Guía 06

1 - Complejo de cines

Desarrollar un programa que permita procesar funciones de un complejo de cines. Por cada función se conoce: cantidad de espectadores y descuento (S/N). La carga termina cuando la cantidad de espectadores sea igual a 0 (cero).

El programa deberá:

- Calcular la recaudación total del complejo, considerando que el valor de la entrada es de $50 en los días con descuento y $75 en los días sin descuento.
- Determinar cuántas funciones con descuento se efectuaron y qué porcentaje representan sobre el total de funciones.

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'

print('COMPLEJO DE CINES')
print('*' * 80)

# Inicializar contadores y acumuladores
monto = 0
funciones = 0
funciones_dto = 0

# Primera carga de datos de corte antes del ciclo
espectadores = int(input('Espectadores (con 0 termina): '))

# Proceso repetitivo
while espectadores != 0:
    # Carga de datos que no determinan el corte del ciclo
    descuento = input('Descuento (S/N): ')
    # Definir precio
    if descuento == 'S':
        precio = 50
    else:
        precio = 75
    # Acumular monto recaudado por función
    monto = monto + (espectadores * precio)
    # Contar funciones con descuento y total de funciones
    if descuento == 'S':
        funciones_dto += 1
    funciones += 1
    # Nueva carga de datos de corte dentro del ciclo
    espectadores = int(input('Espectadores (con 0 termina): '))
```

```
# Resultados
print('*'*80)
print('Recaudación total $',monto)
if funciones != 0:
    porc = funciones_dto * 100 / funciones
else:
    porc = 0
print('Porcentaje de funciones con descuento:', porc,'%')
```

2 - Ventas por sucursal

Ingresar una serie de números por teclado que representan la cantidad de ventas realizadas en las diferentes sucursales de un país de una determinada empresa.

Los requerimientos funcionales del programa son:

- Informar la cantidad de ventas ingresadas.
- Total de ventas.
- Cantidad de ventas cuyo valor este comprendido entre 100 y 300 unidades.
- Cantidad de ventas con 400, 500 y 600 unidades.
- Indicar si hubo una cantidad de ventas inferior a 50 unidades.

Usted deberá ingresar cantidades de ventas hasta que se ingrese un valor negativo.

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'

cant_venta = 0
tot_venta = 0
cant_venta1 = 0
cont_venta400 = 0
cont_venta500 = 0
cont_venta600 = 0
hay_menor_50 = False

venta = int(input('Ingrese una cantidad de ventas (negativo para terminar): '))
while venta >= 0:
    # Permite ir contabilizando la cantidad de ventas
    cant_venta += 1
    # Permite acumular el total de ventas.
    tot_venta += venta

    # Permite contabilizar la cantidad de ventas entre 200 y 300 unidades.
    if (venta >= 100 and venta <= 300):
        cant_venta1 += 1
    # Permite contabilizar la cantidad de ventas con 400, 500 y 600 unidades.
    if venta == 400:
        cont_venta400 += 1
    if venta == 500:
        cont_venta500 += 1
    if venta == 600:
        cont_venta600 += 1

    #Determina si hubo una venta inferior a 50 unidades.
    if venta < 50:
        hay_menor_50 = True

    venta = int(input('Ingrese otra cantidad de ventas: '))
```

```
print('La cantidad de ventas ingresadas fueron: ', cant_venta)
print('El total de ventas ingresadas fue:', tot_venta)
print('La cantidad de ventas entre 200 y 300 unidades es:', cant_venta1)
print('La cantidad de ventas con 400 unidades es:', cont_venta400)
print('La cantidad de ventas con 500 unidades es:', cont_venta500)
print('La cantidad de ventas con 600 unidades es:', cont_venta600)

if hay_menor_50 == True:
    print ('Hubo al menos alguna venta con menos de 50 unidades')
else:
    print('No hubo venta con menos de 50 unidades')
```

3 - Promedio de números aleatorios

Realice un programa que permita calcular el promedio de 1000 números aleatorios generados en el rango de [0, 100000].

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'

import random

print('Calcular promedio de aleatorios')
print('=' * 80)

acumulador = 0
i = 0
while i < 1000:
    n = random.randint(0, 100000)
    acumulador += n
    i += 1
promedio = acumulador / i
print("El promedio es", promedio)

print("Fin del programa :)")
```

4 - Búsqueda de mayor

Realizar un programa que permita buscar el mayor de 10.000 números aleatorios generados en el rango de [0, 100.000].

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'

import random

print('Buscar el mayor de los valores')
print('=' * 80)

mayor = 0
i = 0
while i < 10000:
    n = random.randint(0, 100000)
    if mayor < n:
        mayor = n
    i += 1

print('El mayor valor de los 10000 numeros aleatorios es:', mayor)
print("Fin del programa :)")
```

5 - Menores y promedio

Realizar un programa que genere 5000 numeros aleatorios en el rango de [0, 100000] y que permita:

- Determinar el menor de los numeros generados en forma aleatoria.
- Calcular el valor promedio de los números menores a 10.000.

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'

import random

menor = None
i = 0
suma = 0
cont = 0
while i < 5000:
    n = random.randint(0, 100000)
    if menor is None or menor > n:
        menor = n
    if n < 10000:
        suma += n
        cont += 1
    i += 1
if cont != 0:
    p = suma / cont
else:
    p = 0
print("El menor es", menor, "y el promedio de los menores a 10000 es:", p)
```

6 - Números pares e impares

Se pide desarrollar un programa que permita leer una serie de números. La finalización de carga de datos se presenta cuando el usuario ingrese un número negativo.

Los requerimientos funcionales del programa son:

- La sumatoria de solo los números que estén comprendidos entre 50 y 100.
- Cantidad de valores pares ingresados.
- Cantidad de valores impares ingresados.
- Informar si en la carga de números se ingreso al menos un número 0.
- Informar si la serie contiene solo números pares e impares alternados

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'

print('Procesamiento de Numeros Pares e Impares')

sumatoria = 0
cantidad_pares = 0
cantidad_impares = 0
ingreso_cero = False
anterior = -1
alternan = True

numero = int(input('Ingrese un numero (fin con un numero negativo): '))
while numero >= 0:
    if 50 < numero < 100:
        sumatoria += numero

    paridad = numero % 2
    if paridad == 0:
        cantidad_pares += 1
    else:
        cantidad_impares += 1

    if numero == 0:
        ingreso_cero = True

    if anterior == paridad:
        alternan = False
    anterior = paridad
    numero = int(input('Ingrese otro numero a procesar: '))

print('Resultados')
print('=' * 80)
print('La sumatoria de los numeros entre 50 y 100 fue de', sumatoria)
print('La cantidad de numeros pares procesados fue de', cantidad_pares)
print('La cantidad de numeros impares procesados fue de', cantidad_impares)
```

```
if ingreso_cero:
    print('El usuario al menos ingreso un numero cero en la secuencia')

if alternan:
    print('La secuencia tiene numeros pares e impares alternados')
else:
    print('La secuencia no tiene numeros pares e impares alternados')
```

7 - Censo

Desarrollar un programa que permita procesar los datos del último censo de una pequeña población.

Por cada habitante se ingresa: sexo (M/F) y edad. La carga de datos finaliza al ingresar cualquier otro valor para sexo.

El programa debe informar:

- A qué sexo corresponde la mayor cantidad de habitantes (considerar que puede ser igual)
- Cantidad de mujeres en edad escolar (4 a 18 años inclusive)
- Si hay algún varón que supere los 80 años de edad

```python
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'

print('CENSO POBLACIONAL')
print('*' * 80)

# Inicializar contadores y acumuladores
varones = 0
mujeres = 0
escolares = 0
mayor_80 = False

# Primera carga de datos de corte antes del ciclo
sexo = input('Ingrese sexo (M/F - Otro valor termina): ')

# Proceso repetitivo
while sexo == 'M' or sexo == 'F':
    # Carga de datos que no determinan el corte del ciclo
    edad = int(input('Ingrese edad: '))
    # Contar varones y mujeres
    if sexo == 'M':
        varones += 1
    else:
        mujeres += 1
    # Contar mujeres en edad escolar
    if sexo == 'F' and edad >= 4 and edad <= 18:
        escolares += 1
    # Detectar hombres de más de 80 años
    if sexo == 'M' and edad > 80:
        mayor_80 = True
    # Nueva carga de dato de corte dentro del ciclo
    sexo = input('Ingrese sexo (M/F - Otro valor termina): ')

# Resultados
print('*' * 80)
if varones > mujeres:
    print('Hay más varones que mujeres')
```

```
elif mujeres > varones:
    print('Hay más mujeres que varones')
else:
    print('La cantidad de mujeres y varones es igual')

print('Las mujeres en edad escolar son:',escolares)

if mayor_80 == True:
    print('Hay hombres mayores a 80 años')
else:
    print('NO hay hombres mayores a 80 años')
```

8 - Mayor número en orden par

Ingresar de a uno una serie de números. Encontrar e imprimir el mayor de todos los números pares cuyo número de orden sea par, el proceso terminará cuando el número leído sea igual a cero.

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'

print('Busqueda del mayor par de orden par')
print('=' * 80)

orden = 0
mayor = None

numero = int(input('Ingrese un numero (finaliza cuando ingrese 0): '))

while numero != 0:
    # Si es número par en orden par,
    if orden % 2 == 0 and numero % 2 == 0:
        # se busca el mayor
        if mayor is None or numero > mayor:
            mayor = numero
    orden += 1
    numero = int(input('Ingrese otro numero: '))

if not mayor is None:
    print('El mayor numero par ingresado en orden par es', mayor)
else:
    print('No se ingresaron números pares en orden par')
```

9 - Comisión de vendedores

Una empresa debe calcular el total de comisiones que debe abonar por ventas realizadas por sus vendedores, para ello le solicita un sistemita que le permita calcular dichos montos.

Se tiene conocimiento que la empresa tiene cuatro categorías de vendedores (1 a 4). Usted debe solicitar el ingreso de la categoría del vendedor y el total de la venta (el proceso termina cuando se ingrese una categoría igual a cero) y acumular las comisiones de las ventas rendidas por los vendedores de diferentes en base a los siguientes cálculos:

- Categoría 1: cobra una comisión de 10 %
- Categoría 2: cobra una comisión de 25 %
- Categoría 3: cobra una comisión de 30 %
- Categoría 4: cobra una comisión de 40 %

Una vez procesadas todas las ventas mostrar el total de comisiones a pagar por cada categoría de vendedores que tiene la empresa junto con el total general

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'

print('Sistema de Calculo de Comisiones a Vendedores')
print('*' * 80)

tot_cat_1 = 0
tot_cat_2 = 0
tot_cat_3 = 0
tot_cat_4 = 0

print('Inicio de calculo de comisión, con categoria 0 el proceso termina')
categoria = int(input('Ingrese la categoria de vendedor (1-4): '))

while categoria != 0:
    venta = float(input('Ingrese el total vendido por este vendedor: '))

    if categoria == 1:
        tot_cat_1 += venta * 0.10
    elif categoria == 2:
        tot_cat_2 += venta * 0.25
    elif categoria == 3:
        tot_cat_3 += venta * 0.30
    else:
        tot_cat_4 += venta * 0.40

    categoria = int(input('Ingrese una nueva categoria de vendedor (1-4): '))

total = tot_cat_1 + tot_cat_2 + tot_cat_3 + tot_cat_4

print('Resultados de comisiones calculadas')
print('*' * 80)
```

```
print('El total a pagar para la categoria 1 es de $', tot_cat_1, sep='')
print('El total a pagar para la categoria 2 es de $', tot_cat_2, sep='')
print('El total a pagar para la categoria 3 es de $', tot_cat_3, sep='')
print('El total a pagar para la categoria 4 es de $', tot_cat_4, sep='')
print('El total de comisiones a pagar es de $', total, sep='')
```

10 - Proceso de discriminantes

Un matemático desea un simple programa que le permita cargar una serie de números que representan los discriminantes de diferentes ecuaciones de segundo grado, el proceso de la secuencia finaliza cuando el matemático no desea seguir cargando discriminantes. Usted debe:

a) Determinar la cantidad de discriminantes que darán 2 raíces
b) Determinar la cantidad de discriminantes que darán una única raíz
c) Determinar la cantidad de discriminantes que daran raíces en el campo de los números imaginarios
d) Indicar el porcentaje que representa el punto c sobre el total de discriminantes procesados por el matemático

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'

print('Proceso de discriminantes de polinomios de segundo grado')
print('=' * 75)

cant_2raices = 0
cant_1raiz = 0
cant_raices_complejas = 0

sigue = input('Desea procesar un discriminante (S/N): ')
while sigue == 'S':
    delta = int(input('Ingrese el discriminante a procesar: '))
    if delta > 0:
        cant_2raices += 1
    elif delta == 0:
        cant_1raiz += 1
    else:
        cant_raices_complejas += 1
    sigue = input('Desea procesar otro discriminante (S/N): ')

total = cant_2raices + cant_1raiz + cant_raices_complejas

print('Cantidad de discriminantes con que se obtendran: ')
print('Dos raices:', cant_2raices)
print('Una raiz:', cant_1raiz)
print('Raices imaginarias:', cant_raices_complejas)

if total > 0:
    porc = cant_raices_complejas * 100 / total
    print('El porcentaje de discriminantes que obtienen ' \
          'raices complejas es: ', round(porc,2), '%', sep='')
```