



Universidad Fidélitas

Proyecto: Cajero Automático

Dairon Janaikel Arias Garita

María Laura Murillo Bravo

Gabriel Gerardo Hernández Otarola

Primer Avance

Curso: Programación Básica

Código de curso: SC-115

Profesor: Álvaro Camacho Mora

Fecha: 25 de abril de 2023

Índice

Introducción.....	3
Objetivos.....	3
Objetivo General.....	3
Objetivos Específicos.....	3
Desarrollo	4
Descripción de los módulos implementados.....	5
Menú Principal.....	5
Requerimientos del sistema	16
Bibliotecas/Estructuras complementarias (si aplica).....	16
Manual de Usuario.....	18
Conclusiones.....	24
Bibliografía.....	24

Introducción

En la evolución del ser humano, la comercialización y economía han sido de gran importancia en el asentamiento de las diferentes culturas alrededor del mundo. Estas derivaron la creación de distintas monedas o formas de pago, para así mantener buenas relaciones entre las personas. Asimismo, con el paso del tiempo, la creación de nuevas tecnologías permitió que se desarrollaran nuevos sistemas con el fin de hacer trámites monetarios, tales como los cajeros automáticos. En la actualidad, los cajeros permiten acciones como el retiro de dinero y otras transacciones. A modo de resumen, el presente escrito tiene como función desarrollar la programación de un cajero con ayuda del código Python.

Objetivos

Objetivo General

1. Crear un programa que asemeje las funcionalidades de un cajero automático, así como un manual para su uso, implementando todos los conocimientos vistos en el curso de Programación Básica.

Objetivos Específicos

1. Realizar un algoritmo que permita cumplir las diversas funciones correspondientes a las de un cajero automático utilizando los conocimientos aprendidos en el curso de Programación Básica.
2. Ejecutar el código de un cajero automático con el fin de elaborar un programa funcional y similar a los de la realidad, aplicando los conocimientos aprendidos en el curso de Programación Básica
3. Mostrar los pasos a seguir para el uso adecuado del programa y con ello comprender los conocimientos aprendidos en el curso de Programación Básica.

Desarrollo

El problema que se presenta en este escenario es el desarrollo de un software de última generación para el manejo de cajeros automáticos, que pueda ofrecer una experiencia de usuario fluida e intuitiva al tiempo que se garantiza la seguridad y la integridad de las transacciones financieras realizadas.

Para abordar este problema en el lenguaje de programación Python, existen varias estructuras de software que podrían resultar útiles; como funciones, listas, diccionarios y bucles. Algunas potenciales estructuras que podrían ayudar en el desarrollo de software de cajeros automáticos son las siguientes:

1. **Funciones:** se pueden definir diferentes funciones para realizar distintas tareas en el software, como la validación de los datos de usuario, la realización de transacciones, la actualización de saldos, etc. Algunas funciones específicas podrían incluir "retirar_efectivo", "depositar_efectivo()", "pagar_servicio()", "nuevo_usuario()" y "actualizar_saldo()".
2. **Listas:** Se pueden utilizar diferentes listas para diferentes propósitos. Un inventario de artículos se puede utilizar para reservar datos sobre los clientes y sus transacciones. Esta información puede incluir nombres de usuario, contraseñas, saldos de cuenta, movimientos ejecutados y más. Inventarios distintos también pueden servir a diferentes objetivos que apuntan a un amplio alcance; como un inventario para clientes o uno designado exclusivamente para actividades de transacciones, como una lista de usuarios, una lista de movimientos y una lista de servicios.
3. **Diccionarios:** los diccionarios son una estructura de datos útil para almacenar información relacionada con los usuarios, como sus nombres, apellidos, números de identificación, números de cuenta, etc. También se pueden utilizar diccionarios para almacenar información sobre los servicios disponibles y las transacciones realizadas.
4. **Bucles:** Los bucles son una herramienta valiosa para realizar tareas repetitivas como verificar datos de usuario, realizar transacciones y actualizar saldos. Se puede implementar un bucle while para solicitar repetidamente la contraseña del usuario hasta que coincida con lo que se espera, o alternativamente, utilizar un bucle for para iterar a través de una lista de usuarios y actualizar el saldo de cada individuo en consecuencia.

Descripción de los módulos implementados

Menú Principal

Problema

1. Presentará las opciones que se pueden realizar en el cajero.

Algoritmo

1. Presentar al usuario las opciones que tiene para ejecutar en el programa (Registrar nuevo usuario).
2. Estructura de decisión para identificar la opción que ha escogido el usuario.

Estructuras de Software

1. While, True, variable, int, input.
2. If, igual que (==).

1. Registrar nuevo usuario

1. Solicitud de número de cédula

1. Problema

1. Permitirá almacenar el número de cédula del nuevo usuario.

2. Algoritmo

1. Imprimir un mensaje que le indique al usuario, la sección donde se está.
2. Preguntar al usuario su número de cédula y se le indica que debe ser y únicamente nueve dígitos.
3. Estructura de decisión para indicar error si hay más o menos de nueve dígitos.
4. Estructura de decisión para identificar si el usuario está previamente registrado. Si está registrado, se le dirá al usuario y se producirá un error.

5. Se tendrán máximo tres intentos para ingresar la cédula del nuevo usuario.

3. *Estructuras de Software*

1. Print
2. Def, variable, input, while, return.
3. If, break, else, print.
- 4.
5. Variable, if, and, igual que (==), menos igual (-=)

2. Solicitud del nombre

1. *Problema*

1. Parte del código que permitirá solicitar el nombre del nuevo usuario.

2. *Algoritmo*

1. Imprimir un mensaje que le indique al usuario la sección donde se encuentra.
2. Preguntar al usuario el nombre completo para el nuevo usuario.

3. *Estructuras de Software*

1. Print, def, return
2. Variable, input.

3. Escogencia del PIN

1. *Problema*

1. Sección del código que le permitirá al usuario crear un PIN, de forma que a futuro pueda ingresar a su cuenta.

2. *Algoritmo*

1. Imprimir un mensaje que permita orientar al usuario del siguiente paso a realizar.
2. Preguntar al usuario el PIN de cuatro dígitos que desea crear.
3. Se le pedirá al usuario que reingrese el PIN para su verificación.
4. Comprobar que ambos, el PIN inicial y el de verificación sean iguales.

3. *Estructuras de Software*

1. Print, def, return
2. Variable, int, input
3. While, True, variable, int, input
4. If, igual que (=), break, else, print, len, .isdigit()

4. Depósito obligatorio

1. *Problema*

1. Sección del código donde se realiza un depósito con el fin de activar y crear la nueva cuenta o nuevo usuario.

2. *Algoritmo*

1. Imprimir un mensaje que le permita al usuario ubicarse en el proceso de creación del nuevo usuario.
2. Presentar un menú para que el usuario elija en qué moneda desea hacer el depósito.
3. Calcular el tipo de cambio de acuerdo a la moneda.
4. Pedirle al usuario un depósito igual o mayor a la equivalencia de 100000 colones.
5. Permitirle al usuario 3 intentos para hacer el depósito si no, volver al menú.

3. *Estructuras de Software*

1. Print, def, return
2. Variable, int, input, while, true.
3. If, igual que (==), variables, división entera (/), break
4. Print, float, input, variable, while.
5. Variable, while, mayor que (>), if, mayor o igual que (>=), break, menos igual (<=), else, print

5. Seleccionar automática y aleatoriamente tres servicios al usuario

1. *Problema*

1. Sección de código que permite establecer el estado de un servicio (activo o inactivo) de forma automática y aleatoria para un nuevo usuario.

2. *Algoritmo*

1. Establecer el sitio donde se guardarán los datos.
 2. Elegir aleatoriamente si el servicio está activo o inactivo.
 3. Seleccionar un monto aleatorio a cada servicio.
3. Estructuras de Software
 1. Variable, `os.path.join()`, `open`, 'a'
 2. Variable, `random.randint()`, `str`
 3. Variable, `random.randint()`, `str`
6. Guardar la información del nuevo usuario
 1. Problema
 1. Sección del código que permite almacenar los datos del usuario para poder acceder a ellos posteriormente.
 2. Algoritmo
 1. Verificar que la carpeta no exista.
 2. Crear una carpeta por usuario para guardar sus datos de servicios y cuentas.
 3. Crear un archivo de texto para almacenar la información de los usuarios para que sea de fácil acceso.
 3. Estructuras de Software
 1. If not, `os.path.exists()`, else, `print`
 2. Variable, `os.makedirs()`, `os.path.join()`, `open`, "a", `.write()`, `.close()`
 3. Variable, `open`, "a", `.write()`, `.close()`
7. Regresar a menú
 1. Problema
 1. Sección del código que permite que el programa vuelva al menú.
 2. Algoritmo
 1. Volver al menú principal
 3. Estructuras de Software
 1. Break

2. Usuario Registrado

1. Cargar información de usuarios registrados

1-Problema

1. Sección del código que cargará los números de cédula, nombres y pines, del archivo de “UsuariosYPines” a una matriz para que usarse esta información.

2. Algoritmo

1. Utilizando la dirección del archivo, abrir, leerlo y colocar los datos dentro de una matriz. (Esto tanto para el archivo de los usuarios y pines como para el de los tipos de cambio.)

3. Estructuras de Software

1. Os.path.exists, variables, .readlines(), .close(), una matriz, for i in range, if, else, open(), comentarios.

2. Verificar que exista al menos un usuario

1. Problema

1. Sección del código que revisará si existen usuarios registrados, de lo contrario no podrá acceder a la opción “2” del menú principal.

2. Algoritmo

1. A la hora de digitar la opción “2” se verificará si hay usuarios registrados, de lo contrario no se podrá acceder a esta sección.

3. Estructuras de Software

1. Def, if, else, print, matriz, variable, comentarios

3. Autenticación de usuarios

1. Problema

1. Sección del código donde se le solicitará al usuario ingresar su número de cédula y pin. Si el pin coincide al ingresado al registrarse, se procederá.

2. Algoritmo:

1. El usuario tendrá 3 intentos para ingresar un número de cédula válido.

2. Si este está registrado, tendrá 3 intentos para ingresar el pin correspondiente a la cédula.
3. Si se ingresa el PIN correcto, se le presentará al usuario un submenú.

3. *Estructuras de Software*

1. Variable, while, if, else, return, print, input, os.path.exists, and, len(), comentarios.

4. Darle un Mensaje de bienvenida al usuario

1. Retirar dinero

1. *Problema*

Sección del programa que le permitirá al usuario retirar dinero de sus cuentas (dólares, colones, bitcoins).

2. *Algoritmo*

1. Se ingresa a la carpeta del usuario ingresado en la opción “2”.
2. Se ingresa al archivo “saldos” de este usuario y se extraen los montos de sus cuentas para ser utilizados.
3. De acuerdo a la opción escogida se hace un retiro de la cuenta, si, y solo si, hay suficientes fondos para hacerlo (tendrá solo tres intentos para retirar dinero, luego se volverá al menú principal).
4. Se actualizan los montos tanto en la matriz como en el archivo “saldos”.

3. *Estructuras de Software*

1. Def, variable, os.path.join, open(), .readlines(), .close(), una arreglo, while, True, print, input, if, elif, else, print, break, float, .write(), comentarios.

2. Depositar dinero

1. *Problema*

1. Sección del programa que le permitirá al usuario depositar dinero en cualquiera de sus cuentas (dólares, colones, bitcoins).

2. *Algoritmo*

1. Se ingresa a la carpeta del usuario ingresado en la opción “2”.
2. Se ingresa al archivo “saldos” de este usuario y se extraen los montos de sus cuentas para ser utilizados.
3. De acuerdo a la cuenta escogida se hace un depósito de la cantidad definida por el usuario.
4. Se actualizan los montos tanto en la matriz como en el archivo “saldos”.

3. *Estructuras de Software*

1. Def, os.path.join(), open(), .readlines(), .close(), arreglo, while, True, print, variable, input, float, str, if, elif, else, break, comentarios.

3. Ver saldo actual

1. *Problema*

1. Sección del código que le presentará al usuario el saldo en cada una de sus cuentas (dólares, colones, bitcoins).

2. *Algoritmo*

1. Se carga la información para ver el saldo del usuario

3. *Estructuras de Software*

1. Def, variable, os.path.join(), open(), .readlines(), .close(), lista, print, comentarios.

4. Pagar servicios

1. *Problema*

1. Sección del programa que le permite al usuario pagar con cualquiera de sus cuentas (dólares, colones, bitcoins) los diferentes servicios asignados de forma aleatoria al inicio.

2. *Algoritmo*

1. Se ingresa a la carpeta del usuario escogido.
2. Se le muestra un menú al usuario para que escoja el menú que desea pagar.

3. De acuerdo a la opción ingresada, se abre el servicio correspondiente a la opción.
4. Se muestra el monto a pagar.
5. El usuario debe escoger de cuál cuenta se obtendrán los fondos para hacer el pago.
6. Si el monto a pagar es menor se procederá a hacer el pago, de lo contrario no se podrá.
7. Se guardarán los cambios en las cuentas.

3. *Estructuras de Software*

1. Def, variable, os.path.join(), open(), .readlines(), .close(), arreglo, while, True, print, input, if, int, for i in range, len, float, str, .write(), break, else, elif comentarios.

5. Compra y venta de divisas

1. *Problema*

1. Sección del código que le permitirá al usuario comprar o vender divisas y escoger la cuenta de origen de los fondos para hacerlo (dólares, colones, bitcoins).

2. *Algoritmo*

1. Se ingresa a la carpeta del usuario ingresado en la opción “2”.
2. Se ingresa al archivo “saldos” de este usuario y se extraen los montos de sus cuentas para ser utilizados.
3. Se le mostrará al usuario un menú de las conversiones que puede realizar y una última opción para salir (volver al menú principal).
4. El usuario deberá ingresar el monto que desea convertir y la cuenta de origen de los fondos.
5. Si hay fondos suficientes, se realizará la conversión y se volverá al submenú.
6. Se actualizan los montos tanto en la matriz como en el archivo “saldos”.

3. *Estructuras de Software*

1. Def, variable, os.path.join(), open(), .readlines(), .close(), arreglo, while, Trues, print, input, if, break, float, str, .write(), else, comentarios.

6. Eliminar usuario

1. *Problema*

1. Sección del programa que le permite eliminar su cuenta si así lo desea.

2. *Algoritmo*

1. Al ingresar esta opción el usuario deberá ingresar su pin.
2. Una vez ingresado, el usuario registrado será borrado.

3. *Estructuras de Software*

1. Def, variable, getpass.getpass, prompt, if, is, True, open(), for i in range, len, .write(), .close(), print, else, comentarios.

7. Salir

1. *Problema*

1. Sección del programa que lo devuelve al menú principal.

2. *Algoritmo*

1. Se carga la información para sacar al usuario.

3. *Estructuras de Software*

1. break

3. Configuración Avanzada

1. Solicitud de PIN especial

1. *Problema*

1. El problema en este caso es cómo implementar una solicitud de PIN especial que permita a ciertos usuarios acceder a características adicionales del programa.

2. *Algoritmo*

1. Se puede implementar un algoritmo que solicite al usuario un PIN especial al inicio del programa. Si el PIN es correcto, se mostrarán

las características adicionales del programa. Si el PIN es incorrecto, se mostrará el menú normal.

3. *Estructuras de Software*

1. Se puede utilizar una estructura de software que permita almacenar y verificar el PIN especial. Por ejemplo, se puede utilizar una base de datos o un archivo de configuración para almacenar el PIN y una función de verificación para verificar si el PIN ingresado es correcto.

2. Menú

1. Eliminar usuario

1. *Problema*

1. El problema en este caso es cómo implementar una funcionalidad que permita eliminar un usuario del programa.

2. *Algoritmo*

1. Se puede implementar un algoritmo que solicite al usuario el nombre de usuario que se desea eliminar. Si el usuario existe, se eliminará del programa. Si el usuario no existe, se mostrará un mensaje de error.

3. *Estructuras de Software*

1. Se puede utilizar una estructura de software que permita almacenar los usuarios del programa. Por ejemplo, se puede utilizar una base de datos o un archivo de texto para almacenar los usuarios y una función de eliminación para eliminar el usuario seleccionado.

2. Modificar tipos de cambio

1. *Problema*

1. El problema en este caso es cómo implementar una funcionalidad que permita modificar los tipos de cambio de una moneda a otra.

2. *Algoritmo*

1. Se puede implementar un algoritmo que solicite al usuario los tipos de cambio a modificar. Si los tipos de cambio existen, se modificarán. Si los tipos de cambio no existen, se crearán. Si el usuario desea eliminar un tipo de cambio, se eliminará.

3. Estructuras de Software

1. Se puede utilizar una estructura de software que permita almacenar los tipos de cambio. Por ejemplo, se puede utilizar una base de datos o un archivo de texto para almacenar los tipos de cambio y una función de modificación para modificar los tipos de cambio seleccionados.

3. Salir

1. Problema

1. El problema en este caso es cómo implementar una funcionalidad que permita salir del programa.

2. Algoritmo

1. Se puede implementar un algoritmo que muestre un mensaje de despedida y cierre el programa.

3. Estructuras de Software

1. Se puede utilizar un break para salir del ciclo del menú y para el mensaje de despedida por ejemplo se puede hacer un print con el nombre del usuario que este usando el sistema.

4. Salir

1. Problema

1. Sección del código que permite salir o apagar el programa.

2. Algoritmo

1. Salir del programa.

3. Estructuras de Software

1. Break

Requerimientos del sistema

El programa no requiere de un sistema muy avanzado para ser corrido, simplemente se necesita usar una computadora que permita instalar Python.3. Para ejecutar el programa, podrá hacerlo en el CMD o “Command Prompt” al darle doble clic sobre el archivo .py llamado “programaPrincipal”. Asimismo, puede utilizar algún editor de código Python como “IDLE” de Python y PyCharm.

Bibliotecas/Estructuras complementarias (si aplica)

Biblioteca os.

- ¿Porque y para que se usa?

Se utilizó para verificar la existencia de archivos o carpetas.

- En qué parte del código lo usó

Se utilizó en diversas secciones donde era necesario comprobar la existencia de un archivo. Por ejemplo, en el programa principal se usó para verificar que el archivo de “Usuarios_y_Pines” y “config” existen.

- Como se instala

La biblioteca funciona, importándola con “import os”; y utilizándola al escribir “os.path.exists()”, en el paréntesis va el nombre del archivo.

- Fuentes bibliográficas

Documentación de Python. (s. f.). os — Interfaces misceláneas del sistema operativo.
<https://docs.python.org/es/3.10/library/os.html>

Documentación de Python. (s. f.). os.path — Manipulaciones comunes de nombre de ruta.
<https://docs.python.org/es/3.10/library/os.path.html#module-os.path>

Biblioteca random.

- ¿Porque y para que se usa?

La biblioteca random se utiliza para colocar un valor aleatorio de acuerdo a un rango establecido.

- En qué parte del código lo usó

La biblioteca random se utilizó para asignar valores aleatorios a los servicios de los usuarios. Por ejemplo, si el servicio está activo o no, y el monto que debe el usuario de ese servicio.

- Como se instala

La biblioteca funciona, importándola con “import random”; y utilizándola al escribir “random.randint ():”, en el paréntesis va el rango; ejemplo, (0,1) o (1000, 100000).

- Fuentes bibliográficas

Python Documentation. (s. f.). random — Generate pseudo-random numbers.
<https://docs.python.org/3/library/random.html>

Manual de Usuario

Para el uso adecuado del programa, se ha realizado el siguiente manual el cual guiará al usuario por una serie de pasos para utilizar el programa.

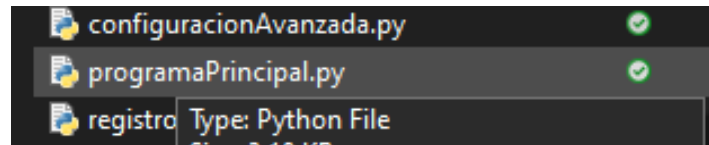


Figura 1.

1. En primer lugar, el usuario tendrá que abrir el archivo donde podrá ejecutar el programa. Dando doble clic al nombre del archivo, como se muestra en *figura 1*.

```
Bienvenido.  
Menú Principal  
Por favor digite el número de acuerdo a la opción que aparece en pantalla.  
1. Registrar nuevo usuario  
2. Usuario Registrado  
3. Configuración Avanzada  
4. Salir  
Digite su opción aquí:
```

Figura 2.

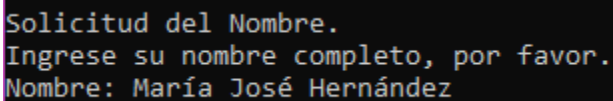
2. Seguidamente, se le mostrará un menú con las diferentes opciones que se pueden realizar en el programa (a este se le conocerá como menú principal). Así como se observa en *figura 2*, el usuario debe digitar el número que corresponde a la opción que desea realizar.

Registrar nuevo usuario

```
Digite su opción aquí: 1  
Solicitud de número de cédula.  
Ingresa el número de cédula, por favor. Nota: Esta no puede ser mayor o menor a 9 dígitos.  
Cédula: 102340567  
Cédula procesada correctamente.
```

Figura 3.

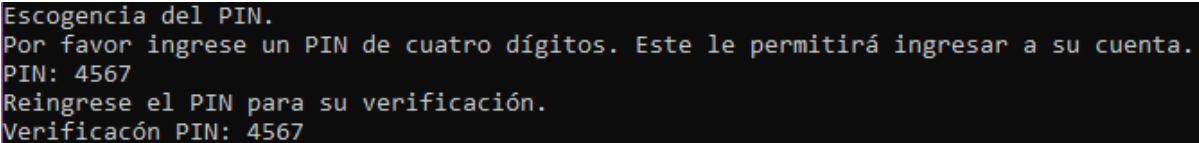
- a. Si se digita la opción “1” del menú principal, la pantalla que se le mostrará será como lo mostrado en *figura 3*. Aquí deberá digitar el número de cédula del usuario que desea crear, asimismo es importante que lo ingresado dígitos y no más ni menos de 9. Si se equivoca al ingresar el número de cédula podrá hacerlo hasta tres veces, luego se retornará al menú principal.



```
Solicitud del Nombre.  
Ingrese su nombre completo, por favor.  
Nombre: María José Hernández
```

Figura 3.1

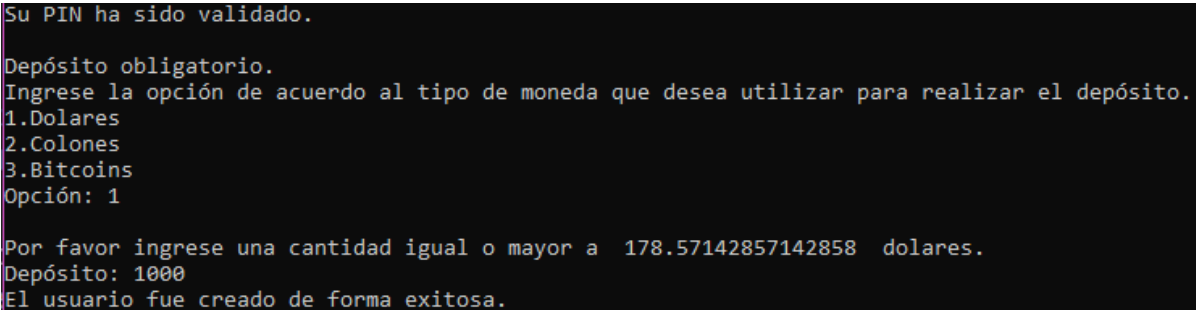
- i. Una vez ingresado el número de cédula, se le hará solicitud del nombre del nuevo usuario.



```
Escogencia del PIN.  
Por favor ingrese un PIN de cuatro dígitos. Este le permitirá ingresar a su cuenta.  
PIN: 4567  
Reingrese el PIN para su verificación.  
Verificación PIN: 4567
```

Figura 3.2

- ii. Luego de escribir el nombre del usuario, se procederá con la creación de un PIN. Este debe ser de cuatro dígitos. **Nota: en la figura 3.2 se puede observar el PIN, no obstante, este es solo un ejemplo, el PIN estará escondido. Al ingresar datos parecerá que no se escribe, sí lo hace, solo no se muestra en su pantalla.**
- iii. Para continuar, se deberá reingresar el PIN anterior.



```
Su PIN ha sido validado.  
Depósito obligatorio.  
Ingrese la opción de acuerdo al tipo de moneda que desea utilizar para realizar el depósito.  
1.Dolares  
2.Colones  
3.Bitcoins  
Opción: 1  
Por favor ingrese una cantidad igual o mayor a 178.57142857142858 dolares.  
Depósito: 1000  
El usuario fue creado de forma exitosa.
```

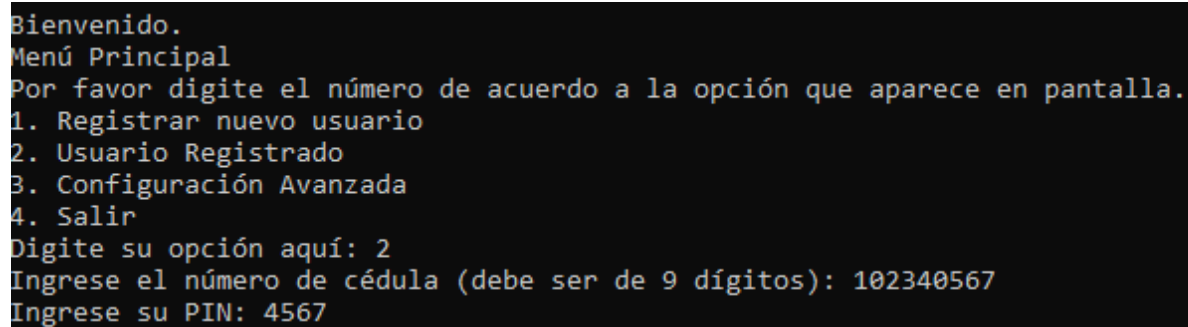
Figura 3.3

- iv. Al ingresar el PIN de verificación y sea este igual al anterior, se le presentara una pantalla similar a la *figura 3.4*. En esta tendrá que realizar un depósito

obligatorio para activar la nueva cuenta; podrá escoger la moneda en la que quiere hacer el depósito.

- v. De acuerdo al tipo de cambio, el monto mínimo del depósito cambiará, por ello es importante leer el monto mínimo que es pedido en la indicación como se observa en figura 3.5. En este ejemplo, el usuario a escogido hacer un depósito en dólares, por lo que deberá introducir un monto mayor a 180 dólares.
- vi. Al finalizar con estos pasos, la sección de registrar nuevo usuario ha concluido, por lo que la siguiente pantalla será el menú principal, así como se ve en *figura 2*.

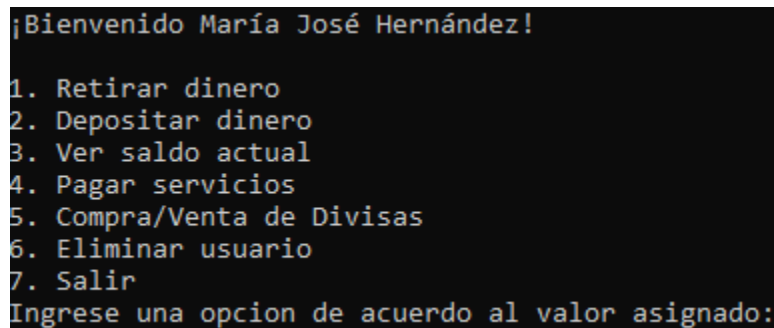
Usuario registrado



```
Bienvenido.  
Menú Principal  
Por favor digite el número de acuerdo a la opción que aparece en pantalla.  
1. Registrar nuevo usuario  
2. Usuario Registrado  
3. Configuración Avanzada  
4. Salir  
Digite su opción aquí: 2  
Ingrese el número de cédula (debe ser de 9 dígitos): 102340567  
Ingrese su PIN: 4567
```

Figura 4.

- b. Si se digita la opción “2” del menú principal, la pantalla que se le mostrará será como lo mostrado en *figura 4*. Aquí deberá ingresar su número de cédula, así como el pin.



```
¡Bienvenido María José Hernández!  
1. Retirar dinero  
2. Depositar dinero  
3. Ver saldo actual  
4. Pagar servicios  
5. Compra/Venta de Divisas  
6. Eliminar usuario  
7. Salir  
Ingrese una opcion de acuerdo al valor asignado:
```

Figura 4.1

- i. Al ingresar su número de cédula y pin correctamente, podrá realizar cualquiera de las siguientes opciones ingresando el número de acuerdo a la opción. Solo deberá seguir los pasos que se le indican. Para salir de este menú y volver al principal, digite “7”.

```
Ingrese una opcion de acuerdo al valor asignado: 1
1. Dolares
2. Colones
3. Bitcoins
Por favor ingrese la cuenta de la que desea retirar dinero.
Opción:
```

Figura 4.2

- ii. Esta es la pantalla al retirar dinero. Deberá ingresar el valor correspondiente a la cuenta de donde desea retirar el dinero.

```
Ingrese una opcion de acuerdo al valor asignado: 2
1. Dolares
2. Colones
3. Bitcoins
Por favor ingrese la cuenta a la que desea hacer el depósito.
Opción:
```

Figura 4.3

- i. Esta es la pantalla al depositar dinero. Deberá ingresar el valor correspondiente a la cuenta a la que desea hacer el depósito.

```
Ingrese una opcion de acuerdo al valor asignado: 3
Su saldo actual es:
Dolares: 1212.0
Colones: 568302.0
Bitcoins: 0.9235
```

Figura 4.4

- ii. Esta es la pantalla para ver el saldo.

```
Ingrese una opcion de acuerdo al valor asignado: 4
Ingrese el número correspondiente al nombre del servicio a pagar:
1.Electricidad
2.Agua
3.Telefonía
4.Internet
5.Impuestos
6.Colegios Profesionales
7.Tarjeta de crédito
8.Salir
Opción:
```

Figura 4.5

- iii. Esta es la pantalla al ingresar la opción para pagar los servicios. Aquí deberá ingresar la opción que corresponda al servicio que desea pagar, si el escogido está inactivo se le hará saber, de lo contrario se le indicará el monto a pagar. Podrá escoger una de sus cuentas (dólares, colones o bitcoins) para cancelar el monto del servicio. Para salir de este menú y volver al principal, digite “8”.

```
Ingrese una opcion de acuerdo al valor asignado: 5
Ingrese el número correspondiente a la operación que desea realizar:
1.Compra de Colones
2.Venta de Colones
3.Compra de Dólares
4.Venta de Dólares
5.Compra de Bitcoins
6.Venta de Bitcoins
7.Salir
Opción:
```

Figura 4.6

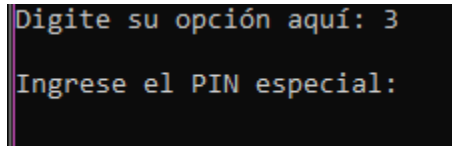
- i. Al ingresar la opción “5”, podrá comprar y vender divisas. Deberá ingresar la opción que corresponda a la conversión que desea realizar. Luego tendrá que escoger una cuenta origen de los fondos para proceder. Para salir este submenú digite “7”.

```
Ingrese una opcion de acuerdo al valor asignado: 6
Ingrese su PIN: 4567
Usuario eliminado con éxito.
```

Figura 4.7

- ii. Al ingresar la opción “6”, se procederá a eliminar el usuario. Para ello, deberá ingresar su pin y se eliminarán cuentas, servicios y la cuenta. Luego, regresará automáticamente al menú principal.

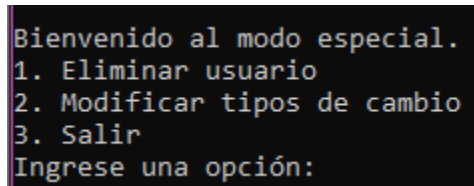
Configuración Avanzada



```
Digite su opción aquí: 3
Ingrese el PIN especial:
```

Figura 5.

- c. Se le solicitará al usuario ingresar un pin especial. Si el pin ingresado es correcto, se mostrará el menú de opciones. Dicho pin se crea automáticamente en un archivo (config.txt).



```
Bienvenido al modo especial.
1. Eliminar usuario
2. Modificar tipos de cambio
3. Salir
Ingrese una opción:
```

Figura 5.1

- i. Si el PIN es ingresado correctamente, se le mostrará el menú presente en la *figura 5.1*. Deberá ingresar el valor que corresponda al valor que desea ejecutar. Si se ingresa “1”, podrá eliminar un usuario, se le solicitará la cédula del usuario a eliminar. Si digita “2”, podrá modificar los tipos de cambio utilizados en la opción de Usuarios Registrados. Por último, si digita “3” volverá al menú principal.

Salir

- d. Si se digita la opción “4” en el menú principal, se terminará el programa, es decir el final del uso del programa.

Conclusiones

1. Con el desarrollo del proyecto, se ha podido desarrollar una diversidad de conocimientos que permiten la elaboración de un algoritmo y con el orden adecuado para que este funcione. De esta manera, se puede decir que no solo se aprendieron estructuras sino también un pensamiento lógico.
2. Por otro lado, a la hora de ejecutar el algoritmo se concluyó que muchas veces se debe ser cuidadoso con los posibles errores o vacíos que pueden quedar en el código, debido a que esto cambia la funcionalidad del programa.
3. Por último, una vez terminado el proyecto, al realizar el manual el cual permite observar los distintos pasos que son necesarios para utilizar el algoritmo, permitieron reflexionar sobre todas aquellas estructuras que son necesarias para llevar a cabo un procedimiento que en pseudocódigo parece ser bastante sencillo.

Bibliografía

- Documentación de Python. (s. f.). *os — Interfaces misceláneas del sistema operativo*.
<https://docs.python.org/es/3.10/library/os.html>
- Documentación de Python. (s. f.). *os.path — Manipulaciones comunes de nombre de ruta*.
<https://docs.python.org/es/3.10/library/os.path.html#module-os.path>
- GeeksforGeeks. (2022). *getpass and getuser in Python Password without echo*.
<https://www.geeksforgeeks.org/getpass-and-getuser-in-python-password-without-echo/>
- Python Documentation. (s. f.). *getpass — Portable password input*.
<https://docs.python.org/3/library/getpass.html>
- Python Documentation. (s. f.). *random — Generate pseudo-random numbers*.
<https://docs.python.org/3/library/random.html>