan más de $200000.
- Cantidad de empleados que ganan menos de $50000, cuya categoría sea 3.
- Legajo del empleado que más gana.
- Sueldo más bajo.
- Importe total de sueldos por cada categoría.
- Salario promedio

Ejercicio 9: Ingresar un número N y validar que sea positivo. Luego:

a. Mostrar los primeros números impares, hasta alcanzar el número ingresado.
b. Informar la suma de esos números impares.

Ejemplo:

- Si se ingresa 5, se debe mostrar 1 3 5 y la suma es 9.
- Si se ingresa 8, se debe mostrar 1 3 5 7 y la suma es 16.
- Si se ingresa -5, se debe pedir otro número.

Trabajo Práctico 6: Funciones

Objetivos:

- Comprender el concepto de modularización a través de subprogramas.
- Construir programas a partir de módulos reutilizables.

Codificar funciones apropiadas para cumplir con los siguientes objetivos, escribiendo también el programa principal correspondiente:

Ejercicio 1: Escribir una función que reciba como parámetros dos números enteros. Calcular y devolver el resultado de la multiplicación de ambos valores utilizando solamente sumas. Por ejemplo 4 * 3 = 4 + 4 + 4 .

Ejercicio 2: Dados dos parámetros enteros A y B, obtener A^B (A elevado a la B) mediante multiplicaciones sucesivas, utilizando la función del ejercicio anterior para multiplicar. Por ejemplo $4^3 = 4 * 4 * 4$.

Ejercicio 3: Imprimir una columna de asteriscos, donde su altura se recibe como parámetro.

Ejercicio 4: Devolver True si el número entero recibido como primer parámetro es múltiplo del segundo, o False en caso contrario. Ejemplo: esmultiplo(40, 8) devuelve True y esmultiplo(50, 3) devuelve False.

Ejercicio 5: Desarrollar la función signo(n), que reciba un número entero y devuelva 1, -1 o 0 según el valor recibido sea positivo, negativo o nulo.

Ejercicio 6: Escribir la función comparar(a, b) que reciba como parámetros dos números enteros y devuelva 1 si el primero es mayor que el segundo, 0 si son iguales o -1 si el primero es menor que el segundo. En este ejercicio debe aprovecharse la función del ejercicio anterior. Ejemplo: comparar(4, 2) devuelve 1, y comparar(2, 4) devuelve -1.

Ejercicio 7: Calcular y devolver el Máximo Común Divisor de dos enteros no negativos, basándose en las siguientes fórmulas matemáticas:

- MCD(X,X) = X
- MCD(X,Y) = MCD(Y, X)
- Si X > Y => MCD(X, Y) = MCD(X-Y, Y).

Ejemplo: MCD(40, 15) devuelve 5.

Ejercicio 8: La raíz cuadrada de un número positivo ***n*** puede calcularse mediante la siguiente fórmula de Newton:

$$\sqrt{n} \approx \frac{\frac{n}{a} + a}{2}$$

donde ***a*** es una aproximación a ***n***. Al aplicar repetidamente esta fórmula reemplazando ***a*** por la aproximación obtenida en el paso anterior, se obtiene cada vez una aproximación mejor. Desarrollar un programa que calcule la raíz cuadrada aproximada de un número entero positivo ***n*** utilizando como primera aproximación a ***n/2***. Detener el proceso cuando la diferencia entre dos cálculos sucesivos sea menor a 0,0001.

Ejercicio 9: Escribir una función que reciba como parámetros un número de día, un número de mes y un número de año y devuelva la cantidad de días que faltan hasta fin de mes. Luego desarrollar tres programas para:

- Ingresar una fecha formada por tres enteros (día, mes y año) e imprimir la cantidad de días que faltan hasta fin de año.
- Ingresar una fecha formada por tres enteros (día, mes y año) e imprimir la cantidad de días transcurridos en ese año hasta esa fecha.
- Ingresar dos fechas formadas por tres enteros (día, mes y año) e imprimir cuánto tiempo transcurrió entre ambas, expresando el resultado en años, meses y días.

Los tres programas deben realizar su trabajo a través de la función indicada al comienzo.

Ejercicio 10: Extraer un dígito de un número. La función recibe como parámetros dos números enteros, uno será del que se extraiga el dígito y el otro indica qué cifra se desea obtener. La cifra de la derecha se considera la número 0. Retornar el valor -1 si no existe el dígito solicitado. Tener en cuenta que el número puede ser positivo o negativo. Ejemplo: extraerdigito(12345, 1) devuelve 4, y extraerdigito(12345, 8) devuelve -1.

Ejercicio 11: Obtener el dígito central de un número entero pasado como parámetro, sólo si la cantidad de dígitos es impar. Si la longitud fuera par devolver -1. Ejemplo: digitocentral(12345) devuelve 3, y digitocentral(123456) devuelve -1.

Trabajo Práctico 7: Estructuras de Datos – Listas

Objetivos:

- Introducir el concepto de estructuras de datos.
- Familiarizarse con el uso de Listas en Python, conocidas como arreglos o vectores en otros lenguajes de programación.

Desarrollar los siguientes programas o funciones, según corresponda:

Ejercicio 1: Escribir una función para ingresar desde el teclado una serie de números entre A y B y guardarlos en una lista. En caso de ingresar un valor fuera de rango la función mostrará un mensaje de error y solicitará un nuevo número. Para finalizar la carga se deberá ingresar -1. La función recibe como parámetros los valores de A y B, y devuelve la lista cargada (o vacía, si el usuario no ingresó nada) como valor de retorno. Tener en cuenta que A puede ser mayor, menor o igual a B.

En los ejercicios 2 a 6 utilizar la función del ejercicio 1 para ingresar datos en una lista y:

Ejercicio 2: Calcular la suma de los números de la lista.

Ejercicio 3: Determinar si la lista es capicúa (palíndromo). Una lista capicúa se lee de igual modo de izquierda a derecha y de derecha a izquierda. Por ejemplo, [2, 7, 7, 2] es capicúa, mientras que [2, 7, 5, 2] no lo es.

Ejercicio 4: Escribir una función para contar cuántas veces aparece un valor dentro de la lista. La función recibe como parámetros la lista y el valor a buscar, y devuelve un número entero.

Ejercicio 5: Desarrollar una función que reciba la lista como parámetro y devuelva una nueva lista con los mismos elementos de la primera, pero en orden inverso. Por ejemplo, si la función recibe [5, 7, 1] debe devolver [1, 7, 5].

Ejercicio 6: Escribir una función para devolver una lista con todas las posiciones que ocupa un valor pasado como parámetro, utilizando búsqueda secuencial en una lista desordenada. La función debe devolver una lista vacía si el elemento no se encuentra en la lista original.

Ejercicio 7: Ídem anterior, utilizando búsqueda binaria sobre una lista ordenada.

Ejercicio 8: Rellenar una lista con números enteros entre 0 y 100 obtenidos al azar e imprimir el valor mínimo y el lugar que ocupa. Tener en cuenta que el mínimo puede estar repetido, en cuyo caso deberán mostrarse todas las posiciones en las que se encuentre. La carga de datos termina cuando se obtenga un 0 como número al azar, el que no deberá cargarse en la lista.

Ejercicio 9: Crear una lista de N números generados al azar entre 0 y 100 pero sin elementos repetidos. El valor de N se ingresa por teclado. Resolver este problema utilizando dos estrategias distintas:

- Impidiendo el agregado de elementos repetidos
- Eliminando los duplicados luego de generar la lista. Asegurarse que la cantidad final de elementos sea la solicitada.

Ejercicio 10: Eliminar de una lista de números enteros los valores que se encuentren en una segunda lista. Imprimir la lista original, la lista de valores a eliminar y la lista resultante. Ambas listas deben rellenarse con números al azar. La cantidad y el rango de los valores los decide el programador.

Ejercicio 11: Cargar dos listas de números A y B con N números al azar entre 1 y 100, donde N se ingresa por teclado. Mostrar ambas listas por pantalla. Luego construir e imprimir tres nuevas listas C, D y E que contengan:

- La concatenación de los valores pares de A con los impares de B. (valores pares o valores impares se refiere a los elementos propiamente dichos y no a sus posiciones).
- La concatenación de los valores impares de A con el reverso de los valores pares de B.
- La intercalación de los elementos de A y B.

Ejercicio 12: Dada una lista ordenada de números llamada A y un nuevo número N, desarrollar un programa que agregue el elemento N dentro de la lista A, respetando el ordenamiento existente. El programa deberá detectar automáticamente si el ordenamiento es ascendente o descendente antes de realizar la inserción. No se permite añadir el elemento al final y reordenar la lista.

Ejercicio 13: Leer dos listas de números M y N y ordenarlas de menor a mayor. Luego se solicita generar e imprimir una tercera lista que resulte de intercalar los elementos de M y N. La nueva lista también debe quedar ordenada, sin utilizar ningún método de ordenamiento en ella.

Ejercicio 14: Una escuela necesita conocer cuántos alumnos cumplen años en cada mes del año, con el propósito de ofrecerles un agasajo especial en su día. Desarrollar un programa que lea el número de legajo y fecha de nacimiento (día, mes y año) de cada uno de los alumnos que concurren a dicha escuela. La carga finaliza con un número de legajo igual a -1. Emitir un informe donde aparezca -mes por mes- cuántos alumnos cumplen años a lo largo del año. Imprimir también una leyenda que indique cuál es el mes con mayor cantidad de cumpleaños.

Trabajo Práctico 8: Matrices

Objetivos:

- Conocer una nueva estructura de datos que permite relacionar la información en forma bidimensional: Las matrices.
- Implementar en Python el uso de matrices a través de listas de listas.

En los siguientes ejercicios los valores de M y N deben ingresarse a través del teclado, en los casos que corresponda:

Ejercicio 1: Crear una matriz de 3x4 (3 filas y 4 columnas) con valores obtenidos al azar entre 1 y 10. Mostrar la matriz por pantalla respetando el formato de 3 filas y 4 columnas.

Ejercicio 2: Generar una matriz cuadrada de NxN. Informar cuál es la suma de los elementos ubicados en el triángulo superior de la misma.

Ejercicio 3: Generar una matriz de MxN con valores al azar comprendidos entre A y B. Mostrar la suma de los valores ubicados sobre la diagonal principal. Los valores de A y B también se ingresan por teclado.

Ejercicio 4: Indicar las coordenadas del mayor valor encontrado en una matriz de MxN, generada con valores aleatorios entre 100 y 1000.

Ejercicio 5: Desarrollar una función para crear una matriz de MxN. La función recibe M y N por parámetro, la rellena con valores al azar entre A y B (también recibidos por parámetro) y retorna la matriz creada. Si no hay valores entre A y B o alguna de las dimensiones es negativa se deberá retornar la matriz vacía. Desarrollar también un pequeño programa principal para invocar a la función y mostrar la matriz por pantalla. Además mostrar la suma de cada fila y cada columna.

Ejercicio 6: Utilizando la función creada en el ejercicio anterior, desarrollar un programa para crear dos matrices de 3x3 con valores al azar entre dos números ingresados por teclado. Verificar que el rango sea válido, en caso contrario solicitar nuevamente ambos valores. Construir una nueva matriz donde cada elemento se obtenga como resultado de sumar los elementos correspondientes en las matrices originales. Mostrar todas las matrices y el total obtenido.

Ejercicio 7: Ingresar un número entero positivo, el que debe ser múltiplo de 4. Luego crear una matriz que contenga 4 elementos por fila, hasta completar la cantidad de elementos indicada. Mostrar la matriz e informar cuántas filas se crearon. La matriz se rellena con números al azar.

Ejercicio 8: Desarrollar una función que genere la siguiente matriz de NxN, donde N debe ser un número par:

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

Observar que el patrón de relleno se realiza por cuadrantes, utilizando números correlativos a partir de 1. Escribir también un programa para ingresar N, invocar a la función y mostrar la matriz obtenida.

Ejercicio 9: Generar una matriz cuadrada de NxN con números al azar entre 0 y 99. Luego crear una lista que contenga la suma de cada una de las filas de la matriz, sin valores repetidos. Es decir que si dos filas suman igual, el valor debe estar una sola vez en la lista. Mostrar por pantalla la matriz y la lista. Esta última debe ordenarse de menor a mayor.

Ejercicio 10: Ingresar valores desde el teclado en una matriz de MxN y mostrar la matriz creada. Ordenar cada una de las filas utilizando los distintos métodos de ordenamiento estudiados. Mostrar nuevamente la matriz.

Trabajo Práctico 9: Ejercicios tipo Final

Objetivos:

- Plantear estrategias modulares de resolución de problemas.
- Diferenciar los distintos tipos de estructuras.

Ejercicio 1: Leer los números de legajo de los alumnos de un curso y su nota de examen final. El fin de la carga se determina ingresando un -1 como legajo. Se debe validar que la nota ingresada esté entre 1 y 10. Terminada la lectura de datos, informar:

- Cantidad de alumnos que aprobaron con nota mayor o igual a 4
- Cantidad de alumnos que desaprobaron el examen. Nota menor a 4
- Promedio de nota y los legajos que superan el promedio

Luego se solicita mostrar un listado de legajos y calificaciones ordenado de manera ascendente según el número de legajo. Resolver de dos formas: Utilizando dos listas paralelas y utilizando una matriz de dos filas.

Ejercicio 2: Una Administradora de Consorcios necesita un sistema para poder gestionar el cobro de las expensas de un edificio de departamentos de 20 unidades. En dos listas almacena la siguiente información: Número de unidad y superficie en metros cuadrados. Validar que no se ingresen números de unidades duplicadas. Cada unidad paga de expensas un valor fijo por metro cuadrado, el que se ingresa por teclado. Se pide:

- Informar el promedio de expensas del mes.
- Ordenar los listados de mayor a menor según la superficie. Mostrar por pantalla el listado ordenado informando el número de unidad y la superficie en metros cuadrados.

Ejercicio 3: Hora de jugar: Desarrollar un programa que genere un número entero al azar de cuatro cifras y le proponga al usuario que lo descubra, ingresando valores repetidamente hasta hallarlo. En cada intento el programa mostrará mensajes indicando si el número ingresado es mayor o menor que el valor secreto. Permitir que el usuario abandone la partida al ingresar -1. Al terminar el juego informar la cantidad de intentos realizada, haciendo que el usuario ingrese su número de documento si mejoró la mejor marca de intentos obtenida hasta el momento. Luego mostrar la lista ordenada de los 5 mejores puntajes (indicando también a quién pertenecen) y preguntar si se desea jugar otra vez, reiniciando el juego en caso afirmativo.

Ejercicio 4: Modificar el programa anterior para que las pistas brindadas por el programa no sean del tipo "es mayor" o "es menor" sino "M dígitos correctos y N dígitos aproximados". Se considera que un dígito es correcto cuando tanto su valor como su posición coinciden con los del número secreto, mientras que un dígito es aproximado cuando coincide el valor, pero no su posición. Ejemplos:

Número secreto: 5739

- Intento 1: 1234 -> 1 correcto
- Intento 2: 5678 -> 1 correcto y 1 aproximado
- Intento 3: 9375 -> 4 aproximados

Ejercicio 5: Ingresar por teclado un número N y construir una lista llamada SECUENCIAS con N números enteros al azar entre 1 y 20. Esta lista se caracterizará porque sus valores deben encontrarse divididos en secuencias de números separadas por ceros, cuya suma no sea mayor que 20. Para eso se deberá agregar un elemento de valor 0 a fin de separar cada secuencia de la siguiente, cuidando que ninguna

secuencia sume más de 20. Agregar un 0 adicional al final de la lista y mostrar la lista obtenida por pantalla. Ejemplo:

Valor de N ingresado: 11

Números al azar generados: 5, 2, 9, 6, 4, 15, 3, 19, 12, 1, 5

Lista SECUENCIAS construida:

5	2	9	0	6	4	0	15	3	0	19	0	12	1	5	0

Ejercicio 6: A partir de la lista SECUENCIAS generada en el ejercicio anterior, imprimir la secuencia más larga almacenada en la misma. Si hubiera varias secuencias con la misma longitud máxima deberán mostrarse todas las que correspondan.