UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Intro. a Ciencia de la Computación
ORDENAMIENTO
Pregrado
2019-II
Ciencia de la Computación
Laboratorio

1. (7 points) Elabore un programa que solicite al usuario el ingreso de números enteros positivos. El ingreso de números termina cuando el usuario ingresa -1.
   El programa tiene que almacenar en una primera lista los números ingresados que sean múltiplos de 5.

   A continuación, el programa solicita el ingreso de palabras. El ingreso de palabras termina cuando se ingresa la palabra: "FIN". El programa tiene que almacenar las palabras que tengan 3 ó más caracteres en una segunda lista. Ya sea que la palabra haya sido ingresada en mayúscula o minúscula, se almacena con mayúsculas en la segunda lista.

   Utilice el algoritmo de ordenamiento Insertion Sort para ordenar la primera lista de números de menor a mayor y el mismo algoritmo para ordenar las palabras de la segunda lista en orden alfabético (A-Z).

   Algunos ejemplos de diálogo de este programa serían:

```
Ingrese n mero: 55
Ingrese n mero: 12
Ingrese n mero: 35
Ingrese n mero: 11
Ingrese n mero: 90
Ingrese n mero: 25
Ingrese n mero: 88
Ingrese n mero: -1
---------------
Ingrese palabra: sol
Ingrese palabra: su
Ingrese palabra: comida
Ingrese palabra: el
Ingrese palabra: fuego
Ingrese palabra: aire
Ingrese palabra: luz
Ingrese palabra: ok
Ingrese palabra: FIN
----------------
Lista ordenada n meros(ascendente): 25 35 55 90
Lista ordenada palabras(A-Z): AIRE COMIDA FUEGO LUZ SOL
```

```
Ingrese n mero: 34
Ingrese n mero: 15
Ingrese n mero: 17
Ingrese n mero: 45
Ingrese n mero: 40
Ingrese n mero: 66
Ingrese n mero: -1
---------------
Ingrese palabra: colmillo
Ingrese palabra: vi
Ingrese palabra: lente
Ingrese palabra: campo
Ingrese palabra: cabeza
Ingrese palabra: la
Ingrese palabra: dos
Ingrese palabra: FIN
----------------
Lista ordenada n meros(ascendente): 15 40 45
Lista ordenada palabras(A-Z): CABEZA CAMPO COLMILLO DOS
   LENTE
```

2. (7 points) Desarrolle un programa que permita el ingreso de los meses del año (validar que solo deben ingresar los meses de "enero" a "diciembre" en minúscula). Se termina de ingresar con "FIN" Utilizando el algoritmo de ordenamiento Insertion Sort, mostrar los meses en orden descendente. EJEMPLO

```
Mes: Enero
Error
Mes: FEBRERO
Error
Mes: enero
Mes: diciembre
Mes: marzo
Mes: FIN
Lista ordenada en descendente
marzo
diciembre
enero
```

3. (7 points) El Instituto Nacional de Estadística e Informática (INEI) dio a conocer que de acuerdo con los Resultados de los Censos Nacionales 2017: XII de Población, VII de Vivienda y III de Comunidades Indígenas, ejecutados el 22 de octubre del año pasado, la población de los distrito de Lima es el siguiente:

```
inei = {"Molina":175237, "Callao":426649, "Victoria":174958,
  "Comas":532403, "Agustino":194474, "Carabayllo":305963,
  "Olivos":377532, "Cieneguilla":47860, "Lince":51054,
  "Independencia":220654, "Jesus Maria":73439, "Ate":638345,
  "Carmen Legua":43156, "Mi Peru":52722, "Ventanilla
     ":356040,
  "Lima":276861, "Ancon":43951, "Barranco":30698,
  "Bellavista":78489, "La Perla":64111, "Brena":77291,
  "Chaclacayo":44271, "Chorrillos":330483, "Lince":51054 }
```

Escribe un programa que realice lo siguiente:

- Ordenar la población de cada distrito en forma descendente y guardelo en una lista.
- Imprimir la lista ordenada.
- Imprimir la población más alta.
- Imprimir la población más baja.

Para obtener una lista con la población ordenada en forma descendente, implemente y use la siguiente función:

- `ordenar_poblacion` recibe como parámetro el `diccionario inei`, y utilizando el método `Insert Sort` retorne la lista de la población ordenada en forma descendente.

Restricciones: No puede utilizar las funciones `sort, min y max`.

El resultado de la ejecución de programa es el siguiente:

```
Poblacion ordenada: [638345, 532403, 426649, 377532, 356040,
    330483, 305963, 276861, 220654, 194474, 175237, 174958,
   78489, 77291, 73439, 64111, 52722, 51054, 47860, 44271,
   43951, 43156, 30698]
Poblacion mas alta: 638345
Poblacion mas baja: 30698
```

4. (7 points) La investigadora Counterpoint Research presentó su informe sobre ventas de teléfonos móviles correspondientes al cierre de 2018, en el que se evidenció, por primera vez en una caída de 4% en las ventas de estos dispositivos móviles frente a la misma fecha en 2017, Las ventas por millones de unidades es la siguiente:

```
ventas = {"Xiaomi":96, "Oppo":119, "LG":55.9, "Lenovo":49.9,
          "Nokia":7.7, "TecnoPro":13.1, "Dell":6.1,
          "IBM":0.6, "Sahito":34.6, "Samsung":318.1,
          "Apple":215.8, "Vivo":100.2, "Huawei":153.1,
          "Aio":45.6, "FirePhone": 29.7, "Motorola":20.3 }
```

Escribe un programa que realice lo siguiente:

- Ordenar las ventas de cada empresa en forma ascendente y guardelo en una lista.
- Imprimir la lista ordenada.
- Imprimir la venta más baja.
- Imprimir la venta más alta.

Para obtener una lista ordenada de ventas en forma ascendente, implemente y use la siguiente función:

- `ordenar_ventas` recibe como parámetro el diccionario `ventas`, y utilizando el método `Insert Sort` retorne la lista ordenada de ventas en forma ascendente.

Restricciones: No puede utilizar las funciones `sort, min y max`.

El resultado de la ejecución de programa es el siguiente:

```
Ventas ordenada: [0.6, 6.1, 7.7, 13.1, 20.3, 29.7, 34.6,
   45.6, 49.9, 55.9, 96, 100.2, 119, 153.1, 215.8, 318.1]
Venta mas baja: 0.6
Venta mas alta: 318.1
```

5. (7 points) Dada la siguiente matriz de números compuestos:

```
compues = [[8, 9, 14, 15, 16, 18, 4, 6, 10, 12 ],
           [21, 22, 25, 26, 27, 20, 24],
           [34, 28, 32, 33, 30, ],
           [38, 39, 42, 44, 35, 36, 40],
           [52, 54, 45, 48, 49, 50, 51, 46 ]]
```

Escribe un programa que realice lo siguiente:

- Ordenar los números de cada lista en forma ascendente
- Imprimir las listas ordenadas.
- Imprimir el máximo número de cada lista

Para ordenar los números de cada lista en forma ascendente, implemente y use la siguiente función:

- `ordenar_matriz` recibe como parámetro la matriz`compues`, y utilizando el método `Insert Sort` ordenada las listas en forma ascendente.

Restricciones: No puede utilizar las funciones `sort, min y max`.

El resultado de la ejecución de programa es el siguiente:

```
[4, 6, 8, 9, 10, 12, 14, 15, 16, 18]
[20, 21, 22, 24, 25, 26, 27]
[28, 30, 32, 33, 34]
[35, 36, 38, 39, 40, 42, 44]
[45, 46, 48, 49, 50, 51, 52, 54]
18
27
34
44
54
```

6. (7 points) Se quiere hacer un análisis de las aplicaciones móviles más populares por algunos estudiantes. El objetivo es mostrar las aplicaciones más usadas desde las más populares hasta las menos populares. Escribe una función en Python que reciba una lista de aplicaciones y su ranking, representado por un número, y retorne el nombre de las aplicaciones más populares de forma decreciente.

   Por ejemplo:

```
Productos: [("Tinder", 100), ("Gmail", 20), ("Glovo", 50)]
Tinder Glovo Gmail
```

```
Productos: [("WhatsApp", 100), ("Instagram", 100)]
WhatsApp Instagram
```

   Consideraciones: No se considerará válido el uso de la función sort() de python o de alguna otra función que modifique listas como append() o pop(), sin embargo se puede usar la función len(). Además su algoritmo no deberá hacer uso de alguna lista o diccionario adicional al parámetro de la función.

7. (7 points) El Señor Demetrio viajó a Brasil para apoyar a la selección peruana en la Copa América 2019 por un intervalo de tiempo no definido, dejando en su casa una serie de deudas ocasionadas por los fuertes gastos de dicho viaje. A su regreso, él decide trabajar duro para amortiguar dichas deudas empezando desde la mayor hasta la menor. Para ello, se le pide a usted ayudar a don Demetrio diseñando una función que ordene las deudas de forma descendente. Las deudas estan almacenadas en una lista de tuplas.

   Plantilla del código:

```
def ordenarDeudas(deudas):
        #
        # implemente la funcion de ordenacion aqui
        #

deudas=[("luz",115),("agua",50),
        ("banco",3000),("colegio",850)]
print("Deudas ordenadas:", ordenarDeudas(deudas))
```

Salida:

```
Deudas ordenadas: [("banco",3000),("colegio",850),
                   ("luz",115),("agua",50)]
```

Restricciones: No se considerará válido el uso de la función sort() de python o de alguna otra función que modifique listas como append() o pop(), sin embargo se puede usar la función len(). Además su algoritmo no deberá hacer uso de alguna lista o diccionario adicional al parámetro de la función.

8. (7 points) Ordenar los siguientes números de menor a mayor usando cualquiera de los algoritmos estudiados en clase.

```
n meros = [10, 30, 110, 110.2, 20, 50, 2, 900, 2,13]
```

9. (7 points) Se tiene un diccionario de las edades de los asistentes a 5 salas de conferencias. Se quiere ordenar las listas para poder obsequiar presentes a los asistentes por rango de edades.

   Implemente el algoritmo de ordenacion de menor a mayor para cada una de las listas del diccionario. Complete la funcion ordenardict e implemente su propio algoritmo de ordenacion.

```
asistentes = {
  1: [43, 39, 23, 52, 21, 48, 31, 26, 55, 32],
  2: [51, 29, 47, 40, 32,25, 31, 29, 51, 36],
  3: [30, 43, 52, 23, 37, 51, 29, 50, 26, 35],
  4: [36, 44, 49, 22, 44, 49, 55, 48, 52, 51],
  5: [32, 29, 43, 32, 32, 36, 22, 48, 38, 29]
  }
 def ordenarfila(fila):
 # ToDO
 return fila

 def ordenardict( asistentes ):
    # ToDO
    return asistentes
```

```
Output esperado:
{
1: [21, 23, 26, 31, 32, 39, 43, 48, 52, 55],
2: [25, 29, 29, 31, 32,36, 40, 47, 51, 51],
3: [23, 26, 29, 30, 35, 37, 43, 50, 51, 52],
4: [22, 36, 44, 44, 48, 49, 49, 51, 52, 55],
5: [22, 29, 29, 32, 32, 32, 36, 38, 43, 48]
}
```

10. (7 points) Un simulacro de emergencia en un centro comercial a obligado a las personas a agruparse en 5 grupos. En la matriz M se encuentran las edades de cada integrante de los grupos. Ordene de menor a mayor todas las listas de edades de esta matriz. Utilice un algoritmo de ordenacion explicado en clase.

```
M = [[67, 74, 22, 48, 86, 20, 91, 69, 4, 66],
[78, 44, 70, 88, 88, 100, 58, 15,73, 26],
[74, 60, 99, 42, 90, 48, 28,85, 88, 98],
[62, 76, 29, 54, 51, 49,63, 26, 25, 57],
[26, 30, 100, 8, 98,7, 73, 19, 48, 16]]

 def ordenarfila(fila):
 # ToDO
 return fila

 def ordenarmatriz(matriz):
    # ToDO
    return matriz
```

```
output:
[[4, 20, 22, 48, 66, 67, 69, 74, 86, 91],
[15, 26, 44, 58, 70, 73, 78, 88, 88, 100],
[28, 42, 48, 60, 74, 85, 88,90, 98, 99],
[25, 26, 29, 49, 51, 54,57, 62, 63, 76],
[7, 8, 16, 19, 26, 30, 48, 73, 98, 100]]
```

11. (7 points) Se tiene un diccionario de las edades de los asistentes a 5 salas de una feria de productos agricolas. Se quiere ordenar las listas para poder obsequiar presentes a los asistentes por rango de edades.

    Implemente el algoritmo de ordenacion de mayor a menor para cada una de las listas del diccionario. Complete la funcion ordenardict e implemente su propio algoritmo de ordenacion.

```
asistentes = {
  1: [43, 39, 23, 52, 21, 48, 31, 26, 55, 32],
  2: [51, 29, 47, 40, 32,25, 31, 29, 51, 36],
  3: [30, 43, 52, 23, 37, 51, 29, 50, 26, 35],
  4: [36, 44, 49, 22, 44, 49, 55, 48, 52, 51],
  5: [32, 29, 43, 32, 32, 36, 22, 48, 38, 29]
  }
 def ordenarfila(fila):
 # ToDO
```

```
 return fila

 def ordenardict( asistentes ):
    # ToDO
    return asistentes
```

```
Output esperado:

{1: [55, 52, 48, 43, 39, 32, 31, 26, 23, 21],
2: [51, 51, 47, 40, 36, 32, 31, 29, 29, 25],
3: [52, 51, 50, 43, 37, 35, 30, 29, 26, 23],
4: [55,52, 51, 49, 49, 48, 44, 44, 36, 22],
5: [48, 43, 38, 36, 32, 32, 32, 29, 29, 22]}
```

12. (7 points) **Estrellas:** Genere una tabla de la siguiente lista de estrellas y sus propiedades ordenadas por luminosidad (L), donde $L = T^4 * R^2$, siendo T la temperatura y R el radio de la estrella

estrella	**radio**	**temperatura**
UY-Scuti	1.71	3365
Canis Mayoris	1.42	3490
KY Cyg	1.42	3500
Mu Cephei	1.26	3750
S Persei	1.20	3000
VX Sagittarii	1.12	3000

```
Ordenado por luminosidad:
VX Sagittarii , S Persei , UY-Scuti , Canis Mayoris , KY Cyg
    , Mu Cephei
```

13. (7 points) Crea una función que reciba una lista, la ordene de acuerdo a alguno de los metodos de ordenamiento vistos en clase, y también indique cuantos movimientos en la lista tuvo que realizar.

14. (7 points) Crea una función que reciba dos listas cuyo contenido sean números enteros y que te devuelva una lista con todos los elementos de las dos listas, ordenados de menor a mayor, usando alguno de los algoritmos de ordenamiento aprendidos en clase.

15. (7 points) Dado un conjunto de números leidos del archivo: `"numeros.txt"`

```
14 25
23 12
45 56
23 45
```

```
12 34
90 12
```

- Se pide leer los datos del archivo y convertirlos en un array llamado `numeros`
- Se pide ordenar los numeros de forma `ascendente` usando Insertion Sort.

Un ejemplo de diálogo de este programa sería:

```
numeros = [14, 25, 23, 12, 45, 56, 23, 45, 12, 34, 90, 12]
insertionsort(numeros)
print(numeros)
[12, 12, 12, 14, 23, 23, 25, 34, 45, 45, 56, 90]
```

16. (7 points) Diseñe e implemente una función que permita hallar el segundo mayor elemento de una lista de números. Considere los siguientes ejemplos:

```
test = [5, 4, 1, 2, 8, 3, 2, 9]
output = 8

test = [4, 1, 9, 3, 10, 6, 4, 3]
output = 9

test = [8, 3, 4, 8, 2, 1, 6, 7]
output = 8
```

17. (7 points) Diseñe e implemente una función que permita hallar el segundo menor elemento de una lista de números. Considere los siguientes ejemplos:

```
test = [5, 4, 1, 2, 8, 3, 2, 9]
output = 2

test = [4, 1, 9, 3, 10, 6, 4, 3]
output = 3

test = [8, 3, 4, 8, 2, 1, 6, 7]
output = 2
```

Considerar las siguientes restricciones:

- No se debe utilizar sorted ni sort (si requiere ordenar la lista, deberá implementar la función).
- No se deben crear listas adicionales.
- No se debe utilizar métodos que alteren el tamaño de la lista (remove, slicing, pop).
- El tamaño minimo de la lista será 2.

18. (7 points) La empresa de juegos mecánicos "Divertilandia" desea usar un programa en Python para su nueva atracción "Montaña rusa" la cual tiene 5 distintas filas. Cada fila está determinada para determinado rango de altura.

    Los rangos de las filas son los siguientes:

    - Fila 1: 50-89 cm
    - Fila 2: 90- 119 cm
    - Fila 3: 120 – 160 cm
    - Fila 4: 160 – 179 cm
    - Fila 5: 180 – 200 cm

    Crear un programa que cumpla las siguientes condiciones:

    - El usuario debe ingresar 5 números los cuales representan la altura en centímetros.
    - El mínimo de altura es 50 y el máximo 200. No se debe admitir un número que esté fuera del rango.
    - Generar una función que ordene la lista utilizado el método de inserción de mayor a menor
    - Generar una función que genere e imprima un diccionario donde se indiquen las filas y los números ingresados correspondientes a cada una de ellas.
    - Tener en cuenta que puede haber filas vacías

    Ejemplo:

```
Altura 1: 125
Altura 2:  190
Altura 3: 70
Altura 4:  95
Altura 5: 150
Resultado:  {'Fila 1':'70', 'Fila 2':'95', 'Fila 3':'125', '
   Fila 3','150', 'Fila 5','190'}
```

19. (7 points) Diseñar y crear un programa que genere una lista de datos de tamaño n con valores entre 1 y 99 generados aleatoriamente y que ordene solo los elementos en la posiciones impares de la lista.

    NOTA: El programa no debe utilizar la función sort de python, ni listas, ni diccionarios adicionales o estructuras similares.

```
    Ingrese el numero: 6
    La lista original es: [22, 3, 11, 9, 7, 2]
    La lista ordenada es: [22, 2, 11, 3, 7, 9]
```

```
Ingrese el numero: 8
La lista original es: [10, 1, 11, 8, 14, 4, 17, 5]
La lista ordenada es: [10, 1, 11, 4, 14, 5, 17, 8]
```

20. (7 points) Diseñar y crear un programa que genere una lista de datos de tamaño n con valores entre 1 y 99 generados aleatoriamente y que ordene solo los elementos en la posiciones multiplo de 3 (incluida la posicion 0) de la lista.

    NOTA: El programa no debe utilizar la función sort de python, ni listas, ni diccionarios adicionales o estructuras similares.

```
Ingrese el numero: 9
La lista original es: [5, 10, 14, 1, 11, 12, 8, 20, 10]
La lista ordenada es: [1, 10, 14, 5, 11, 12, 8, 20, 10]
```

```
Ingrese el numero: 10
La lista original es: [20, 1, 4, 30, 14, 2, 41, 5, 7, 10]
La lista ordenada es: [10, 1, 4, 20, 14, 2, 30, 5, 7, 41]
```