Modelo de Sistema Bancario

El programa consiste en emular todas las opciones que se pueden hacer a través de una plataforma web de un banco.

Se hizo uso de la librería "string", importando el atributo "puntuaction", el cual contiene todos los signos de puntuación del lenguaje inglés.

```
from string import punctuation
```

En primera instancia se tiene un submétodo el cual será llamado cada vez que se quiera crear un nuevo usuario, para esto se hace uso de un diccionario el cual tendrá valores vacíos que se llenarán mediante el transcurso del programa.

```
def cuenta():
      account = {
      "nombre": None,
      "apellido": None,
      "cedula": None,
      “usuario": None,
      "clave": None,
      "posicion consolidada": 0.0,
      }
      return account
```

El submétodo **validarClave()** será encargado de que, al momento de escribir una contraseña para un usuario determinado, ésta cumpla con una serie de prerrequisitos como lo son: mínimo 6 caracteres, uso de mayúsculas, uso de minúsculas, uso de dígitos, uso de caracteres especiales

```
def validarClave(contraseña):
      longitud = False
      minuscula = False
      mayuscula = False
      caracter_especial = False
      numero = False
      if len(contraseña) >= 6:
      longitud = True
      for i in contraseña:
            if i.islower():
            minuscula = True
            if i.isupper():
            mayuscula = True
            if i in punctuation:
                  caracter_especial = True
            if i.isdigit():
            numero = True
            if longitud and minuscula and mayuscula and caracter_especial and numero:
                  return True
            else:
                  return False
```

El método **crearCuenta()** engloba los anteriores submétodos, aquí se le solicitará al usuario: nombre, apellido, cedula, usuario, contraseña

```
def crearCuenta():
    print("\nExcelente. Ha introducido la opcion 1. (Crear cuenta)\n")
    registro = cuenta()
    nombre = input("Introduce el nombre: ")
    while not nombre.isalpha():
        print("\nEl nombre no debe tener caracteres especiales ni numeros.")
        nombre = input("Nombre: ")
    print("\nNombre validado.")
    apellido = input("\nIntroduce el apellido: ")
    while not apellido.isalpha():
        print("\nEl apellido no debe tener caracteres especiales ni numeros.")
        apellido = input("Apellido: ")
    print("\nApellido validado.")
    cedula = input("\nCedula: ")
    while not cedula.isdigit():
        print("\nLa cedula no debe tener caracteres especiales ni letras.")
        cedula = input("Cedula: ")
    print("\nCedula validada.\n")
    usuario = input("Introduce tu nombre de usuario: ")
    print("\nUsuario validado.")
    print("\nRequisitos para la creacion de la clave.")
    print("* Que tenga como minimo 6 caracteres.")
    print("* Que tenga al menos una letra en mayuscula y en minusculas.")
    print("* Que tenga al menos un caracter especial y un numero")
    print("Ejemplo: Pablo123@")
    clave = input("Introduce la clave: ")
    validacion_clave = validarClave(clave)
    while not validacion_clave:
        print("\nIntroduce una clave que cumpla con los requisitos.")
        print("* Que tenga como minimo 6 caracteres.")
        print("* Que tenga al menos una letra en mayuscula y en minusculas.")
        print("* Que tenga al menos un caracter especial y un numero")
        print("Ejemplo: Pablo123@\n")
        clave = input("Introduce una clave valida: ")
        validacion_clave = validarClave(clave)
    print("\nClave validada.")
    registro["nombre"] = nombre
    registro["apellido"] = apellido
    registro["cedula"] = cedula
    registro["usuario"] = usuario
    registro["clave"] = clave
    saldo_inicial = 50.0
    registro["posicion consolidada"] += saldo_inicial
    existe = False
    for fila in database:
        if fila["cedula"] == cedula or fila["usuario"] == usuario:
            existe = True
            print("La persona con el usuario y/o cedula anteriormente
introducida ya existe.")
    if existe == False:
        database.append(registro)
        print("\nUsuario registrado correctamente. Retornando al menu
principal")
```

Todos los datos solicitados pasarán por un proceso de validación (en este momento se hará uso del submétodo **validarClave()** para verificar si la contraseña cumple los requisitos para ser aceptada)

Una vez aceptados los datos, se procederá a pasar los valores a las claves del diccionario anteriormente

mencionado, además de establecer un monto de 50 Bs como saldo inicial. Por último, se valida si la persona que se registró existe anteriormente en el banco mediante su apodo y/o cédula, en caso de estar registrada no se tomará en cuenta el registro y retornara al menú principal, en caso contrario se añadirán los datos guardados en ese diccionario en un vector global denominado "database", el cual se encarga de llevar los usuarios de cada persona para los distintos procesos de los que se quiera hacer uso

Pasando al método **depositoBancario()**, se procederá a pedirle a un usuario que introduzca su apodo y suclave (previamente ya existentes), el cual será validado, para verificar si retorna al menú principal (al estar erróneo) o si continúa con el depósito bancario. En caso de seguir con el depósito, se pedirá introducir el monto a depositar en la cuenta (deberá ser mayor a 0), en este momento se sumará ese monto al saldo actual de esa persona (actualizando el valor de la clave "posición consolidada")

```
def depositoBancario():
    print("\nHa seleccionado la opcion 2. (Realizar un deposito bancario)\n")
    existe_usuario = False
    existe_clave = False
    usuario = input("Introduzca un nombre de usuario existente: ")
    for fila in database:
        if fila["usuario"] == usuario:
            existe_usuario = True
            print("\nUsuario encontrado\n")
            clave = input("Introduzca la clave respectiva: ")
            if fila["clave"] == clave:
                existe_clave = True
                print("\nUsuario y clave correcta.\n")
                print("Bienvenido! ",fila["nombre"]," ",fila["apellido"])
                monto = float(input("\nIntroduzca el monto a depositar en su
cuenta: \n"))
                if monto > 0:
                    fila["posicion consolidada"] += monto
                    print("Monto introducido: ",monto)
                    print("Monto total: ",fila["posicion consolidada"])
                else:
                    print("\nError al introducir un monto. Intentalo de
nuevo...\n")
    if existe_usuario == False:
        print("\nEl usuario introducido no se encuentra registrado en el
sistema.\n")
    else:
        if existe_clave == False:
 print("\nLa clave introducida es incorrecta...\n")
```

El siguiente método es **retiroBancario()**, el cual pedirá a un usuario ingresar su apodo y contraseña, se validará que el mismo exista y que no sea erróneo, posteriormente se le pedirá al usuario que indique el monto a retirar, el cual deberá ser menor o igual a su saldo disponible, una vez retirado el dinero, seactualizará el saldo disponible en su cuenta (actualizando el valor de la clave "posición consolidada")

```
def retiroBancario():
    print("\nHa seleccionado la opcion 3. (Realizar un retiro bancario)\n")
    existe_usuario = False
    existe_clave = False
    usuario = input("Introduzca un nombre de usuario existente: ")
    for fila in database:
        if fila["usuario"] == usuario:
            existe_usuario = True
            print("\nUsuario encontrado\n")
            clave = input("Introduzca la clave respectiva: ")
            if fila["clave"] == clave:
                existe_clave = True
                print("\nUsuario y clave correcta.\n")
                print("Bienvenido! ",fila["nombre"]," ",fila["apellido"])
                monto = float(input("\nIntroduzca el monto a retirar de su cuenta: "))
                if monto > fila["posicion consolidada"] or monto <= 0:
                    print("\nIntroduzca un monto menor o igual al saldo disponible de
su cuenta. Intente de nuevo\n")
                elif monto > 0 and monto <= fila["posicion consolidada"]:
                    fila["posicion consolidada"] -= monto
                    print("Monto introducido: ",monto)
                    print("Monto total: ",fila["posicion consolidada"])
    if existe_usuario == False:
        print("\nEl usuario introducido no se encuentra registrado en el sistema.\n")
    else:
        if existe_clave == False:
            print("\nLa clave introducida es incorrecta...\n")
```

Pasando al siguiente modulo denominado **consultarCuenta()**, se pedirá al usuario que introduzca su apodo y su contraseña, validando que este exista y que los datos no sean incorrectos. Una vez validado el usuario, se procederá a enseñar: su nombre, apellido, cédula y su saldo disponible. Posteriormente se procede a volver al menú principal.

```
def consultarCuenta():
    print("\nHa seleccionado la opcion 4. (Consultar datos de una cuenta)\n")
    existe_usuario = False
    existe_clave = False
    usuario = input("Introduzca un nombre de usuario existente: ")
    for fila in database:
        if fila["usuario"] == usuario:
            existe_usuario = True
            print("\nUsuario encontrado\n")
            clave = input("Introduzca la clave respectiva: ")
            if fila["clave"] == clave:
                existe_clave = True
                print("\nUsuario y clave correcta. Mostrando los datos de la
cuenta...\n")
                print("Nombre: ",fila["nombre"])
                print("Apellido: ",fila["apellido"])
                print("Cedula: ",fila["cedula"])
                print("Usuario: ",fila["usuario"])
                print("Posicion Consolidada: ",fila["posicion consolidada"],"\n")
    if existe_usuario == False:
        print("\nEl usuario introducido no se encuentra registrado en el
sistema.\n")
    else:
        if existe_clave == False:
 print("\nLa clave introducida es incorrecta...\n")
```

Seguidamente está el método **realizarTransferencia()**, el cual pedirá al usuario introducir su apodo y contraseña, las cuales se validaran al momento de introducirlas, una vez verificadas se pedirá al usuario introducir la cédula de la persona a la que desea realizar la transferencia, haciendo un llamado al submetodo transferirACedula(), el cual enviará como argumento: la persona que se busca en la variable global database(), la cédula ingresada de la persona a transferir, y el vector global "database" para manejar las cédulas validas dentro del banco.

```
def realizarTransferencia():
    print("\nHa seleccionado la opcion 5. (Realizar una transferencia
bancaria)\n")
    existe_usuario = False
    existe_clave = False
    usuario = input("Introduzca un nombre de usuario existente: ")
    for fila in database:
        if fila["usuario"] == usuario:
            existe_usuario = True
            print("\nUsuario encontrado\n")
            clave = input("Introduzca la clave respectiva: ")
            if fila["clave"] == clave:
                existe_clave = True
                print("\nBienvenido! ",fila["nombre"]," ",fila["apellido"])
                cedula = input("\nIntroduzca la cedula de la persona a la que
desea realizar la transferencia: \n")
```

```
                    while not cedula.isdigit():
                        print("La cedula no debe tener caracteres especiales ni
    letras.\n")
                        cedula = input("Cedula: ")
                    print("\nCedula procesada...")
                    transferirACedula(fila,cedula,database)
        if existe_usuario == False:
            print("\nEl usuario introducido no se encuentra registrado en el
    sistema.")
        else:
            if existe_clave == False:
         print("\nLa clave introducida es incorrecta...")
```

El submódulo **transferirACedula ()** validará si el parámetro "cedula" se encuentra dentro del sistema delbanco, de ser así se mostrará el nombre y apellido de la persona a transferir y se solicitará un monto atransferir, el cual deberá ser mayor a 0 y menor al saldo disponible, posteriormente verificado el proceso, se restará el dinero transferido al usuario y se sumará el dinero transferido al destinatario, a su vez se mostrará el monto transferido y el saldo actual del usuario.

```
        def transferirACedula(persona,cedula,database):
            existe_cedula = False
            for fila in database:
                if fila["cedula"] == cedula:
                    existe_cedula = True
                    print("\nCedula encontrada.\n")
                    print("Nombre: ",fila["nombre"])
                    print("Apellido: ",fila["apellido"])
                    monto = float(input("\nIntroduzca el monto a transferir: "))
                    if monto > 0 and monto <= persona["posicion consolidada"]:
                        persona["posicion consolidada"] -= monto
                        fila["posicion consolidada"] += monto
                        print("\nMonto transferido: ",monto)
                        print("Saldo disponible en su cuenta: ",persona["posicion
    consolidada"])
                    else:
                        print("\nIntroduzca un monto correcto y/o que no supere su saldo
    disponible en la cuenta.\n")
            if existe_cedula == False:
    print("\nLa persona a la cual desea realizarle una transferencia no se encuentra en
el sistema.\n")
```

Por último, está el método principal **main()**, el cual será el menú principal de todos los demás métodos, aquí se pedirá al usuario que realice una acción dentro del banco, y cuando no quiera realizar otra acción, se procederá a salir del programa

```
    def main():

        continuidad = True

        print("Hola, bienvenido/a al banco nacional, a continuacion le presentare las
    opciones que tenemos disponibles:\n")

        while continuidad:
```

```
        try:
            print("\n\nIntroduzca su opcion marcando cualquiera de los numeros
    indicados.")
            print("1.- Crear una Cuenta.")
            print("2.- Realizar un Deposito Bancario.")
            print("3.- Realizar un Retiro")
            print("4.- Consultar los datos de una cuenta.")
            print("5.- Realizar una transferencia a otro usuario.")
            print("6.- Salir del programa.")
            opcion = int(input("Opcion > "))
            if opcion == 1:
                crearCuenta()
            elif opcion == 2:
                depositoBancario()
                continue
            elif opcion == 3:
                retiroBancario()
                continue
            elif opcion == 4:
                consultarCuenta()
            elif opcion == 5:
                realizarTransferencia()
                continue
            elif opcion == 6:
                print("\nHasta luego!")
                continuidad = False
            else:
                raise Exception("Dato introducido invalido. Intente de nuevo.")
        except Exception as error:
            print("\nERROR: ",error)
    main()
```

Como elementos a mejorar en el programa en un futuro, se puede hacer uso de la refactorización de código para hacerlo más entendible y más compacto (menos líneas de código), además de que se puede implementar la Programación Orientada a Objetos (POO), para la resolución del problema.