Manual Técnico

Este es el menú que se muestra al usuario donde menú es un numero que hace la entrada en el if que se encuentra dentro de try catch para evitar posibles errores al presionar una tecla

```
print('\n1.Menu Entretenimiento\n2.Menu Educacion\n3.Salir')

try:
    menu = int(input('Seleccione una Opcion : '))

    if menu == 1:
        (leer_Archivo())
    elif menu == 2:
        """print(E2)"""
    elif menu == 3:
        sys.exit()
    else:
        print(" NO EXISTE TAL OPERACION  ")

except ValueError:
    print(" Lo que ingreso no es un numero")
```

Se muestra la función leer archivo. El cual en este se lee el archivo que el usuario esta mandando al software. también este lo separa en arreglos con el .split, esto para realizar mas fácil la entrada de datos para las funciones de cada mascota.

```
def leer_Archivo():
    juego = open("C:/Users/DELL/Desktop/archivo.mascotas")
    lineas = juego.read()
    linea = lineas.splitlines()
    for i in range(len(linea)):  # Recorrer las lineas
        splitl = linea[i].split(":")
```

El método que hace posible el archivo log donde se van mostrando todas las acciones de las mascotas

```
        result = open("archivo.mascotas_result", "w")
```

Ya habiendo inicializado con los códigos de las 2 imágenes anteriores, se procede a la lógica del programa de entretenimiento, el cual es la creación de los pájaros y gatos

```
    if (splitl[0] == "Crear Pajaro"):
        newpajaro=pajaro(splitl[1])
        lista.append(newpajaro)        # se guarda la mascota en el arreglo lista
        result.write(fecha()+' Se crea el pajaro : '+splitl[1]+"\n")
    if (splitl[0] == "Crear Gato"):
        newcat= Gatito(splitl[1])
        lista.append(newcat)  # se guarda la mascota en el arreglo lista
        result.write(fecha() + ' Se crea el gato : ' + splitl[1] + "\n")
```

Este fragmento de código nos muestra para darle comida a las mascotas los cuales pueden ser gatos o pájaros, este debe verificar que si las mascotas a quien se les va a dar alimento existen o ya fueron creadas, el cual al darle comida estos recargan de energía y no es la misma energía para ambas mascotas.

```
    if (splitl[0] == "Dar de Comer"):
        split2=splitl[1].split(",")
        obteniendo=newpajaro
        if obteniendo.nombre==split2[0]:
            if obteniendo.Estado=="vivo":
                escribir=split2[0]+ ' --Gracias. Ahora mi energia es ',pajaro.poder(pajaro,int(split2[1]))+1
                result.write(fecha()+ str(escribir)+"\n")
                obteniendo.Energia=obteniendo.Energia+pajaro.poder(pajaro,int(split2[1]))
            else:
                result.write(fecha()+ split2[0]+" Muy tarde. Ya he muerto")

        elif newcat.nombre==split2[0]:
            if newcat.Estado=="vivo":
                escribir = split2[0] + ' --Gracias. Ahora mi energia es ',Gatito.poder(Gatito, int(split2[1])) + 1
                result.write(fecha() + str(escribir) + "\n")
                newcat.Energia = newcat.Energia + Gatito.poder(Gatito, int(split2[1]))
            else:
                result.write(fecha()+ split2[0]+" Muy tarde. Ya he muerto")
```

Para crear el resumen de las mascotas estos deben de verificar si existe o no, entonces este debe obtener los datos directamente del objeto que se va modificando constantemente.

```
    if splitl[0]=="Resumen Mascota":
        for k in lista:
            if k.nombre==splitl[1]:
                if newpajaro.nombre==splitl[1]:
                    result.write("\n\n------------ Resumen de Mascota ----------------------\n")
                    result.write(fecha() + str(newpajaro)+"\n")
                    result.write("-------------------------------------------------------------------------------\n\n")

                else:
                    result.write("\n\n------------ Resumen de Mascota ----------------------\n")
                    result.write(fecha() + str(newcat)+"\n")
                    result.write("-------------------------------------------------------------------------------\n\n")
```

Para enviar el mensaje nosotros solo verificamos que el pájaro sea el único que tenga esa habilidad, a continuación el código.

```
if split1[0]=="Puede Entregar Mensaje":
    split2=split1[1].split(",")
    if obteniendo.nombre==split2[0]:

        if  obteniendo.Energia>pajaro.enerigia(pajaro,int(split2[1]),int(split2[2]),int(obteniendo.CoordenadaX),int(obteniendo.CoordenadaY)):
            result.write(fecha()+split2[0]+" Si puedo entregar el mensaje \n")
        elif  obteniendo.Energia<pajaro.enerigia(pajaro,int(split2[1]),int(split2[2]),int(obteniendo.CoordenadaX),int(obteniendo.CoordenadaY)):
            result.write(fecha()+split2[0] +" Estoy exhausto dame de comer : "+str( pajaro.food(pajaro,int(split2[1]),int(split2[2]),int(obteniendo.CoordenadaX),
                                                                                               int(obteniendo.CoordenadaY),obteniendo.Energia))+" gramos, Para ir \n")
        elif obteniendo.Energia == 0:
                obteniendo.Estado="Muerto"
                result.write(split2[0]+ " Ya me mori \n")
```

Asi como se verifica si se puede envíar el mensaje para el pájaro tambien se verifico para poder enviar al gato a comer un raton.

Para el resumen global de las mascotas solo se llama a imprimir a nuestro arreglo para ello llamamos al arreglo que tiene el método _str_ (el toString de python). El cual se recorre con un FOR.

```
if split1[0]=="Resumen_Global":
    result.write("\n ---------------------- Resumen Global --------------------------")

    for i in lista:
        result.write("\n"+str(i))
    result.write("\n ----------------------  --------------------------")
```

La clase mascota la cual es la principal en nuestro programa de entretenimiento.

```
class Mascotia:
    def __init__(self, name, x, y, peso, e):
        self.nombre = name
        self.CoordenadaX = x
        self.CoordenadaY = y
        self.Peso = peso
        self.Energia = e
        self.Estado='vivo'

    def __str__(self):
        sia = self.nombre +", Energia: "+str(self.Energia) +", X: " +str(self.CoordenadaX)+ ", Y: "+str(self.CoordenadaY)+", "+ self.Estado
        return sia
```

La clase pájaro esta siendo heredada de la clase mascota la cual esta es su padre, de donde se llama con su _Super_ . en esta clase se hacen solo los métodos matemáticos para calcular lo que necesita cada pájaro que se esta creando.

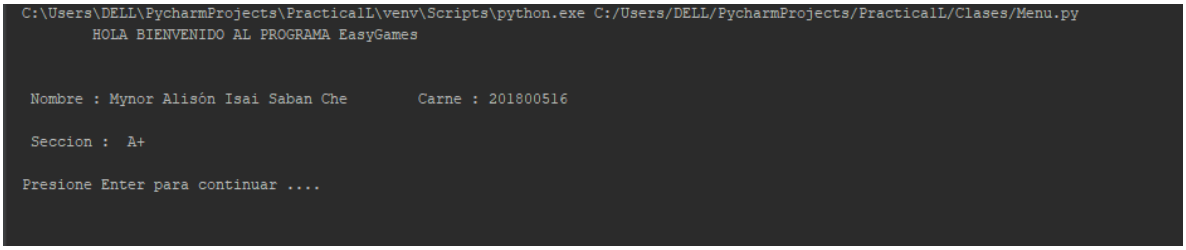
```
class pajaro(Mascotia):
    def __init__(self, name):
        Mascotia.__init__(self, name, 0, 0, 0, 1)

    def poder(self, dato):
        resultado = dato*4
        return resultado

    def mensaje(self,x, y, x1,x2):

        Distancia = math.sqrt(((x - x1) ** 2) + ((y - x2) ** 2))
        return round(Distancia)

    def enerigia(self,x,y,x1,x2):
        gasto=pajaro.mensaje(pajaro,x,y,x1,x2)+10
        return gasto

    def food(self,x,y,x1,y1,einicial):
        calculo = (pajaro.enerigia(pajaro,x,y,x1,y1)-einicial)/4
        return round(calculo)
```

La clase Gatito esta tiene herencia de la clase principal la cual es Mascotia, la cual se hace su llamado con su _Super_ , de la clase principal en este solo se hacen las funciones matemáticas para calcular su energía, comida, distancia para sus traslaciones.

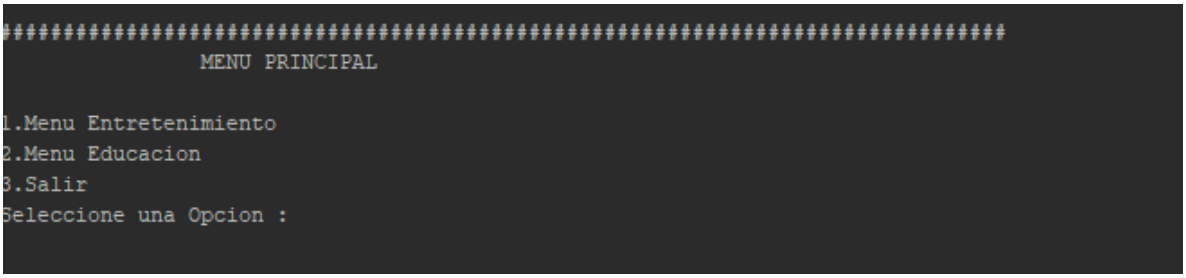
```python
class Gatito(Mascotia):
    def __init__(self, name):
        Mascotia.__init__(self, name, 0, 0, 0, 1)

    def poder(self, dato):
        resultado = dato+12
        return resultado

    def conviene(self,x, y, x1,x2):

        Distancia = math.sqrt(((x - x1) ** 2) + ((y - x2) ** 2))
        return round(Distancia)

    def energia(self,x,y,x1,x2):
        gasto=Gatito.conviene(Gatito,x,y,x1,x2)+10
        return gasto

    def food(self,x,y,x1,y1,einicial):
        calculo = (Gatito.energia(Gatito,x,y,x1,y1)-einicial)/4
        return round(calculo)
```

Manual de usuario

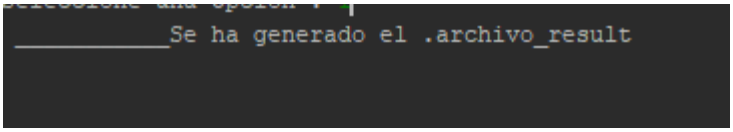
Al momento de ingresar al programa nos muestra los datos del estudiante el cual se muestra en la siguiente imagen:



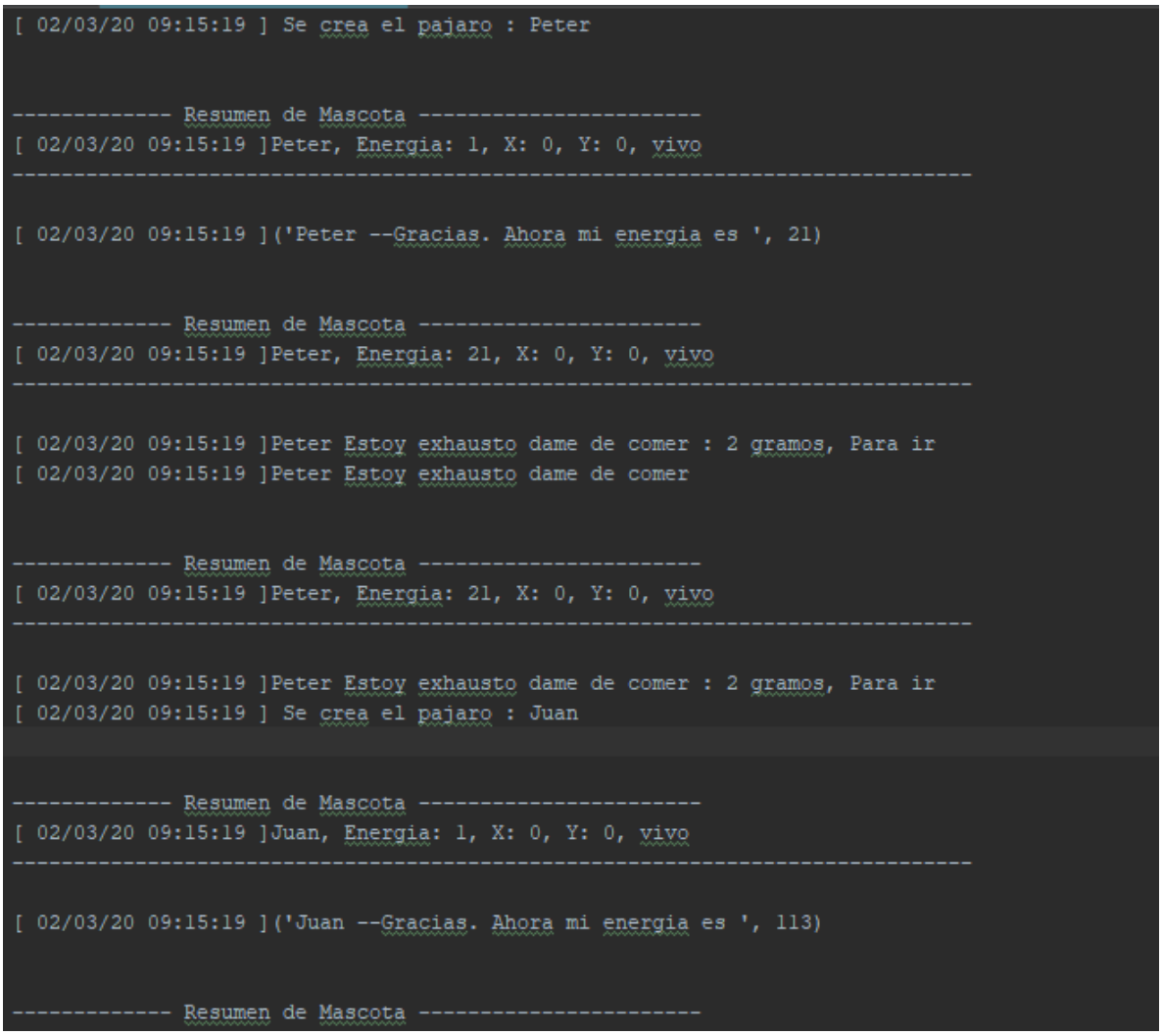
Pide el programa que el usuario presione una tecla para continuar, al momento de realizar la acción, este entra al menú del software, entonces este pide una opción.



Si ingresamos a entretenimiento este nos pide la dirección del archivo que se lee, el cual debe de ser .mascotas, este lo abre lee el archivo y genera un archivo .mascotas_resultados, en el ultimo archivo se encuentra todos los movimientos o acciones que hace nuestra mascota virtual.



Este dice que ya se genero y abrimos el archivo que se nos genero.



Este archivo es el que nos genera. Son todos las instrucciones que el usuario le da a la mascota virtual