UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA, 2023-2

Fundamentos de programación y Generación de Entornos Virtuales

Proyecto Final

INSTRUCTORES:
César Mauricio Ramos Villaseñor
Carolina Kennedy Villa

ALUMNOS:
ALFARO DOMÍNGUEZ PATRICIO
CHONG HERNÁNDEZ SAMUEL
RAMÍREZ HERNÁNDEZ ROCÍO

ÍNDICE **pág.**

INTRODUCCIÓN…….3

CLASES…….4

MAQUINA EXPENDEDORA……..4

PRODUCTO……5

ADMINISTRADOR……6

EXCEPCIONES……10

FUNCIÓN PRINCIPAL……11

MANUAL DE USUARIO…….15

EJECUCIÓN……..19

Introducción.

Encontramos máquinas expendedoras en casi cualquier ámbito privado o público, sean escuelas, oficinas, fábricas, etc. Esto debido a su comodidad y fácilidad de proveernos un “aperitivo” o “botana” para quitar el hambre. En estas máquinas se presentan objetos con características cuantitativas (o dicho de otra manera, cuentan con atributos), sean sabores, tamaños, cantidades, entre otras; además, cuentan con una interacción máquina-usuario la cual suele presentarse de la forma más amena posible en el panel con el que se puede escoger el producto.

Entre otros factores, podemos abstraer esta información y trasladarla a un paradigma de programación, en específico, **la programación orientada objetos.** Que tiene este mismo objetivo, acercar a como se expresan las cosas en la vida real por medio de la programación.

En el siguiente escrito se observa la construcción de un programa cuya estrcutructura se basa en el funcionamiento de una máquina expendedora, esto a través de la creación de clases, objetos con atributos y métodos de los mismos que permitirán un buen funcionamiento de la misma.

Clases.

Máquina expendedora.

Al estar en un paradigma orientado a objetos, el programa se basa en la creación de clases con atributos y métodos. Primero, se comenzará explicando la clase principal de este proyecto, la de la Máquina Expendedora.

En esta clase podemos encontrar un constructor vacío, su visualización se verá expresada a través de una lista:

```
class MaquinaExpendedora():
    """
    Esta clase se encarga de construir un objeto de tipo Maquina Expendedora.
    """
    def __init__(self) -> None:
        pass
```

Posteriormente encontraremos el método encargado de mostrar los productos dentro de la máquina, productos que serán leídos de un archivo ".csv". El método es el siguiente:

```
def mostrarProductos(self) -> str:
    """
    Creado el objeto de tipo maquina expendedora, este método se encarga \n
    de mostrar los productos contenidos en la máquina expendedora.
    """
    lista = []
    print("\n\t-Maquina expendedora krunkito-\n")
    with open('productos.csv', 'r') as f:
        i=1
        for linea in f.readlines():
            linea2 = linea.split(",")
            if int(linea2[2]) > 0:
                print(f"{linea2[0]}.{linea2[1]}: ${linea2[3]}", end="   ")
            else:
                print(f'{linea2[0]}.sin existencias', end="   ")
            if i%3 == 0:
                print("\n")
            clase = linea2[6].rstrip('\n')

            lista.append(globals()[clase](linea2[0], linea2[1], linea2[2], linea2[3],linea2[4],linea2[5]))
            i+=1

    print("\n")
    return lista
```

En esta clase igualmente se encuentra un método para actualizar el archivo ".csv" de acuerdo a los productos que se vayan seleccionando.

Producto.

Pasamos con la clase padre de la cual heredarán clases hijas, la clase Producto. En ella encontraremos atributos como la clave con la se seleccionará, el nombre del producto, el costo y la cantidad.

```
class Producto():
    """
    Esta es la clase padre para los productos, de la que heredarán atributo
    """
    def __init__(self,clave,  nombre, cantidad, costo):
        self.clave = clave
        self.nombre = nombre
        self.costo = costo
        self.cantidad = cantidad
```

El primer método en esta clase se encarga de mostrar la información general del producto.

```
    def  infoGeneral(self) -> str:
        """
        El método muestra los atributos del objeto.
        """
        print(f"clave: {self.clave}\n\
nombre: {self.nombre}\n\
costo: {self.costo}\n\
cantidad en existencia:{self.cantidad}")
```

La clase cuenta con un método para escribir sobre el archivo que contiene los productos y de esta manera añadir el tipo de producto que se requiera.

```
def añadirProductoGen(self) -> None:
    """
    Este método añade prodcutos del tipo que se refiera. Por ejemplo:\n
    Botanas, Bebidas, etc.
    """
    with open('productos.csv', 'a+') as f:
        f.write(f"\n{self.clave},{self.nombre},{self.cantidad},{self.costo}")
```

A continuación tenemos un getter, que retorna el precio del producto.

```
def retornarPrecio(self):
    """
    Este método retorna el precio del producto.
    """
    return int(self.costo)
```

Por último tenemos un método que restará una unidad a la cantidad del producto en el caso que se presente una compra en la máquina expendedora.

```
def comprar(self):
    """
    Este método resta un producto a la cantidad que yace en la máquina, una vez sea comprado.
    """
    self.cantidad = int(self.cantidad)-1
```

Estos métodos posteriormente son heredados por las clases hijas, sean Botanas, Bebidas y Dulces.

Administrador.

Tras mirar las clases principales de la máquina expendedora, pasamos a un segundo plano en las funciones de esta. Esto refiere al caso en que algún administrador desee hacer uso de la máquina expendedora para realizar cambios en esta. Esto se hizo con el proposito de simular un tipo de "mantenimiento" a la máquina expendedora, sea para el reabastecimiento de la misma.

Para lograr lo anterior, se creó la clase Administrador, que cuenta con los atributos "usuario" y "contraseña". De esta manera iniciará sesión el administrador:

```
class Administrador():
    """
    Esta clase crea un objeto de tipo Administrador. Sus atributos son:
    Nombre y Contraseña.
    """
    def __init__(self, nombre: str, contraseña: str) -> None:
        self.__nombre = nombre
        self.__contraseña = contraseña
```

El o los administradores se guardarán en un archivo ".csv", el cual será leído para la verificación del nombre y contraseña del usuario que introduce. Esto se ve en el siguiente método.

```
def validacion(self) -> bool:
    """
    Este método valida si el usuario y contraseña introducidos coiniciden con \n
    los presentes en el archivo "administradores.csv".
    """
    with open("administradores.csv", "r") as f:
        for linea in f.readlines():
            linea  = linea.split(sep=',')
            if self.__nombre == linea[0] and (self.__contraseña+'\n' == linea[1] or self.__contraseña == linea[1]):
                return 1
        return 0
```

Por último, se creó un método para los casos en que se requiera dar de alta a un administrador nuevo. Los datos se escribirán en el archivo con los usuarios contenidos.

```python
def nuevoAdmin(self) -> None:
    """
    Este método registra a un administrador nuevo en el archivo "administradores.csv"\n
    si se es necesario.
    """
    with open("administradores.csv", "a+") as f:
        f.write(f"\n{self.__nombre},{self.__contraseña}")
```

Para las funciones que desempeñará un objeto de tipo administrador se creó un archivo ".py" que contenrá a estas. A continuación se hará una breve explicación de estas:

La siguiente función se encarga de que las cantidades de productos introducidas sean las correctas, siendo estas mayores que cero.

```python
def validarCantidades (cantidades: int) -> bool:
    """
    Esta función valida que se introduzcan cantidades de productos validas, o sea,\n
    mayores a cero.
    """
    if cantidades<=0:
        print("Ingrese cantidad valida")
        return False
    else:
        return True
```

Esta función actualiza el archivo que contiene a los productos de la máquina expendedora, según la acción que se realice.

```python
def actualizarDatos(lista: list) -> None:
    """
    Esta función actualiza los datos que se contengan en el archvio ".csv".
    """
    act = []
    for i in range(len(lista)):
        act.append(lista[i].retornarInfo())
    maquina.actualizarCSV(act)
```

La función añade un producto a la máquina expendedora, a través del ingreso de los datos del mismo. Añadiendo el producto al archivo que contiene a los productos.

```
def añadirProducto(lista: list) -> None:
    """
    Esta función lee los datos nuevos ingresados del producto y posteriormente\n
    son añadidos al archivo ".csv". En esta función también se hace manejo de exepciones.
    """
    global product
    try:
        tipo = int(input("¿Qué clase de producto va a ingresar?\t1.Botana\t2.Bebida\t3.Dulce: \n"))
    except ValueError:
            print("\nIntroduzca solo numeros!!!\n")
    else:
        if int(tipo) == 1 or int(tipo) == 2 or int(tipo) == 3:

            marca = input("Ingrese el nombre del producto: ")
            try:
                verifCadena(marca)
            except cadenaVacia:
                print('No ingrese cadenas vacías')
                return 1
```

Esta función modifica la información de algún producto introducido, reescribiendo el archivo que contiene a los productos.

```
def modificarProducto(lista:list) -> None:
    """
    Esta función modifica un atributo (sea sabor, costo, gramos, etc) de un producto \n
    y posteriormente actualiza la información del archivo ".csv".
    """
    try:
        clave = int(input('Ingrese la clave del producto que desea modificar: '))
    except ValueError:
            print("\nIntroduzca solo numeros!!!")
    else:
        clave = clave-1
        modif = int(input('ingrese el dato que desee cambiar:\n\
    1. Nombre\n2. Costo\n3. Cantidad\n4. Gramos\n5. Sabor\n'))
```

La siguiente función elimina el producto que se desee de la máquina expendedora.

```
def eliminarProducto(lista: list) -> None:
    """
    La función elimina un producto de los presentes en el archivo ".csv".
    """
    try:
        clave = int(input('Ingrese la clave del producto que desea eliminar'))
    except ValueError:
            print("\nIntroduzca solo numeros!!!")
    else:
```

Función encargada de crear un nuevo administrador en el caso que se desee dar de alta.

```
def crearAdmin()-> None:
    """
    Esta función permite generar un nuevo administrador. Añadiendolo al arc
    """
    nombre = input("Ingrese el nombre del nuevo administrador: ")
    contraseña = input("Ingrese su contraseña: ")
    if len(nombre) != 0 or len(contraseña) != 0:
        nuevoAdmin = Administrador(nombre, contraseña)
        nuevoAdmin.nuevoAdmin()
    else:
        'No se puede ingresar cadenas vacias'
```

Por útlimo la función principal de estas, la del inicio de sesión. En esta se verifica si el nombre de usuario, así como contraseña, se encuentran en el archivo que contiene a los administradores; de ser así se le dará acceso a las funciones anteriormente mencionadas; de lo contrario no se le permitirá el acceso a dichas funciones.

```
def iniciarSesion(lista: list) -> None:
    """
    Esta función valida si el usuario y contraseña introducidos existen en el archivo\n
    donde se encuentran los administradores.
    """
    usuario = input("Ingrese su usuario: ")
    contraseña = input("Ingrese su contraseña: ")
    admin = Administrador(usuario, contraseña)
    validar = admin.validacion()
    print("\n")
    if validar == 1:
```

Excepciones.

Antes de pasar a la función principal, se mostrará el manejo de excepciones que se formularon para este programa.

Primeramente se hizo una excepción propia para los casos en que el usuario introduzca una clave desconocida, refiriendonos a las claves con las que se escogen, ingresan o eliminan los productos de la máquina expendedora.

```
class ClaveDesconocida(Exception):
    """
    Esta clase se encarga de manejar las expeciones que puedan presentarse\n
    en el transcurso de la ejecución del programa.
    """
    def __init__(self, mensaje="Ingrese una clave existente") -> None:
        """
        Método constructor de la excepción. Se hereda de la clase Exception.
        """
        self.mensaje = mensaje

        super().__init__(self.mensaje)
```

Ella hereda de la clase **Exception**, y el método para poder utilizarla es el siguiente:

```
def verifClave(lista: list, clave: int) -> str:
    """
    Maneja la excepción en que el usuario no introduzca la clave \n
    del producto correctamente.
    """
    if clave - 1 < 0 or clave > len(lista)-1:
        raise(ClaveDesconocida())
```

Se creó otra excepción para el manejo de casos en que el usuario introduce una cadena vacía como opción en cualquiera de las opciones que introduzca; sea para escoger producto, sea un administrador añadiendo un producto, etc.

```
class cadenaVacia(Exception):
    """
    Clase para la excepción en que se introduzca una cadena vacía.
    """
    def __init__(self, mensaje="No ingrese cadenas vacias") -> None:
        self.mensaje = mensaje

        super().__init__(self.mensaje)
```

El método para utilizarla es el siguiente:

```
def verifCadena(cadena: str) -> str:
    """
    Excepción que maneja los casos en que el usuario introduce cadenas vacías.
    """
    if len(cadena) ==  0:
        raise(cadenaVacia())
```

De esta forma se manejaron las excepciones del programa, además de esto se utilizaron excepciones nativas de Python: ValueError y IndexError.

Función principal.

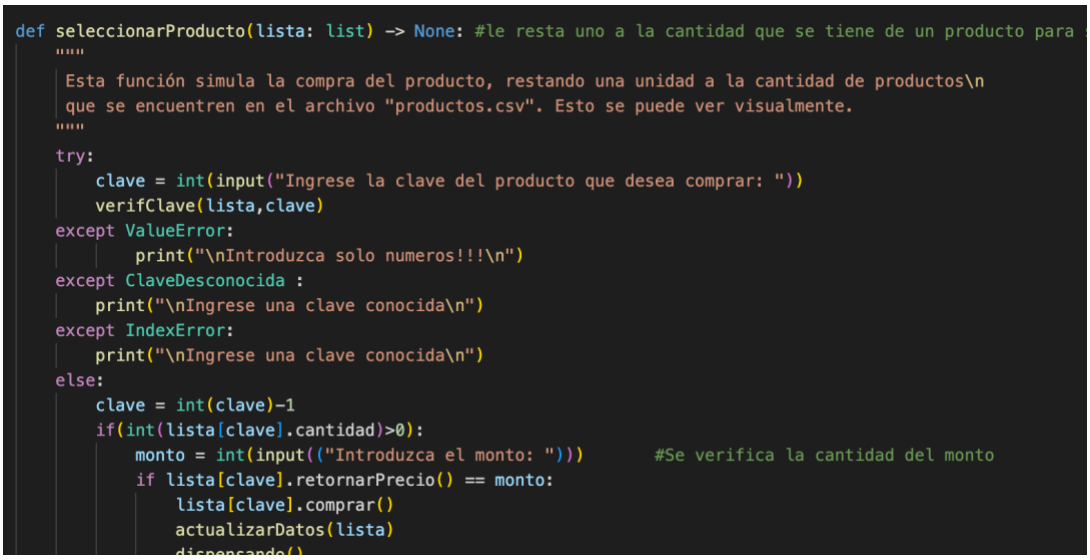
A continuación se mostrará la función principal, así como las funciones que se crearon y utilizaron en esta. La función principal es la siguiente:

```
def main():
    """
    Esta es la función principal.
        """
    while 1:
        lista = maquina.mostrarProductos()
        try:
            opcion = int(input("¿Qué es lo que desea hacer?\n\
  1. Seleccionar producto\n\
  2. Mirar productos por tipo\n\
  3. Ver informacion de un producto\n\
  4. Modo Administrador\n\
  5. Salir\n"))
        except ValueError:          #verifica que solo ingresen números
            print("\nIntroduzca solo numeros!!!\n")
        else:
            if opcion == 1:
                seleccionarProducto(lista)
            elif opcion == 2:
                productosTipo(lista)
            elif opcion == 3:
                infoProducto(lista)
            elif opcion == 4:
                iniciarSesion(lista)
            elif opcion == 5:
                print("Gracias por haber utilizado la maquina expendedora 'Kunkito'")
                break

main()
```

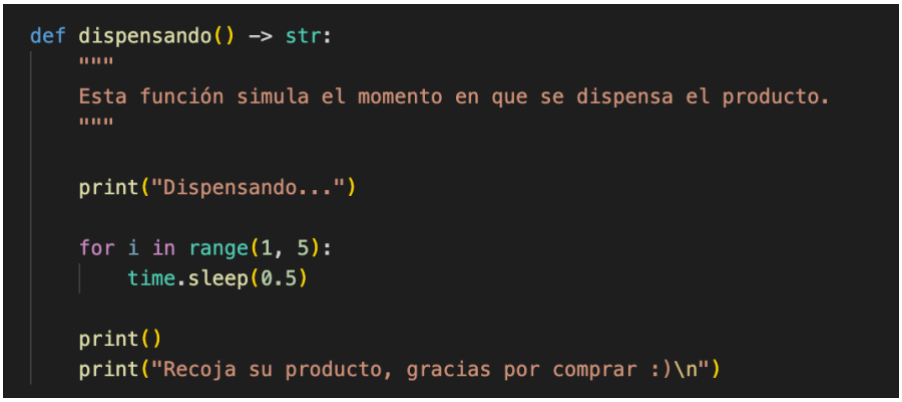
En ella sedesplegará un menú mostrando las opciones que el usuario puede realizar, posteriormente se le pedirá que ingrese la opción deseasa y de acuerdo a esta opción se realizará la acción. En las opciones podemos ver el llamamiento de funciones, funciones que se describirán a continuación:

En seleccionar producto se utilizan métodos, excepciones y funciones anteriormente vistas, con el fin de hacer que una vez el usuario escoja el producto que desee, el archivo de productos se actualice con la disminución de una unidad de este producto.



```
def seleccionarProducto(lista: list) -> None: #le resta uno a la cantidad que se tiene de un producto para
    """
    Esta función simula la compra del producto, restando una unidad a la cantidad de productos\n
    que se encuentren en el archivo "productos.csv". Esto se puede ver visualmente.
    """
    try:
        clave = int(input("Ingrese la clave del producto que desea comprar: "))
        verifClave(lista,clave)
    except ValueError:
            print("\nIntroduzca solo numeros!!!\n")
    except ClaveDesconocida :
        print("\nIngrese una clave conocida\n")
    except IndexError:
        print("\nIngrese una clave conocida\n")
    else:
        clave = int(clave)-1
        if(int(lista[clave].cantidad)>0):
            monto = int(input(("Introduzca el monto: ")))       #Se verifica la cantidad del monto
            if lista[clave].retornarPrecio() == monto:
                lista[clave].comprar()
                actualizarDatos(lista)
                dispensando()
```

En la misma función se hace llamado de una función, llamada “dispensando” que se encargará de simular el tiempo en que se dispensa un producto.



```
def dispensando() -> str:
    """
    Esta función simula el momento en que se dispensa el producto.
    """

    print("Dispensando...")

    for i in range(1, 5):
        time.sleep(0.5)

    print()
    print("Recoja su producto, gracias por comprar :)\n")
```

Contenido en esta misma función, se hace el requerimiento del dinero para pagar el producto, donde se separo en casos donde el usuario introduce el dinero exacto, donde le sobra (se regresa el cambio) o le hace falta dinero.

```
if(int(lista[clave].cantidad)>0):
    monto = int(input(("Introduzca el monto: ")))       #Se verifica la cantidad
    if lista[clave].retornarPrecio() == monto:
        lista[clave].comprar()
        actualizarDatos(lista)
        dispensando()
    elif monto > lista[clave].retornarPrecio(): #Si excede, regresa el cambio
        print("Su cambio: " + str(monto - lista[clave].retornarPrecio()))
        lista[clave].comprar()
        actualizarDatos(lista)
        dispensando()
    elif monto < lista[clave].retornarPrecio():
        print("Monto insuficiente :(")
        print()
else:
    print('Producto sin existencia')
```

Siguiendo con las opciones del menú, nos encontramos con la función que se encargará de mostrar los productos de un solo tipo. Esta función mostrará solo el tipo de producto que el usaurio desee ver.

```
def productosTipo(lista: list) -> str:
    """
      Esta función muestra los productos de un solo tipo, sean Bebidas, Botanas o Dulces.
    """
    try:
        tipo = int(input('Ingrese el tipo de productos que desee ver: \n1. Botanas\n2. Bebida\n3. Dulces\n
        if tipo == 1:
            for i in range(len(lista)):
                if lista[i]._tipo == 'Botana':
                    print(lista[i].infoGeneral())
        elif tipo == 2:
            for i in range(len(lista)):
                if lista[i]._tipo == 'Bebida':
                    print(lista[i].infoGeneral())
        elif tipo == 3:
            for i in range(len(lista)):
                if lista[i]._tipo == 'Dulces':
                    print(lista[i].infoGeneral())
    except ValueError:
        print("Ingrese solo valores válidos")
```

La siguiente función se encarga de mostrar los datos específicos de algún producto, esto por medio de la introducción de su clave.

```python
def infoProducto(lista: list) -> str: #muestra la información de un solo producto
    """
      Esta función muestra la información del producto que se requiera saber.
    """
    try:
        clave = int(input("Ingrese la clave del producto del que desa saber mas información: "))
        verifClave(lista,clave)
    except ValueError:
            print("\nIntroduzca solo numeros!!!\n")
    except ClaveDesconocida :
        print("\nIngrese una clave conocida\n")
    except IndexError:
        print("\nIngrese una clave conocida\n")
    else:
        clave = int(clave)-1
        info = lista[clave].infoEspecifica()
        print(info)
```

En la cuarta opción encontramos la función que se encargará de permitir al usuario administrador iniciar sesión, esta función se mostró anteriormente. La última opción se encarga de finaliazar el ciclo y de este modo terminando el programa y uso de la máquina expendedora.

De esta manera se concluye con la explicación de las clases, métodos, excepciones y funciones del programa hecho para simular la máquina expendedora.

Manual de usuario.

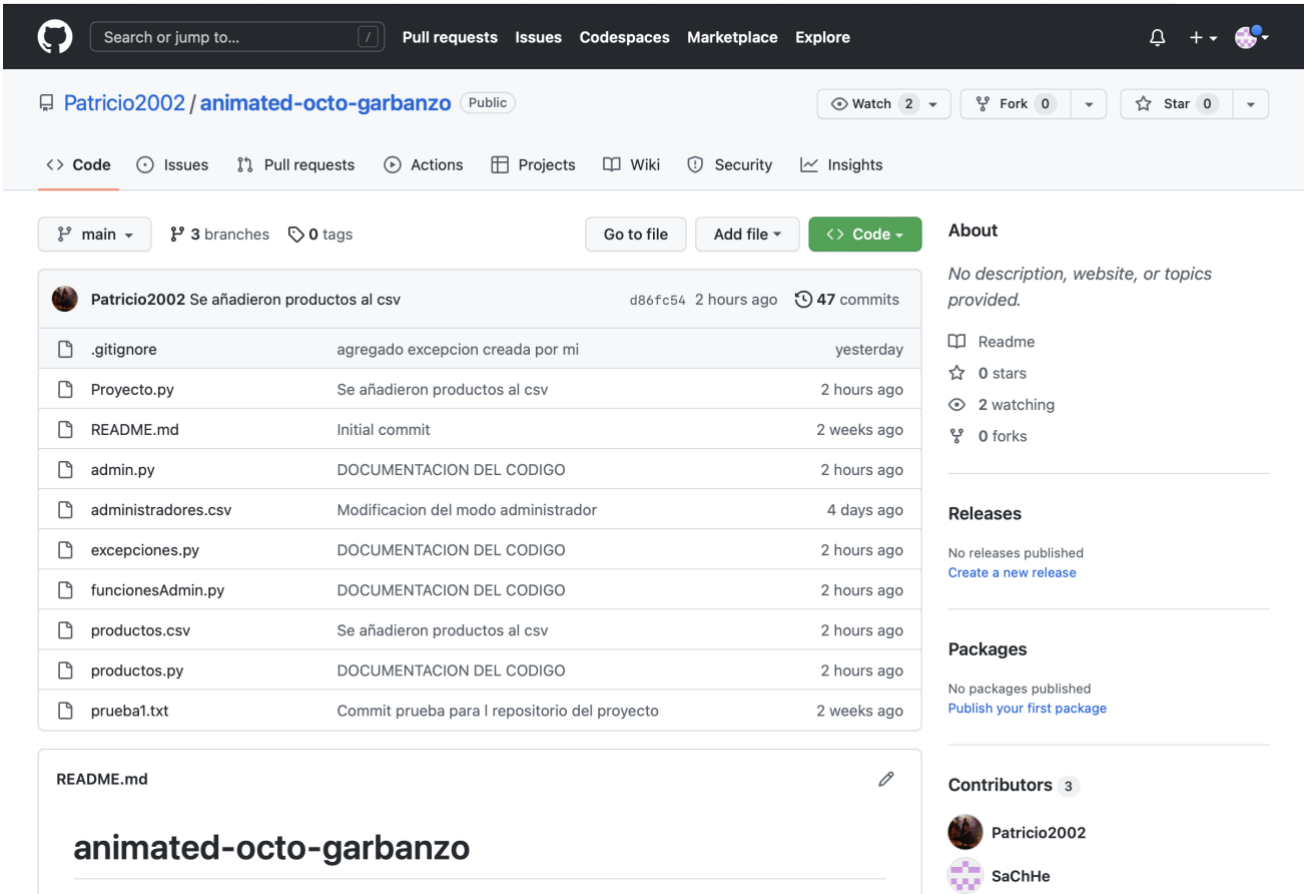
Antes de pasar a la demostración y guía a través de la interfaz de usaurio en consola, se mencionaran los requerimientos para el uso de este programa. Primeramente se requiere de un IDE, de preferencia Visual Studio Code; una vez con el IDE se requiere de la isntalación del interprete de Python, el cual es fácil de instalar por medio del siguiente enlace: https://www.python.org/downloads/

Para saber si el interprete de Python se ha descargado de manera efectiva, en la terminal de nuestra computadora podemos introducir el siguiente comando:

```
SamChong@MacBook-Air-de-Sam-2 Documents % Python3 --version
Python 3.10.4
SamChong@MacBook-Air-de-Sam-2 Documents %
```

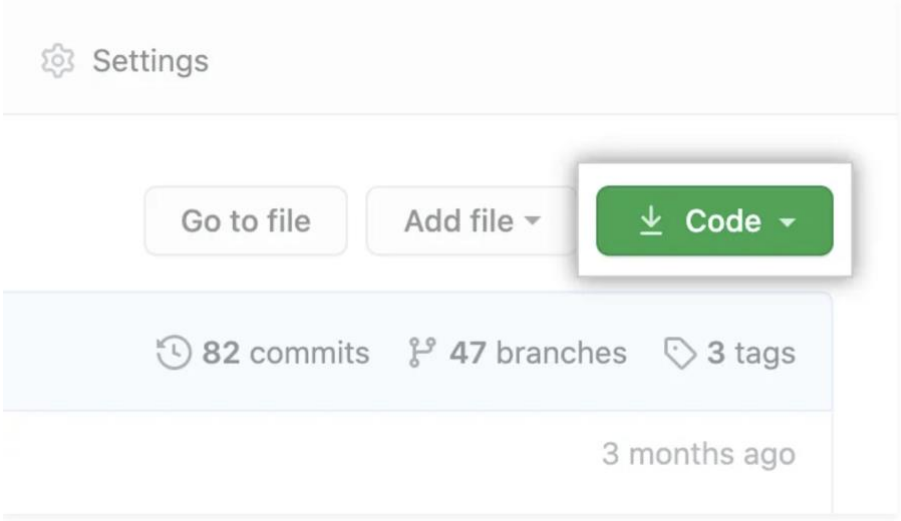
Una vez instalado el interprete de Python y con un IDE podemos obtener el repositorio de este proyecto en GitHub a través del siguiente enlace: https://github.com/Patricio2002/animated-octo-garbanzo

Tras ingresar al enlace se mostrará la siguiente página:

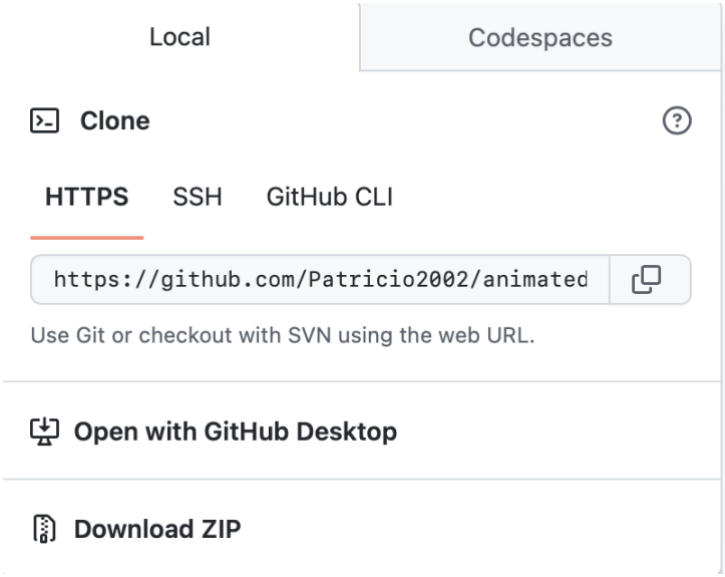


Para poder hacer uso del programa, se tiene que clonar el repositorio de manera local en nuestro dispositivo, para esto debemos seguir los siguientes pasos:

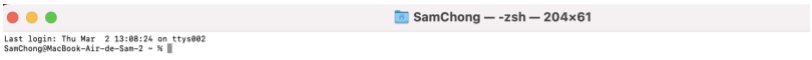
1. Seleccionar la pestaña <> Code.



2. Copiar el enlace en la pestaña HTTPS.



3. Abrir la terminal de nuestra computadora.

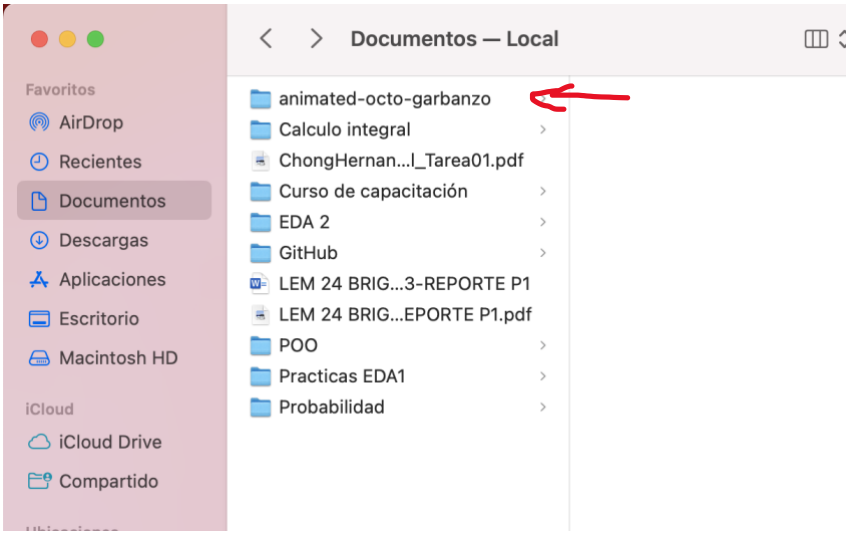


4. Dirigirnos al directorio de trabajo donde se desee clonar el repositorio e ingresar el siguiente comando:

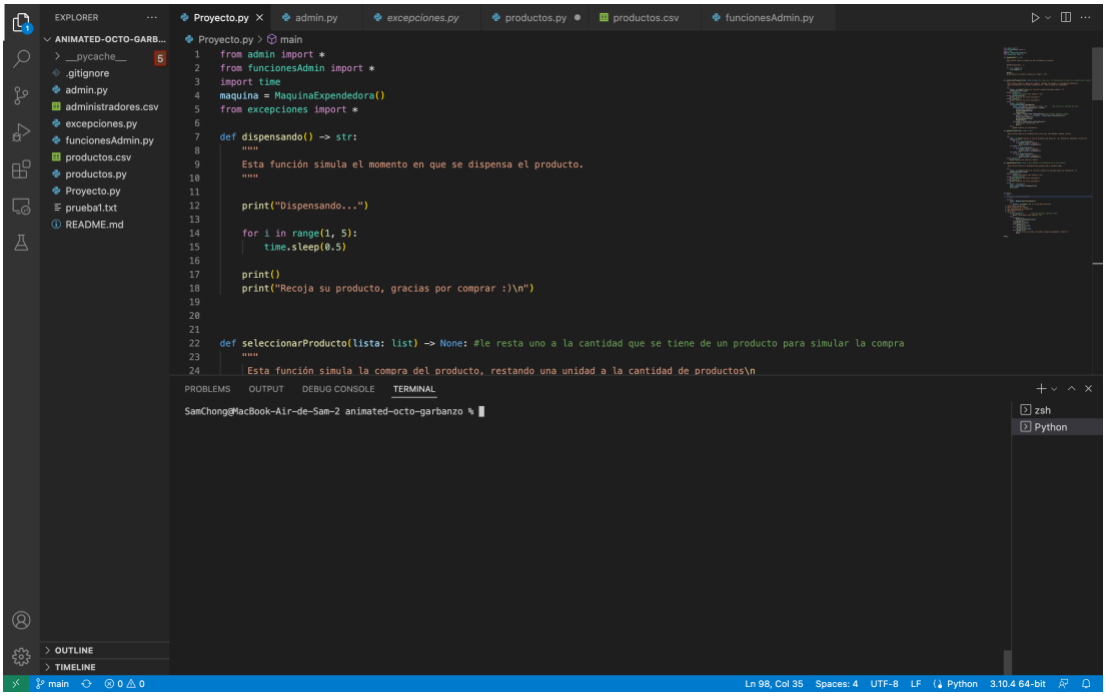
```
$ git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY
```

```
Last login: Thu Mar  2 13:08:24 on ttys002
SamChong@MacBook-Air-de-Sam-2 ~ % cd Documents
SamChong@MacBook-Air-de-Sam-2 Documents % git clone https://github.com/Patricio2002/animated-octo-garbanzo.git
Clonando en 'animated-octo-garbanzo'...
remote: Enumerating objects: 217, done.
remote: Counting objects: 100% (217/217), done.
remote: Compressing objects: 100% (133/133), done.
remote: Total 217 (delta 131), reused 156 (delta 77), pack-reused 0
Recibiendo objetos: 100% (217/217), 39.49 KiB | 201.00 KiB/s, listo.
Resolviendo deltas: 100% (131/131), listo.
SamChong@MacBook-Air-de-Sam-2 Documents %
```

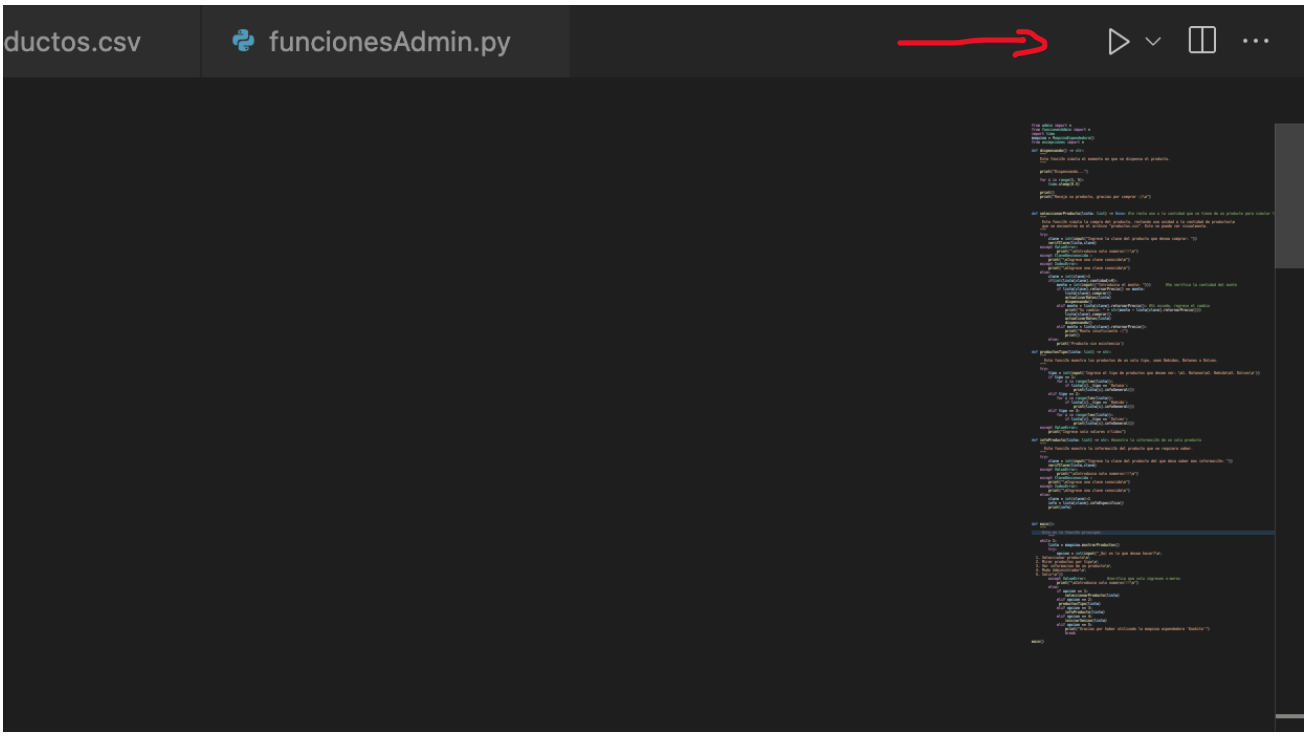
5. Hecho esto el repositorio se encontrará en el directorio actual.



Una vez hechos estos pasos, abrimos nuestro IDE, en este caso **Visual Studio Code** y abrimos el repositorio. En esta parte se mostrará todo el código fuente:



El siguiente paso simplemente es correr el programa, lo cual puede hacerse de dos formas; la primera es simplemente oprimiendo el botón de “play” de Visual Studio Code que se encuentra en la parte superior derecha:



La otra forma es corriendo por medio de la terminal de la siguiente manera:

```
SamChong@MacBook-Air-de-Sam-2 animated-octo-garbanzo % Python3 Proyecto.py

        -Maquina expendedora krunkito-

1.Sabritones: $15   2.Coca Cola: $20   3.Takis: $40

4.boing: $28   5.Chetos: $1   6.skittles: $23
```

Ejecución.

Una vez en ejecución el programa se nos mostrarán los productos contenidos en la máquina, así como el menú de opciones. Aquí el usuario únicamente debe introducir la opción deseada.

```
        -Maquina expendedora krunkito-

1.Sabritones: $15   2.Coca Cola: $20   3.Takis: $40

4.boing: $28   5.Chetos: $1   6.skittles: $23

7.Pepsi: $17   8.Pelon: $14   9.gomitas de panda: $10

10.kit kat: $20   11.Fanta: $16   12.Manzanita Sol: $14

13.Takis Fuego: $15

¿Qué es lo que desea hacer?
  1. Seleccionar producto
  2. Mirar productos por tipo
  3. Ver informacion de un producto
  4. Modo Administrador
  5. Salir
```

Si desea seleccionar un producto, el programa le indicará que ingrese la clave del producto la cual se encuentra previamente al nombre del producto. Por ejemplo: clave de sabritones = 1, clave de Coca Cola = 2, etc.

Si el usaurio introduce correctamente la clave, se le pedirá que ingrese el monto a pagar. Posterior a esto le será dispensado su producto.

```
¿Qué es lo que desea hacer?
  1. Seleccionar producto
  2. Mirar productos por tipo
  3. Ver informacion de un producto
  4. Modo Administrador
  5. Salir
1
Ingrese la clave del producto que desea comprar: 1
Introduzca el monto: 15
Dispensando...

Recoja su producto, gracias por comprar :)
```

En este caso el usuario ingreso el monto correcto de acuerdo al costo del producto, pero la máquina puede devolver el cambio en caso quese exceda del precio del producto o cancelar la compra en el caso que el usaurio no ingrese correctamente el monto:

```
¿Qué es lo que desea hacer?
  1. Seleccionar producto
  2. Mirar productos por tipo
  3. Ver informacion de un producto
  4. Modo Administrador
  5. Salir
1
Ingrese la clave del producto que desea comprar: 1
Introduzca el monto: 30
Su cambio: 15
Dispensando...

Recoja su producto, gracias por comprar :)
```

Tras elegir esta opción, podemos ver la modificación realizada en el archivo “productos.csv” mirando que en efecto se ha restado un producto a la cantidad que se tenía en un inicio.

La segunda opción corresponde al caso en que el usuario desee ver los productos de un solo tipo, sea solo Bebidas, solo Botanas o solo Dulces. Para esto el programa mostrará las opciones y el usuario solo debe ingrear la opción deseada. Posterior a esto, se mostrarán únicamente las opciones con las que dispone la máquina del mismo tipo.

```
Ingrese el tipo de productos que desee ver:
1. Botanas
2. Bebida
3. Dulces
2
clave: 2
nombre: Coca Cola
costo: 20
cantidad en existencia:12
None
clave: 4
nombre: boing
costo: 28
cantidad en existencia:8
None
clave: 7
nombre: Pepsi
costo: 17
cantidad en existencia:14
None
clave: 11
nombre: Fanta
costo: 16
```

La última opción que corresponde a un usuario común corresponde a aquella encargada de mostrar la información de un producto en específico contenido en la máquina expendedora. Para esto el usuario únicamente debe introducir la clave del producto que desea conocer su información específica.

```
3
Ingrese la clave del producto del que desa saber mas información: 6
clave: 6
producto: Dulces
nombre: skittles
cantidad en existencia:15
costo:23
sabor(es): fresa
gramos: 46
```

Pasando a la opción para los usuarios administradores, se debe introducir la opción 4. Para esto se les requerirá que introduzcan su usuario y contraseña, los cuales deben estar almacenados en el archivo "administradores.csv". Si son correctos ambos, se le dará acceso a las demás funciones.

```
4
Ingrese su usuario: SamChong
Ingrese su contraseña: samchong


Bienvenido SamChong.
¿Qué desea hacer?

1. Añadir producto
2. Eliminar producto
3. Modificar información de un producto
4. Crear nuevo administador
5. Cerrar Sesión (Volver al menú inicial)
```

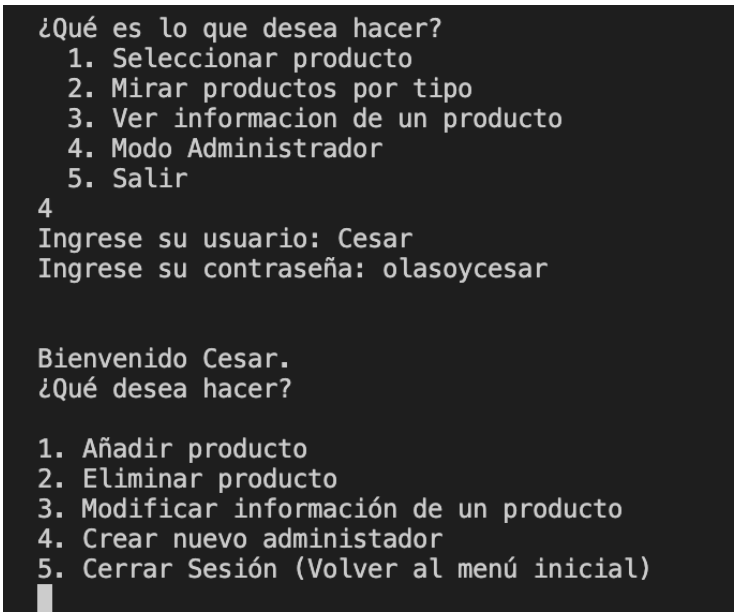
Con esto, el administrador puede realizar el mantenimiento de la máquina, sea para reabastecerla, eliminar algún producto, modificar su información o añadir un nuevo administrador. Tras terminar, puede cerrar sesión y seguir con el uso de la máquina expendedora. Añadiendo un nuevo administrador:

```
Bienvenido SamChong.
¿Qué desea hacer?

1. Añadir producto
2. Eliminar producto
3. Modificar información de un producto
4. Crear nuevo administador
5. Cerrar Sesión (Volver al menú inicial)
4


Ingrese el nombre del nuevo administrador: Cesar
Ingrese su contraseña: olasoycesar
```

Verificando:



De esta manera se puede hacer uso de la máquina expendedora. Cabe aclarar que a lo largo del programa se pueden presentar excepciones que terminen con la secuencia del programa o que arrojen ciertos mensajes, esto dependiendo al dato ingresado por el usuario.