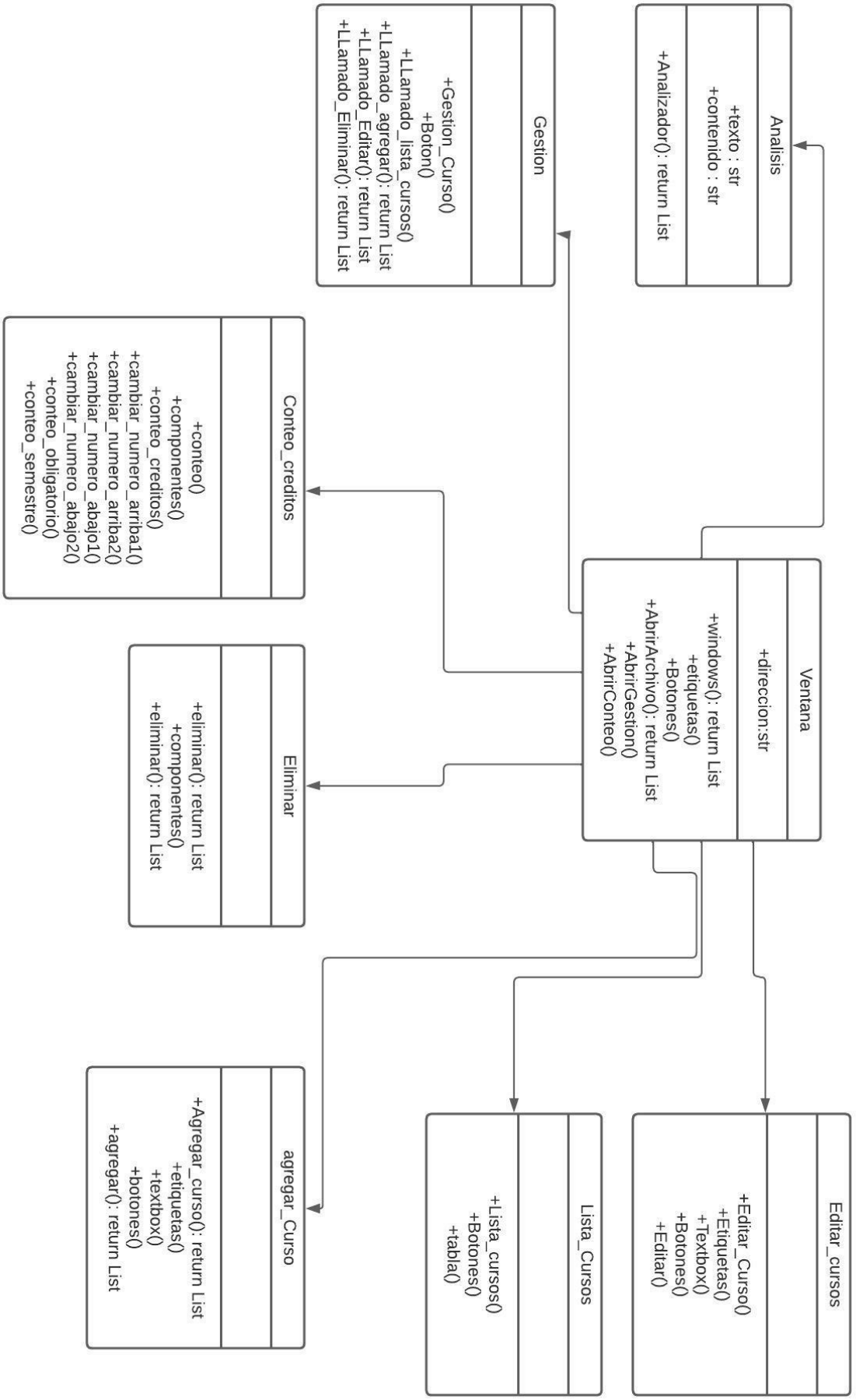
UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
INGENIERÍA EN CIENCIAS Y SISTEMAS
LENGUAJES FORMALES Y DE PROGRAMACIÓN 2
SECCIÓN A+

NOMBRE: LUDWING ALEXANDER LÓPEZ ORTIZ
CARNÉ: 201907608
CUI :3005455760101

ESTRUCTURA DE LA APLICACIÓN

DIAGRAMA DE CLASES:



PRINCIPIO TÉCNICO

PROGRAMACIÓN ORIENTADA A OBJETOS (POO)

La programación orientada a objetos (POO) es un paradigma de programación que usa objetos en sus interacciones para diseñar aplicaciones y programas informáticos.

Está basada en varias técnicas incluyendo herencia, cohesión, abstracción, polimorfismo, acoplamiento y encapsulamiento.

Su uso se popularizó a principios de la década de 1990. En la actualidad, existe una gran variedad de lenguajes de programación que soportan la orientación a objetos.

A lo largo de la historia, han ido apareciendo diferentes paradigmas de programación. Lenguajes secuenciales como COBOL o procedimentales como Basic o C, se centraban más en la lógica que en los datos. Otros más modernos como Java, C# y Python, utilizan paradigmas para definir los programas, siendo la Programación Orientada a Objetos la más popular.

Con el paradigma de Programación Orientado a Objetos lo que buscamos es dejar de centrarnos en la lógica pura de los programas, para empezar a pensar en objetos, lo que constituye la base de este paradigma. Esto nos ayuda muchísimo en sistemas grandes, ya que en vez de pensar en funciones, pensamos en las relaciones o interacciones de los diferentes componentes del sistema.

Un programador diseña un programa de software organizando piezas de información y comportamientos relacionados en una plantilla llamada clase. Luego, se crean objetos individuales a partir de la plantilla de clase. Todo el programa de software se ejecuta haciendo que varios objetos interactúen entre sí para crear un programa más grande.

COMPONENTES PRINCIPALES APLICACIÓN

ANÁLISIS DE UN DOCUMENTO PARA EXTRAER SUS COMPONTENTES

```python
class Analisis:

    texto = None
    Contenido = []

    def __init__(self, ubicacion):

        print('COMENZANDO ANALISIS')

        try:

            self.ubicacion = ubicacion  # OBTENCION DE UBICACION DE DOCUMENTO

            contenido = open(self.ubicacion, encoding='utf-8')  # ABRIR ARCHIVO

            self.texto = contenido.read()  # OBTENER EL CONTENIDO DEL ARCHIVO

            self.Analizador() #LLAMADO DEL METODO PARA ANALIZAR EL CONTENIDO

        except :

            print('NO SE REGISTRO UN ARCHIVO PARA PODER ANALIZAR')

    def Analizador(self):

        txt = self.texto + '\n'
        linea = ''
        #SEPARACION DEL TEXTO Y MANIPULACION EN LISTA
        for i in txt:

            if i != '\n':

                linea += i

            else:

                lista = linea.split(',')
                self.Contenido.append(lista)
                linea = ''

        ayuda = Herramientas()

        self.Contenido = ayuda.Repetido(self.Contenido)
```

GRAFICACIÓN DE UNA VENTANA

```
class Ventana:

    direccion =''


    def __init__(self):

        print('INICIANDO PROGRAMA...')


    #METODO DE INICIACION VENTANA PRINCIPAL
    def Window(self):

        self.ventana = Tk()
        self.ventana.geometry('500x500+550+150')
        self.ventana.configure(background='LightCyan3')
        self.ventana.resizable(False,False)


        self.Etiquetas()
        self.Botones()


        self.ventana.mainloop()



    #DATOS DEL ESTUDIANTE MEDIANTE LABELS
    def Etiquetas(self):

        label1 = Label(text='LENGUAJES FORMALES Y DE
PROGRAMACION',background='LightCyan3').place(x=5,y=5)
        label2 = Label(text='LUDWING ALEXANDER LOPEZ ORTIZ',background='LightCyan3').place(x=5,y=20)
        label3 = Label(text='201907608',background='LightCyan3').place(x=5,y=35)


    #BOTONES DE LA PANTALLA PRINCIPAL
    def Botones(self):



         boton1 = Button(text='Abrir
Archivo',command=self.AbrirArchivo,height=2,width=15).place(x=200,y=150)
         boton2 = Button(text='Gestionar
Cursos',command=self.AbrirGestion,height=2,width=15).place(x=200,y=200)
         boton3 = Button(text='Conteo de
Creditos',command=self.AbrirConteo,height=2,width=15).place(x=200,y=250)
         boton4 =
Button(text='Salir',command=self.ventana.destroy,height=2,width=15).place(x=200,y=300)




    #METODO PARA LA SELECCION DE ARCHIVO
    def AbrirArchivo(self):

        archivo = filedialog.askopenfilename(title="Abrir",initialdir="C:/", filetypes=(('Archivos
CSV','*.csv'),('Archivos LFP','*.lfp')))
        self.direccion = archivo
        self.Analisis = Analisis(self.direccion)
        self.Contenido = Analisis.Contenido
       # print(self.Contenido)

    #metodod para llamar ventana de gestion de cursos
    def AbrirGestion(self):

       gestiones = Gestiones()

       self.Contenido = gestiones.GestionarCurso(self.Contenido)

    def AbrirConteo(self):

        conteo = Conteo_Creditos()
        conteo.Conteo(self.Contenido)
```

MÉTODO CREACIÓN DE TABLA

```
#TABLA DINAMICA PARA LA LISTA DE LOS CURSOS
    def Tabla(self):

        #UTILIZACION DE TREEVIEW PARA CREAR LA TABLA
        tb = ttk.Treeview(self.ventana_lista_cursos, columns=
('col0','col1','col2','col3','col4','col5','col6'), show='headings')
        tb.place(x=10,y=20)

        #CREACION DE COLUMNAS CON PROPIEDADES
        #tb.column('#0',width=80)
        tb.column('col0',width=80,anchor=CENTER)
        tb.column('col1',width=80,anchor=CENTER)
        tb.column('col2',width=80,anchor=CENTER)
        tb.column('col3',width=80,anchor=CENTER)
        tb.column('col4',width=80,anchor=CENTER)
        tb.column('col5',width=80,anchor=CENTER)
        tb.column('col6',width=80,anchor=CENTER)

        #TITULO DE LAS COLUMNAS
        #tb.heading('#0',text='Código',anchor = CENTER)
        tb.heading('col0',text='Código',anchor = CENTER)
        tb.heading('col1',text='Nombre',anchor = CENTER)
        tb.heading('col2',text='Pre-Requisito',anchor = CENTER)
        tb.heading('col3',text='Opcionalidad',anchor = CENTER)
        tb.heading('col4',text='Semestre',anchor = CENTER)
        tb.heading('col5',text='Creditos',anchor = CENTER)
        tb.heading('col6',text='Estado',anchor = CENTER)

            #CICLO FOR PARA PODER CREAR CONTENIDO EN LA TABLA
        for i in self.contenido:

            tb.insert('',END, values=i)
```

METODO AGREGAR UN CURSO

```
def agregar(self):

        nuevo = []
        codigo = self.Caja1.get()
        nombre = self.Caja2.get()
        prerequisitos = self.Caja3.get()
        semestre = self.Caja4.get()
        opcionalidad = self.Caja5.get()
        creditos = self.Caja6.get()
        estado = self.Caja7.get()

        nuevo.append(codigo)
        nuevo.append(nombre)
        nuevo.append(prerequisitos)
        nuevo.append(opcionalidad)
        nuevo.append(semestre)
        nuevo.append(creditos)
        nuevo.append(estado)

        if codigo.isnumeric() and semestre.isnumeric() and creditos.isnumeric() and (opcionalidad =='1'
or opcionalidad=='0') and (estado == '1' or estado =='0' or estado =='-1'):

            self.contenido.append(nuevo)
            print(nuevo)
            messagebox.showinfo(message="CURSO AGREGADO")

        else:

            messagebox.askretrycancel(message='LOS DATOS QUE INGRESO NO SON CORRECTOS', title='Error')
```

MÉTODO PARA EDITAR

```
def editar(self):

        nuevo = []
        codigo = self.Caja1.get()
        nombre = self.Caja2.get()
        prerequisitos = self.Caja3.get()
        semestre = self.Caja4.get()
        opcionalidad = self.Caja5.get()
        creditos = self.Caja6.get()
        estado = self.Caja7.get()

        nuevo.append(codigo)
        nuevo.append(nombre)
        nuevo.append(prerequisitos)
        nuevo.append(opcionalidad)
        nuevo.append(semestre)
        nuevo.append(creditos)
        nuevo.append(estado)

        if codigo.isnumeric() and semestre.isnumeric() and creditos.isnumeric() and (opcionalidad =='1'
or opcionalidad=='0') and (estado == '1' or estado =='0' or estado =='-1'):

            repetido = BooleanVar()
            ubicacion = IntVar()
            r = 0

            for i in self.contenido:

                if i[0] == codigo:

                    repetido = True

                    ubicacion = r
                else:

                    r+=1

            if repetido == True:

                self.contenido[ubicacion][0] = codigo
                self.contenido[ubicacion][1] = nombre
                self.contenido[ubicacion][2] = prerequisitos
                self.contenido[ubicacion][3] = semestre
                self.contenido[ubicacion][4] = opcionalidad
                self.contenido[ubicacion][5] = creditos
                self.contenido[ubicacion][6] = estado
                messagebox.showinfo(message="CURSO EDITADO!")

            else:

                messagebox.showinfo(message="CURSO NO ENCONTRADO PARA EDITAR")



        else:

            messagebox.askretrycancel(message='LOS DATOS QUE INGRESO NO SON CORRECTOS', title='Error')
```

MÉTODO ELIMINAR

```
def Eliminacion(self):

        codigo = self.Caja1.get()

        Encontrado = BooleanVar()
        ubicacion = IntVar()

        cont = 0

        for i in self.contenido:

            if i[0] == codigo:

                Encontrado = True

                ubicacion = cont
            else:

                cont +=1

        if Encontrado == True:

            self.contenido.pop(ubicacion)
            messagebox.showinfo(message="CURSO ELIMINADO")

        else:

            messagebox.showinfo(message="CURSO NO ENCONTRADO PARA ELIMINAR")
```