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: 7 de marzo de 2023

Índice

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 las diferentes funciones que realiza un cajero automático para la elaboración de un programa que las ejecute.

Objetivos Específicos

1. Realizar un algoritmo que permita cumplir las diversas funciones correspondientes a las de un cajero automático.
2. Ejecutar el código de un cajero automático con el fin de elaborar un programa funcional y similar a los de la realidad.
3. Mostrar los pasos a seguir para el uso adecuado del programa.

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, intput.
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.randit(), 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

Darle al usuario la posibilidad de realizar trámites como depósitos y retiro de dinero, pagos de servicios, eliminación de cuenta, entre otros.

1. Cargar información de usuarios registrados
    1. *Problema*
        1. Cargar información de usuarios registrados desde un archivo de texto.
    2. *Algoritmo*
        1. Definir variables (variable, = igual, #comentario).
        2. Calcular la información de los usuarios (for i + condición, #comentario).
    3. *Estructuras de Software*
        1. #comentarios
2. Verificar que exista al menos un usuario
    1. *Problema*
        1. Verificar que exista al menos un usuario registrado.
    2. *Algoritmo*
        1. Verificar que exista un usuario registrado(#comentario,if not+ condición, print (no hay usuarios registrados)).
    3. *Estructuras de Software*
        1. #comentarios
3. Autenticación de usuarios
    1. *Problema*
        1. Autenticación de usuarios
    2. *Algoritmo*
        1. El sistema solicitara el número de cedula del usuario (int,input,variable, #comentario).
        2. El sistema deberá verificar que sea un numero de cedula valido y registrado (definir variables, while true, input, if).
        3. En caso de que sea invalido: Se le dará hasta un máximo de tres intentos para ingresar un numero de cedula valido. o Si se agotan

estos tres intentos, se devuelve al menú principal. (while, if, else, print).

4. Si la cedula es válida, se le solicitará el PIN ▪ Si se excede la cantidad máxima de tres intentos, se deberá volver al menú principal. (while, if, else, return)

3. *Estructuras de Software*
    1. #comentarios

4. Mensaje de bienvenida
    1. Retirar dinero
        1. *Problema*
            1. Retirar dinero.
        2. *Algoritmo*
            1. Se carga la información referente para retirar el dinero.
        3. *Estructuras de Software*
            1. Variables, while, print, input, if, elif, else, print, def, comentarios.
    2. Depositar dinero
        1. *Problema*
            1. Depositar dinero
        2. *Algoritmo*
            1. Se carga la información para depositar el dinero al usuario o usuarios.
        3. *Estructuras de Software*
            1. while, print, variable, input, if, elif, else, print, def, comentarios.
    3. Ver saldo actual
        1. *Problema*
            1. Ver saldo actual
        2. *Algoritmo*
            1. Se carga la información para ver el saldo del usuario
        3. *Estructuras de Software*

        1. while, print, variable, input, if, elif, else, print, def, comentarios.

4. Pagar servicios
    1. *Problema*
        1. Pagar servicios
    2. *Algoritmo*
        1. Se carga la información para pagar los servicios del usuario
    3. *Estructuras de Software*
        1. While, print, variables, input, if, elif, else, print, def, comentarios.
5. Compra/Venta de Divisas
    1. *Problema*
        1. Compra y venta de divisas
    2. *Algoritmo*
        1. Se carga la información para la compra y venta de divisas para el usuario.
    3. *Estructuras de Software*
        1. While, print, variables, input, if, elif, else, print, def, comentarios.
6. Eliminar usuario
    1. *Problema*
        1. Eliminar usuario
    2. *Algoritmo*
        1. Se carga la información para eliminar usuario .
    3. *Estructuras de Software*
        1. While, print, variables, input, if, elif, else, print, def, comentarios.
7. Salir
    1. *Problema*
        1. Salir
    2. *Algoritmo*

1. Se carga la información para sacar al usuario.

3. *Estructuras de Software*
    1. While, print, variables, input, if, elif, else, print, def, comentarios.

5.

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

Bibliotecas/Estructuras complementarias (si aplica)

- Biblioteca os.
- Biblioteca random.

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.

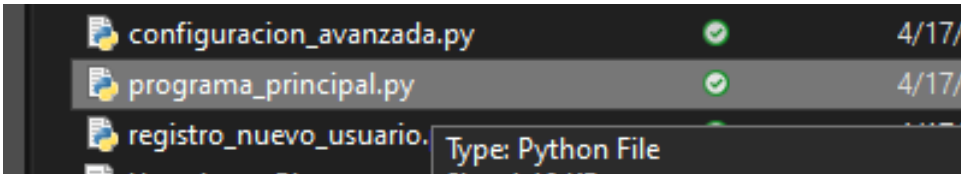

Figura 1.

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

```
Solicitud de número de cédula.
Ingrese el número de cédula, por favor. Nota: Esta no puede ser mayor o menor a 9 dígitos.
Cédula:
```

Figura 3.

   a. Si se digita la opción 1, 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 de número de cédula.
Ingrese el número de cédula, por favor. Nota: Esta no puede ser mayor o menor a 9 dígitos.
Cédula: 234567801
Cédula procesada correctamente.

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

Figura 3.1

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

```
Solicitud del Nombre.
Ingrese su nombre completo, por favor.
Nombre: Juan Vega Cascante

Escogencia del PIN.
Por favor ingrese un PIN de cuatro dígitos. Este le permitirá ingresar a su cuenta.
PIN:
```

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.

```
Escogencia del PIN.
Por favor ingrese un PIN de cuatro dígitos. Este le permitirá ingresar a su cuenta.
PIN: 1234
Reingrese el PIN para su verificación.
Verificacón PIN:
```

Figura 3.3

iii. Para continuar, se deberá reingresar el PIN anterior. (Este es solo un ejemplo por lo que aquí se observa el PIN digitado, en el programa este estará oculto.)

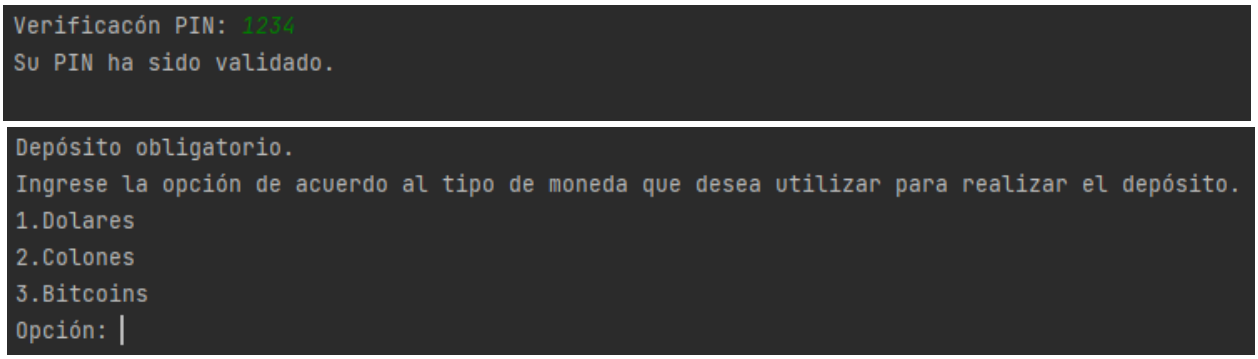

Figura 3.4

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.

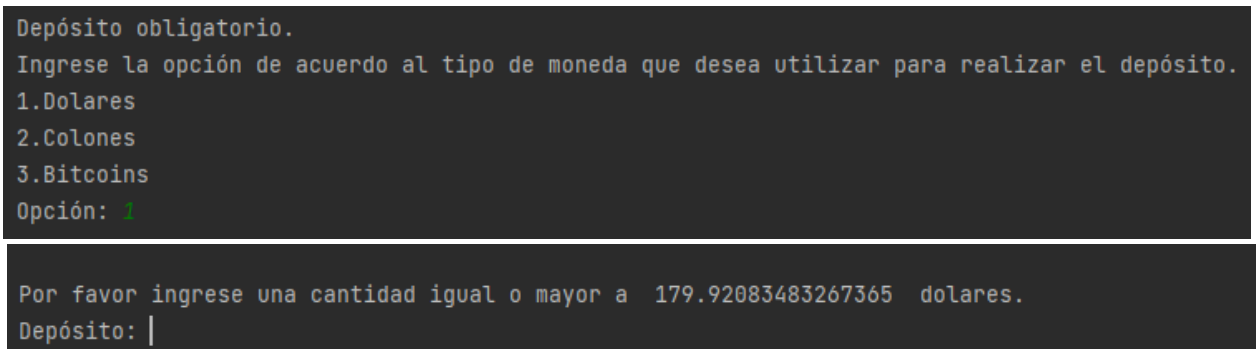

Figura 3.5

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

Figura 4.

b. Si se digita la opción 2, la pantalla que se le mostrará será como lo mostrado en *figura 4*.

Configuración Avanzada

Figura 5.

Descripción

La aplicación de Cambio de Divisas es un programa que permite a los usuarios administrar usuarios y modificar los tipos de cambio de diferentes monedas. El programa utiliza archivos de texto plano para almacenar los datos de usuarios y los tipos de cambio.

Funcionalidades del Programa

El programa tiene las siguientes funcionalidades:

4.1 Comprobar Existencia

Antes de comenzar a utilizar el programa, se verifica si existen los archivos de configuración y de datos de usuarios. Si no existen, se crean automáticamente.

4.2 Ingresar PIN Especial

El programa solicitará al usuario ingresar un PIN especial para acceder al modo especial. Si el PIN ingresado es correcto, se mostrará el menú de opciones para modificar los tipos de cambio.

4.3 Modificar Tipos de Cambio

En el modo especial, el usuario puede modificar los tipos de cambio de diferentes monedas. Se le presentará un menú con las opciones de tipos de cambio disponibles y podrá ingresar nuevos valores para los mismos. Los cambios se guardarán en un archivo de tipos de cambio.

4.4 Mostrar Menú

El programa muestra un menú principal con opciones para eliminar usuarios y modificar tipos de cambio. El usuario puede seleccionar una opción ingresando el número correspondiente.

4.5 Eliminar Usuario

El usuario puede eliminar un usuario ingresando el número de cédula del usuario a eliminar. El programa buscará el usuario en el archivo de datos de usuarios, lo eliminará y actualizará el archivo.

Pasos para Utilizar el Programa

Para utilizar el programa, siga los siguientes pasos:

5.1 Ejecute el código del programa en un entorno de desarrollo o en la línea de comandos de Python.

5.2 El programa verificará si existen los archivos de configuración y de datos de usuarios. Si no existen, se crearán automáticamente.

5.3 Se le solicitará al usuario ingresar un PIN especial. Si el PIN ingresado es correcto, se mostrará el menú de opciones para modificar los tipos de cambio.

5.4 En el menú de opciones, el usuario puede seleccionar la opción deseada ingresando el número correspondiente.

5.5 Para modificar los tipos de cambio, ingrese los nuevos valores cuando se le solicite.

5.6 Para eliminar un usuario, ingrese el número de cédula del usuario a eliminar cuando se le solicite.

5.7 El programa guardará automáticamente los cambios en los archivos correspondientes.

Consideraciones Adicionales

Es importante tener cuidado al modificar los tipos de cambio, ya que esto puede afectar la funcionalidad de la aplicación.

El PIN especial es necesario para acceder al modo especial y realizar modificaciones en los tipos de cambio. Mantenga este PIN seguro y no lo comparta con personas no autorizadas.

El programa guarda automáticamente los cambios realizados en los archivos de configuración y de datos de usuarios.

Salir

c. Si se digita la opción 4, se terminará el programa, es decir el final del uso del programa.

Conclusiones

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