Introducción a la programación

Clase viernes 6-9 pm

Semana 12

Practica 4

Samantha perez rojas

Camila Argeñal Coronado

Ejercicios de practica para prueba de habilidades

Algoritmos correspondientes

Link código git hub

https://github.com/Samifide/practica_programada_4.git

Ejercicio1

La Universidad Fidélitas le ha contratado para desarrollar un programa que le ayude a controlar las notas de los estudiantes de los 3 cursos de Programación Básica, cada curso cuenta con 5 estudiantes. La aplicación tiene por objetivo que una vez que esté ingresadas las notas, la aplicación nos brinde algunos datos.

- Se debe solicitar la nota final del curso de todos los estudiantes y ubicarlos en un arreglo de dos dimensiones.
- A partir de los datos presentes en el arreglo, debe calcular la nota promedio de todos los estudiantes y mostrarla al usuario.
- A partir de los datos presentes en el arreglo, debe calcular la nota promedio por cada uno de los grupos y mostrar los promedio al usuario informando el grupo al que pertenece.
- A partir de los datos presentes en el arreglo, debe mostrar al usuario por cada grupo cuál es el porcentaje de estudiantes aprobados, recuerde que los estudiantes aprobados son aquellos que obtuvieron al menos una nota de 70.
- A partir de los datos presentes en el arreglo, debe mostrar al usuario cuál ha sido la nota mayor y la menor por cada uno de los grupos.
- Cada una de las opciones mencionadas debe ser creada en un sub proceso

```
def ingresar_notas():
    notas = []
    for i in range(3):
        print("Ingrese las notas para el Curso", i+1, ": ")
        curso = []
        for estudiante in range(5):
            nota = input(f"Ingrese la nota del estudiante {estudiante+1} del Curso {i+1}: ")
            nota = float(nota)
            curso.append(nota)
```

```
        notas.append(curso)
    return notas



def cal_prom_curso(curso):
    suma_notas = 0
    total_estudiantes = len(curso)
    for nota in curso:
        suma_notas += nota
    return suma_notas / total_estudiantes

def calcular_prom_todos(notas):
    total_notas = 0
    total_estu = 0
    for curso in notas:
        for nota in curso:
            total_notas = total_notas + nota
            total_estu = total_estu + 1
    promedio_total = total_notas / total_estu
    return promedio_total



def calcular_porcentaje_aprobados(notas):
    num = 0
    for curso in notas:
        aprobados = 0
        for nota in curso:
            if nota >= 70:
                aprobados += 1
        porcentaje_aprobados = 100 * (aprobados / len(curso))
        print("\nEl porcentaje de aprobados del curso", num + 1, "es:", porcentaje_aprobados, "%")
        num += 1
```

```
def obtener_may_men(notas):
    for curso in notas:
        mayor = max(curso)
        menor = min(curso)
        print("Curso", notas.index(curso) + 1, ": Nota mayor:", mayor, "Nota menor:", menor)




notas = ingresar_notas()

print("\nMatriz para los tres cursos:")
for i in range(len(notas)):
    print("\nMatriz para el Curso", i+1, ":")
    for fila in notas[i]:
        print(fila)



promedio_total = calcular_prom_todos(notas)
print("\nPromedio total de los tres cursos:", promedio_total)



for i in range(len(notas)):
    promedio_curso = cal_prom_curso(notas[i])
    print("\nPromedio del curso", i+1, ":", promedio_curso)

calcular_porcentaje_aprobados(notas)

obtener_may_men(notas)
```

Algoritmo

1. Se ingresan los datos de entrada, se solicita al usuario que digite las notas de los 3 curso y las almacena en un arreglo
2. Comienza el proceso de los datos
3. Se define la función para calcular el promedio por curso, suma las notas en el curso y las divide por los estudiantes de ese curso
4. Se define la función para calcular el promedio de los 3 cursos, suma todas las notas de los cursos y las divide por todos los estudiantes totales
5. Se define la función para calcular el porcentaje de aprobados, calcula el número de aprobados por curso y calcula el porcentaje, con una nota igual o superior a 70
6. Se define la función para calcular la nota mayor y menor por curso, se calcula el máximo y mínimo de las notas.
7. Continua la salida de datos
8. Se imprime el promedio total
9. Se imprime mediante un rango y el llamado de la función el promedio por curso
10. Se imprime el porcentaje de aprobados mediante un rango y el llamado de la función aprobados por curso
11. Se imprime la nota mayor y menor por curso mediante un rango y el llamado de la función

Ejercicio 2

La empresa La Casadora S.A. requiere desarrollar un programa que le permita realizar algunos cálculos estadísticos importantes para la toma de decisiones.

- Se debe capturar la cantidad de pasajeros de cada uno de los 4 servicios que se realizan al día para los 5 días de la semana.
- Ya que el autobús tiene un capacidad máxima de 60 pasajeros, se debe verificar que no se ingrese un valor mayor a 60.
- El programa debe mostrar el promedio de pasajeros de cada uno de los días.
- El programa debe mostrar el promedio general de todos los días y todos los servicios.
- El programa debe mostrar cuál de los 4 servicios durante el día es el mejor (en el que se transportan más personas).
- El programa debe mostrar el momento en que menos pasajeros se transportaron (servicio y día).
- La captura de los datos y cada uno de los procesos que muestran información deben estar ubicados en su propio sub proceso.
- Debe utilizar un arreglo de 2 dimensiones.
- Debe utilizar nombres representativos para los elementos (variables, arreglos, sub

procesos, etc.).

```
#ejercicio2
pasajeros = []
def ingresar_pasajeros(pasajeros):
    for i in range(5):
        pasajeros_dia = []
        print("Ingrese la cantidad de pasajeros para el día", i + 1, ":")
        for s in range(4):
            while True:
                suma_pas = input(f"Cantidad de pasajeros para el servicio {s + 1}: ")
                suma_pas = int(suma_pas)
                if suma_pas <= 60:
                    pasajeros_dia.append(suma_pas)
                    break
                else:
                    print("Demasiados pasajeros")
        pasajeros.append(pasajeros_dia)
    return pasajeros

def prom_pasajerostotal(pasajeros):
    total_pas = len(pasajeros)
    total_pas = total_pas * 4
    suma_pasajeros = 0
    for dia in pasajeros:
        for cantidad_pasajeros in dia:
            suma_pasajeros += cantidad_pasajeros
    return suma_pasajeros / total_pas

def prom_pasajeros(pasajeros_dia):
    suma_pasaj = sum(pasajeros_dia)
    total_pas = len(pasajeros_dia)
    return suma_pasaj / total_pas
```

```
def mejor_servicio(pasajeros_dia):
    servicio1 = pasajeros_dia[0]
    servicio2 = pasajeros_dia[1]
    servicio3 = pasajeros_dia[2]
    servicio4 = pasajeros_dia[3]

    if servicio1 >= servicio2 and servicio1 >= servicio3 and servicio1 >= servicio4:
        return 1
    elif servicio2 >= servicio3 and servicio2 >= servicio4:
        return 2
    elif servicio3 >= servicio4:
        return 3
    else:
        return 4

def menor_servicio_dia(pasajeros):
    minimos = []
    for dia in pasajeros:
        minimo_servicio_dia = min(dia)
        minimos.append(minimo_servicio_dia)
    return minimos

ingresar_pasajeros(pasajeros)

num = 1
for dia in pasajeros:
    promedio = prom_pasajeros(dia)
    print("El promedio de pasajeros para el día", num, "es:", promedio)
    mejor = mejor_servicio(dia)
    print("El mejor servicio para el día", num, "es el servicio", mejor)
```

```
    num += 1


promedio_total = prom_pasajerostotal(pasajeros)

print("El promedio total de pasajeros es:", promedio_total)

minimos = menor_servicio_dia(pasajeros)

elmenor = min(minimos)

dia_menor = minimos.index(elmenor) + 1

print("El menor servicio es de ", elmenor, "y el día es:", dia_menor)
```

Algoritmo:

1. Comienza la entrada de datos se le solicita al usuario ingresar la cantidad de pasajeros y se almacena en un arreglo
2. Comienza el proceso de datos
3. Se define la función para calcular el promedio total de los pasajeros, se suma la cantidad de pasajeros y se divide por el total de días y servicios
4. Se define el promedio de pasajeros por día se suma los pasajeros del día y se divide por el total
5. Se define la función del mejor servicio del día, se hacen las validaciones necesarias para calcular cual es el servicio más alto
6. Se define la función del menor servicio se calculan todos los servicios de todos los días y se sacan los mínimos
7. Continua la salida de datos
8. Se imprime el promedio de pasajeros del día mediante un for y el llamado de la función
9. Se imprime el mejor servicio del día mediante un for y el llamado de la función
10. Se imprime el promedio total mediante el llamado de su respectiva función
11. Se calcula el menor servicio y día calculando el servicio menor con los mínimos de todos los días y mediante y el llamado de su respectiva función

Ejercicio 3

El equipo de fútbol nacional Las Tuercas F.C. cuenta con 25 jugadores en su planilla, cada uno de ellos recibe un salario mensual diferente y el equipo paga los salarios en efectivo. Por lo que le ha solicitado desarrollar un programa que reciba el salario de cada uno de los futbolistas, posterior, debe informar cuál es el monto de dinero que se debe retirar del banco y la denominación de billetes y monedas necesaria para poder realizar el pago exacto en efectivo. Considere las denominaciones de billetes y monedas disponibles en nuestro país.

Código

```
def retiro(lista_Futbolistas):
   monto=int(input("Porfavor escriba la cantidad de dinero que desea retirar \n
**!!recuerde¡¡ se le devolvera en el dollar de su país**"))
   nombre=input("Porfavor escriba su nombre con el que se registro previamente")

   for i in range(len(lista_Futbolistas)):
      e=lista_Futbolistas[i]
      Verificar=e[0]
      if nombre == Verificar:
         almacenado=lista_Futbolistas[i][1]
         Sobrante=almacenado-monto
         if Sobrante<0:
            print("no tiene dinero suficiente para realizar el retiro")
         else:
            lista_Futbolistas[i][1]=Sobrante
            montoEnColon=monto*507.63
            billetes=int(montoEnColon/(500*2))
            centabos=int(montoEnColon%(500*2))
            print("Se le estan devolviendo",int(montoEnColon),"colones o ",billetes,
"billetes y ", ("%.2f" % centabos), "centavos \n en su cuenta
quedan",lista_Futbolistas[i][1])


def registro(lista_Futbolistas):
   print("Esto es un registro")
```

```
    nombre_futbolista=input("Porfavor escriba su nombre como futbolista: ")
    salaraio=int(input("Porfavor escriba su salario esxacto en dolares:"))
    lista_Futbolistas.append([nombre_futbolista,salaraio])

def agregar(lista_Futbolistas):
    monto=int(input("Porfavor escriba la cantidad de dinero que desea Agregar en
dolares"))
    nombre=input("Porfavor escriba su nombre con el que se registro previamente")

    for i in range(len(lista_Futbolistas)):
        e=lista_Futbolistas[i]
        Verificar=e[0]
        if nombre == Verificar:
            almacenado=lista_Futbolistas[i][1]
            Sobrante=almacenado+monto
            lista_Futbolistas[i][1]=Sobrante
            print("Se le estan agregando",monto, "\n en su cuenta
quedan",lista_Futbolistas[i][1])



lista_Futbolistas=[]
while True:
    print("Antes de cualquier accion debe registrar futbolistas")
    registro(lista_Futbolistas)
    print(lista_Futbolistas)
    while True:
        Decicion=int(input("que accion desea realizar \n 1-Realizar un registro nuevo \n
2-Retirar dinero \n 3-Agregar dinero"))
        if Decicion==1:
            registro(lista_Futbolistas)
            print(lista_Futbolistas)
        elif Decicion==2:
            retiro(lista_Futbolistas)
        elif Decicion==3:
            agregar(lista_Futbolistas)
        else:
            print("Accion no opcional")
```

Algoritmo

1. creación lista para futbolistas vacia

2. Entrada a primer ciclo
    2.1. llamado de función registro futbolistas
    2.2. dar lista como parámetro
3. pedir insertar un futbolista
4. pedir nombre de futbolista
5. pedir salario correspondiente en dólares
    5.1. regreso ciclo principal
6. dar opciones disponibles de decisión (registrar mas futbolistas, retirar dinero en moneda respectiva del país,)
7. usuario-futbolista selecciona acción
8. caso registro
    8.1. llamado de función registro futbolistas
    8.2. dar lista como parámetro
    8.3. pedir insertar un futbolista
    8.4. pedir nombre de futbolista
    8.5. pedir salario correspondiente en dólares
    8.6. regreso ciclo principal
9. caso retiro de dinero
    9.1. llamado de función retiro
    9.2. dar lista como parámetro
    9.3. pedir monto de dinero a retirar
        9.3.1. junto con advertencia de que se dará ese monto en moneda respectiva a su país
    9.4. pedir nombre de registro como futbolista
    9.5. realizar búsqueda de nombre registrado en lista en parámetro
    9.6. encontrar nombre
    9.7. verificar si tiene dinero suficiente para hacer el retiro
        9.7.1. caso de tener dinero suficiente
            9.7.1.1. realizar resta y actualizar saldo de dinero actual
            9.7.1.2. conversión de dólares a dinero del país
            9.7.1.3. separar monto de billetes
            9.7.1.4. separar monto de monedas
            9.7.1.5. mostrar monto a devolver-restado de la cuenta
            9.7.1.6. mostrar dinero actual
            9.7.1.7. se devuelve al ciclo principal

9.7.2. caso de no contar con el dinero suficiente
9.7.2.1. se manda un mensaje de no tener suficientes fondos
9.7.2.2. se devuelve al ciclo principal

10. caso agregar dinero
10.1. llamado de función retiro
10.2. dar lista como parámetro
10.3. pedir monto de dinero a agregar
10.3.1. junto con advertencia de que se pide el monto en dólares
10.4. pedir nombre de registro como futbolista
10.5. realizar búsqueda de nombre registrado en lista en parámetro
10.6. encontrar nombre
10.7. ubicar monto de dinero del futbolista respectivo
10.8. realizar suma de dinero actual mas dinero agregado
10.9. actualizar cantidad de dinero del futbolista
10.9.1.1. se devuelve al ciclo principal

11. caso de acción no disponible
11.1. Mandar mensaje de acción no opcional

Fin del algoritmo

Ejercicio 3 (continuación)

Para ejemplificar los cálculos a realizar se presenta la siguiente información. El jugador “Luis” gana ₡1,845,725 lo que implica que para hacer su pago se requiere el siguiente desglose. Denominación Cantidad Monto 50,000 36 1,800,000 20,000 2 40,000 5,000 1 5,000 500 1 500 100 2 200 25 1 25 1,845,725 Una vez calculado el desglose de un jugador, debe acumular la cantidad de cada denominación por todos los jugadores. Eso significa que debe indicar la sumatoria de todos los billetes de 50 mil, todos los de 20 mil, etc.

Código

```
def desglose(lista_Futbolistas):
   for i in lista_Futbolistas:
      print("nombre",i[0])
      print("Desglose de salario")
      salario_en_dolares=i[1]
      salario=int(salario_en_dolares*507.63)
      print("salario en dolares: ",salario_en_dolares)
      print("salario en colones: ",salario)

      print("Billetes disponibles: \n 50000 \n 20000 \n 10000 \n 5000 \n 2000 \n 1000
\n 500 \n 100 \n 50 \n 25 \n 10 \n 5 \n 1")

      if salario>= 50000 :
         billetesCanticuenta=int(salario/50000)
         cincuentaMil=billetesCanticuenta*50000
         salario=salario-cincuentaMil
         print(billetesCanticuenta,"billetes de 50000",cincuentaMil )

      if salario>= 20000 :
         billetesCantiVeinte=int(salario/20000)
         VeinteMil=billetesCantiVeinte*20000
         salario=salario-VeinteMil
         print(billetesCantiVeinte,"billetes de 20000",VeinteMil)

      if salario>= 10000 :
         billetesCantidiez=int(salario/10000)
```

```
    diesmill=billetesCantidiez*10000
    salario=salario-diesmill
    print( billetesCantidiez,"billetes de 10000",diesmill)

if salario>= 5000 :
    billetesCanticincomil=int(salario/5000)
    cincoMil=billetesCanticincomil*5000
    salario=salario-cincoMil
    print( billetesCanticincomil,"billetes de 5000",cincoMil)

if salario>= 2000 :
    billetesCantiDosmil=int(salario/2000)
    DosMil=billetesCantiDosmil*2000
    salario=salario-DosMil
    print( billetesCantiDosmil,"billetes de 2000",DosMil)

if salario>= 1000 :
    billetesCantimil=int(salario/1000)
    mil=billetesCantimil*1000
    salario=salario-mil
    print( billetesCantimil,"billetes de 1000",mil)

if salario>= 500 :
    colonesquinientos=int(salario/500)
    quinientos=colonesquinientos*500
    salario=salario-quinientos
    print( colonesquinientos,"monedas de 500",quinientos)

if salario>= 100 :
    colonescien=int(salario/100)
    cien=colonescien*100
    salario=salario-cien
    print( colonescien,"monedas de 100",cien)

if salario>= 50 :
    colonescincuenta=int(salario/50)
    cincuenta=colonescincuenta*50
    salario=salario-cincuenta
    print( colonescincuenta,"monedas de 50",cincuenta)
```

```
      if salario>= 25 :
         colonesveintisinco=int(salario/25)
         veinticinco=colonesveintisinco*25
         salario=salario-veinticinco
         print( colonesveintisinco,"monedas de 25",veinticinco)

      if salario>= 10 :
         colonesdies=int(salario/10)
         dies=colonesdies*100
         salario=salario-dies
         print( colonesdies,"monedas de 10",dies)

      if salario>= 5 :
         colonescinco=int(salario/5)
         cinco=colonescinco*5
         salario=salario-cinco
         print( colonescinco,"monedas de 5",cinco)

      if salario>= 1 :
         colonesuno=int(salario/1)
         uno=colonesuno*1
         salario=salario-uno
         print( colonesuno,"monedas de 1",uno)


def retiro(lista_Futbolistas):
   monto=int(input("Porfavor escriba la cantidad de dinero que desea retirar \n
**!!recuerde¡¡ se le devolvera en el dollar de su país**"))
   nombre=input("Porfavor escriba su nombre con el que se registro previamente")

   for i in range(len(lista_Futbolistas)):
      e=lista_Futbolistas[i]
      Verificar=e[0]
      if nombre == Verificar:
         almacenado=lista_Futbolistas[i][1]
         Sobrante=almacenado-monto
         if Sobrante<0:
            print("no tiene dinero suficiente para realizar el retiro")
         else:
            lista_Futbolistas[i][1]=Sobrante
```

```
            montoEnColon=monto*507.63
            billetes=int(montoEnColon/(500*2))
            centabos=int(montoEnColon%(500*2))
            print("Se le estan devolviendo",int(montoEnColon),"colones o ",billetes,
"billetes y ", ("%.2f" % centabos), "centavos \n en su cuenta
quedan",lista_Futbolistas[i][1])


def registro(lista_Futbolistas):
   print("Esto es un registro")
   nombre_futbolista=input("Porfavor escriba su nombre como futbolista: ")
   salaraio=int(input("Porfavor escriba su salario esxacto en dolares:"))
   lista_Futbolistas.append([nombre_futbolista,salaraio])

def agregar(lista_Futbolistas):
   monto=int(input("Porfavor escriba la cantidad de dinero que desea Agregar en
dolares"))
   nombre=input("Porfavor escriba su nombre con el que se registro previamente")

   for i in range(len(lista_Futbolistas)):
      e=lista_Futbolistas[i]
      Verificar=e[0]
      if nombre == Verificar:
         almacenado=lista_Futbolistas[i][1]
         Sobrante=almacenado+monto
         lista_Futbolistas[i][1]=Sobrante
         print("Se le estan agregando",monto, "\n en su cuenta
quedan",lista_Futbolistas[i][1])



lista_Futbolistas=[]
while True:
   print("Antes de cualquier accion debe registrar futbolistas")
   registro(lista_Futbolistas)
   print(lista_Futbolistas)
   while True:
      Decicion=int(input("que accion desea realizar \n 1-Realizar un registro nuevo \n
2-Retirar dinero \n 3-Agregar dinero \n 4-Ver desglose de pago de cada jugador"))
      if Decicion==1:
```

```
        registro(lista_Futbolistas)
        print(lista_Futbolistas)
    elif Decicion==2:
        retiro(lista_Futbolistas)
    elif Decicion==3:
        agregar(lista_Futbolistas)
    elif Decicion==4:
        desglose(lista_Futbolistas)
    else:
        print("Accion no opcional")
```

Algoritmo

1. creación lista para futbolistas vacia

2. Entrada a primer ciclo
    2.1. llamado de función registro futbolistas
    2.2. dar lista como parámetro
3. pedir insertar un futbolista
4. pedir nombre de futbolista
5. pedir salario correspondiente en dólares
    5.1. regreso ciclo principal
6. dar opciones disponibles de decisión (registrar mas futbolistas, retirar dinero en moneda respectiva del país,)
7. usuario-futbolista selecciona acción
8. caso registro
    8.1. llamado de función registro futbolistas
    8.2. dar lista como parámetro
    8.3. pedir insertar un futbolista
    8.4. pedir nombre de futbolista
    8.5. pedir salario correspondiente en dólares
    8.6. regreso ciclo principal
9. caso retiro de dinero
    9.1. llamado de función retiro
    9.2. dar lista como parámetro
    9.3. pedir monto de dinero a retirar
        9.3.1. junto con advertencia de que se dará ese monto en moneda respectiva a su país

9.4. pedir nombre de registro como futbolista
9.5. realizar búsqueda de nombre registrado en lista en parámetro
9.6. encontrar nombre
9.7. verificar si tiene dinero suficiente para hacer el retiro
9.7.1. caso de tener dinero suficiente
9.7.1.1. realizar resta y actualizar saldo de dinero actual
9.7.1.2. conversión de dólares a dinero del país
9.7.1.3. separar monto de billetes
9.7.1.4. separar monto de monedas
9.7.1.5. mostrar monto a devolver-restado de la cuenta
9.7.1.6. mostrar dinero actual
9.7.1.7. se devuelve al ciclo principal
9.7.2. caso de no contar con el dinero suficiente
9.7.2.1. se manda un mensaje de no tener suficientes fondos
9.7.2.2. se devuelve al ciclo principal
10. caso agregar dinero
10.1. llamado de función retiro
10.2. dar lista como parámetro
10.3. pedir monto de dinero a agregar
10.3.1. junto con advertencia de que se pide el monto en dólares
10.4. pedir nombre de registro como futbolista
10.5. realizar búsqueda de nombre registrado en lista en parámetro
10.6. encontrar nombre
10.7. ubicar monto de dinero del futbolista respectivo
10.8. realizar suma de dinero actual mas dinero agregado
10.9. actualizar cantidad de dinero del futbolista
10.9.1. se devuelve al ciclo principal
11. caso desglose de dinero
11.1. llamado de función desglose
11.2. dar lista como parámetro
11.3. Crear recorrido de lista futbolistas
11.4. Mostrar conversión di dinero de dólares a moneda del país
11.5. Mostrar dinero sin conversión
11.6. Mostrar cantidades de dinero que maneja el pais
11.7. Realización de condicionales en comparación a dinero con moneda del país respectivo

11.8. Ejecutar cambios y actualización de la moneda del país en desglose condición de que el dinero sea ajustable a la cantidad manejable del país
11.9. Mostrar cambios en moneda, dinero necesario para llenar la cantidad actual del futbolista
11.10. Mostrar billetes necesario y monedas necesarias
11.11. se devuelve al ciclo principal
12. caso de acción no disponible
12.1. Mandar mensaje de acción no opcional

Fin del algoritmo

Ejercicio 4

En el abastecedor La Deportiva se ha detectado la necesidad de controlar el aforo de personas que se mantienen dentro del negocio para cumplir con la normativa del Ministerio de Salud. El propietario del abastecedor requiere desarrollar dos aplicaciones, una para registrar el ingreso de cada persona (uno a uno) y otra para registrar el egreso (uno a uno). Son dos aplicaciones independientes porque existe una puerta para el ingreso y otra para el egreso. La capacidad del abastecedor es de 10 personas como máximo. Considere almacenar la cantidad de personas actuales en un archivo de texto para que esta información se comparta entre las aplicaciones.

Código

```
import time
import os

RegistrosCompletos=open("RegistrosCompletos.txt","a")
RegistrosCompletos.close()

def negocio(Nombre,Personas,Eliminados):
   print("Ahora estas en el sistema del negocio \n")
   for i in range(len(Personas)):
      Persona=Personas[i]
      if Nombre==Persona[1]:
         print(Persona)
         print("\n")

         desicion=int(input("Que accion desea realiza \n 1-Volver al programa de
registro \n 2- salir permanenetemente del sistema"))
```

```
        if desicion==2:
            print("\n Se esta registrando sus datos a eliminados")
            Hora=time.strftime("%H:%M:%S")
            fecha=time.strftime("%d/%m/%y")
            eliminado=Persona
            eliminado+=("\n Hora de eliminado",Hora,"\n Fecha de eliminacion de
registro",fecha)
            Eliminados.append(eliminado)
            Personas.remove(Persona)
            print("\n ****Lista de Registrados actuales****")
            print(Personas)
            print("\n ****Lista de eliminados****")
            print(Eliminados)
            print("\n")


def registro(Personas):
    print("esta es el area de registro")
    new=input("Escriba su nombre ")
    Hora=time.strftime("%H:%M:%S")
    fecha=time.strftime("%d/%m/%y")
    Personas.append(["nombre de persona",new, "\n Hora de ingreso en sistema:
",Hora, "\n fecha de ingreso en sistema", fecha])
    print("\n")


Personas=[]
Eliminados=[]

registro(Personas)

while True:
    print(Personas)
    RegistrosCompletos=open("RegistrosCompletos.txt","w")
    RegistrosCompletos.write(str(Personas))
    RegistrosCompletos.write(str(Eliminados))
    RegistrosCompletos.close()

    print("\n")
```

```
   desicion=int(input("¿Desea Ingresar mas personas o quiero dirigisrse al negocio? \n |Deseo ingresar mas personas=1 | Deseo ingresar al area del negocio=2| Visualisar Registros y elimanaciones"))
   if desicion== 1:
      if len(Personas)<10 :
         print("\n")
         registro(Personas)
      elif len(Personas)>=10:
         print("\n")
         print("No puede ingresar a mas personas a saturado el limite de personas, se va a dirigir al negocio por Default")
         Nombre=input("Escriba su nombre porfavor")
         negocio(Nombre,Personas,Eliminados)

   elif desicion== 2:
      Nombre=input("Escriba su nombre porfavor")
      negocio(Nombre,Personas,Eliminados)

   elif desicion== 3:
      print("\n ****Lista de Registrados actuales****")
      print(Personas)
      print("\n ****Lista de eliminados****")
      print(Eliminados)

   else:
      print("no opcional")
```

Algoritmo

1. Importar librerías
   1.1 librería time
   1.2 Importar librería os
2. Creación de archivo registros como “a”
3. Cerrar archivo registros
4. Creación de listas vacío personas actuales ministradas
5. Creación de lista vacía personas eliminadas del registro
6. Llamado de función registro de personas
   6.1 pedir nombre
   6.2 guardar hora de registro

6.3 guardar fecha de registro

6.4 Agregar fecha, hora, nombre de persona a lista de personas (registradas actualmente)

6.5 Volver a ciclo principal

7. abrir archivo como “w”.
8. Agregar-escribir lista de lista de personas (registradas actualmente)
9. Agregar-escribir lista de eliminado actualmente
10. Cerrar archivo
11. Dar opciones a usuario; ingresar mas personas, ingresar al area del negocio, Visualizar Registros y eliminaciones
12. En caso de ingresar mas personas.

12.1 Contar personas actuales en el registro

12.1.1 Caso de tener suficiente espacio de registro

12.1.1.1 Llamado de función ingreso registro

12.1.1.2 pedir nombre

12.1.1.3 guardar hora de registro

12.1.1.4 guardar fecha de registro

12.1.1.5 Agregar fecha, hora, nombre de persona a lista de personas (registradas actualmente)

12.1.1.6 Volver a ciclo principal

12.1.2 Caso de no tener suficiente espacio de registro

12.1.2.1 Decir razón de porque no puede ingresar a registro

12.1.2.2 Pedir nombre

12.1.2.3 Llamado función de negocio

12.1.2.4 Realizar recorrido de datos, mostrar información de la persona ingresada

12.1.2.5 Mostrar información propia

12.1.2.6 Preguntar que acción desea realizar (salir del negocio-sistema o regresar al ciclo principal-registro)

12.1.2.6.1 A-Caso de eliminación

12.1.2.6.2 Ubicar celda de información

12.1.2.6.3 Guardar hora y fecha actuales

12.1.2.6.4 Trasladar información re personas actuales registradas a eliminados

12.1.2.6.5 Agregar a eliminados fecha y hora de eliminación de sistema del negocio

12.1.2.6.6 Mostrar cambios

12.1.2.6.7 B-Caso de volver al resjistro

12.1.2.7 Volver a ciclo principal

1. abrir archivo como “w”.
2. Agregar-escribir lista de lista de personas (registradas actualmente)
3. Agregar-escribir lista de eliminado actualmente
4. Cerrar archivo

13. En caso de ingresar al área del negocio

13.1.1.1 Pedir nombre

13.1.1.2 Llamado función de negocio

13.1.1.3 Realizar recorrido de datos, mostrar información de la persona ingresada

13.1.1.4 Mostrar información propia

13.1.1.5 Preguntar que acción desea realizar (salir del negocio-sistema o regresar al ciclo principal-registro)

13.1.1.5.1 A-Caso de eliminación

13.1.1.5.2 Ubicar celda de información

13.1.1.5.3 Guardar hora y fecha actuales

13.1.1.5.4 Trasladar información re personas actuales registradas a eliminados

13.1.1.5.5 Agregar a eliminados fecha y hora de eliminación de sistema del negocio

13.1.1.5.6 Mostrar cambios

13.1.1.5.7 B-Caso de volver al resjistro

13.1.1.6 Volver a ciclo principal

1. abrir archivo como “w”.
2. Agregar-escribir lista de lista de personas (registradas actualmente)
3. Agregar-escribir lista de eliminado actualmente
4. Cerrar archivo

14. Caso de no tener opcion disponible
    1. Dar mensaje no opcional