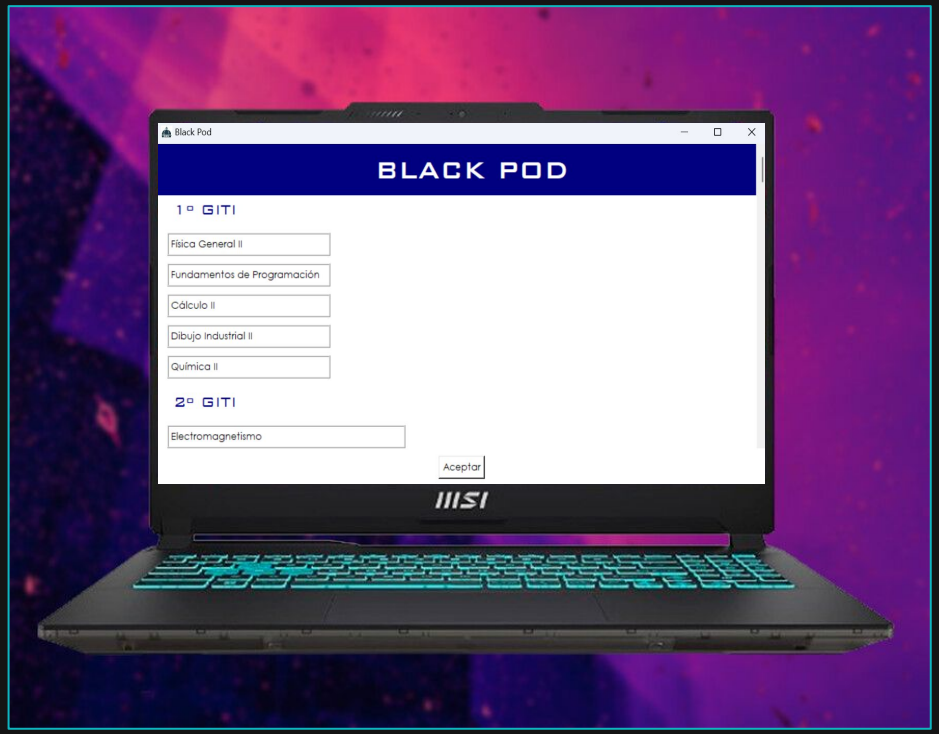
Black POD

Una herramienta ingeniosa para estudiantes ingeniosos.

Mario Montalvo, Anibal Castaño y Antonio Dies

ÍNDICE

INTRODUCCIÓN



Motivación

- Aplicar conocimientos de clase.
- Familiarización con Python y sus múltiples librerías.
- Desarrollar una herramienta de utilidad.

¿Qué es Black POD?

Herramienta de apoyo para la organización académica del alumno.

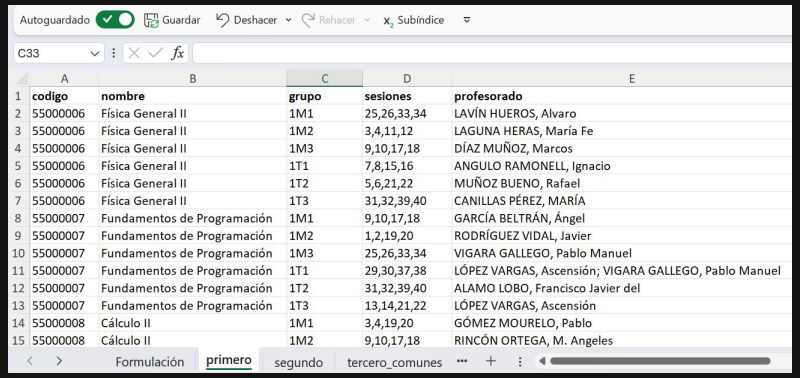
Distribución general de tareas

- Recogida y manejo de datos -> Toni
- Algoritmo principal del programa -> Mario
- Desarrollo de la interfaz -> Aníbal

RECOGIDA Y MANEJO DE DATOS

1a fase: Recogida de datos en Excel

- **Fuente de datos:** POD
- **Gestor de datos:** EXCEL
- Introducción “manual” de los datos.
  Una colaboración agilizaría esta fase
- Una hoja para cada curso
- Datos agrupados en 5 parámetros
- Sesiones definidas por números



	A	B	C	D	E
1	**codigo**	**nombre**	**grupo**	**sesiones**	**profesorado**
2	55000006	Física General II	1M1	25,26,33,34	LAVÍN HUEROS, Alvaro
3	55000006	Física General II	1M2	3,4,11,12	LAGUNA HERAS, María Fe
4	55000006	Física General II	1M3	9,10,17,18	DÍAZ MUÑOZ, Marcos
5	55000006	Física General II	1T1	7,8,15,16	ANGULO RAMONELL, Ignacio
6	55000006	Física General II	1T2	5,6,21,22	MUÑOZ BUENO, Rafael
7	55000006	Física General II	1T3	31,32,39,40	CANILLAS PÉREZ, MARÍA
8	55000007	Fundamentos de Programación	1M1	9,10,17,18	GARCÍA BELTRÁN, Ángel
9	55000007	Fundamentos de Programación	1M2	1,2,19,20	RODRÍGUEZ VIDAL, Javier
10	55000007	Fundamentos de Programación	1M3	25,26,33,34	VIGARA GALLEGO, Pablo Manuel
11	55000007	Fundamentos de Programación	1T1	29,30,37,38	LÓPEZ VARGAS, Ascensión; VIGARA GALLEGO, Pablo Manuel
12	55000007	Fundamentos de Programación	1T2	31,32,39,40	ALAMO LOBO, Francisco Javier del
13	55000007	Fundamentos de Programación	1T3	13,14,21,22	LÓPEZ VARGAS, Ascensión
14	55000008	Cálculo II	1M1	3,4,19,20	GÓMEZ MOURELO, Pablo
15	55000008	Cálculo II	1M2	9,10,17,18	RINCÓN ORTEGA, M. Angeles

	LUNES	MARTES	MIÉRCOLES	JUEVES	VIERNES
08:30 - 09:35	1	9	17	25	33
09:45 - 10:50	2	10	18	26	34
11:10 - 12:15	3	11	19	27	35
12:25 - 13:30	4	12	20	28	36
15:30 - 16:35	5	13	21	29	37
16:45 - 17:50	6	14	22	30	38
18:10 - 19:15	7	15	23	31	39
19:25 - 20:30	8	16	24	32	40

RECOGIDA Y MANEJO DE DATOS

2a fase: Interacción Excel - Código

- **Librería empleada:** Pandas. Permite un manejo de datos intuitivo y completo.
- Cada curso es un Data Frame.

```python
import pandas as pd

#A continuación se crean Data Frames con los datos de cada curso extraídos del Excel.
datos_primero=pd.read_excel("datos_segundo_semestre.xlsx","primero")
df_primero=pd.DataFrame=datos_primero
datos_segundo=pd.read_excel("datos_segundo_semestre.xlsx","segundo")
df_segundo=pd.DataFrame=datos_segundo
datos_tercero_comunes=pd.read_excel("datos_segundo_semestre.xlsx","tercero_comunes")
df_tercero_comunes=pd.DataFrame=datos_tercero_comunes
datos_tercero_especialidad=pd.read_excel("datos_segundo_semestre.xlsx","tercero_especialidad")
df_tercero_especialidad=pd.DataFrame=datos_tercero_especialidad
datos_cuarto_comunes=pd.read_excel("datos_segundo_semestre.xlsx","cuarto_comunes")
df_cuarto_comunes=pd.DataFrame=datos_cuarto_comunes
datos_cuarto_especialidad=pd.read_excel("datos_segundo_semestre.xlsx","cuarto_especialidad")
df_cuarto_especialidad=pd.DataFrame=datos_cuarto_especialidad
datos_competencias=pd.read_excel("datos_segundo_semestre.xlsx","competencias")
df_competencias=pd.DataFrame=datos_competencias
```

```python
#Crear las casillas de los turnos de primero
aux=0
if a==0:
    turnos1 = []
    for i in range(0,6):
        turnos1.append(df_primero.grupo[i] +' ~ '+ df_primero.profesorado[i])
    turnos1.append("Indiferente")
elif a==1:
    turnos1 = []
    for i in range(6,12):
        turnos1.append(df_primero.grupo[i] +' ~ '+ df_primero.profesorado[i])
    turnos1.append("Indiferente")
```

SELECCIÓN DE LAS ASIGNATURAS

¿Qué condiciones deben cumplirse?

ASIGNATURAS ESCOGIDAS

El alumno debe poder asistir a todas las asignaturas que elige

GRUPOS ESCOGIDOS

El alumno debe poder escoger a qué grupos quiere asistir en cada asignatura

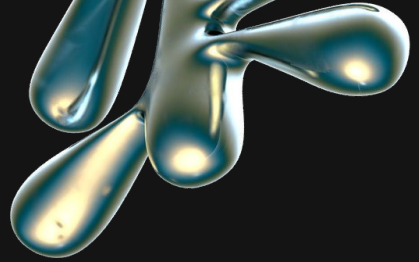
HORAS ESCOGIDAS

El alumno indica las horas en las que no puede asistir a clase

INTERFAZ

Tkinter

Librería incluida por defecto con la instalación para Windows que permite desarrollar interfaces gráficas.



Código

- Frame principal
- Ventanas emergentes
- Ventana horario

Objetivos

- Simple e intuitiva.
- Limpia y práctica.
- Que cuente con todas la utilidades necesarias.

Frame Principal

```
#BOTONES ASIGNATURAS 1º
clases1 = []
for i in range(0,30,6):
    clases1.append(df_primero.nombre[i])
for x, clase in enumerate(clases1):
    Boton_1 = Button(Frame1, text=clase,
    Boton_1.grid(row=6+x, column=2, columnspan=4, sticky="w",padx=5, pady=5)
    Boton_1.config(command=lambda x=x, btn=Boton_1: toggle1(x, variables, btn))
```

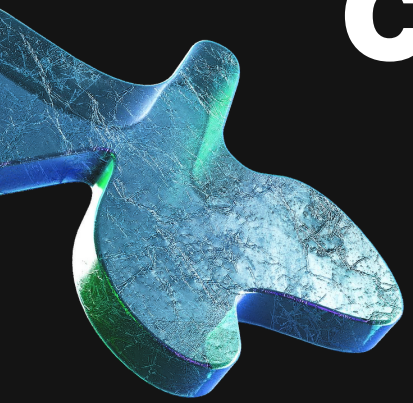
```
#BOTONES DE HORAS
for h, hora in enumerate(horario):
    Boton_lunes = Checkbutton(Frame1, text=hora, variable=variables_horas[h]
    Boton_lunes.grid(row=117+h, column=2, sticky="n", padx=5, pady=5)
```

Ventanas Emergentes:

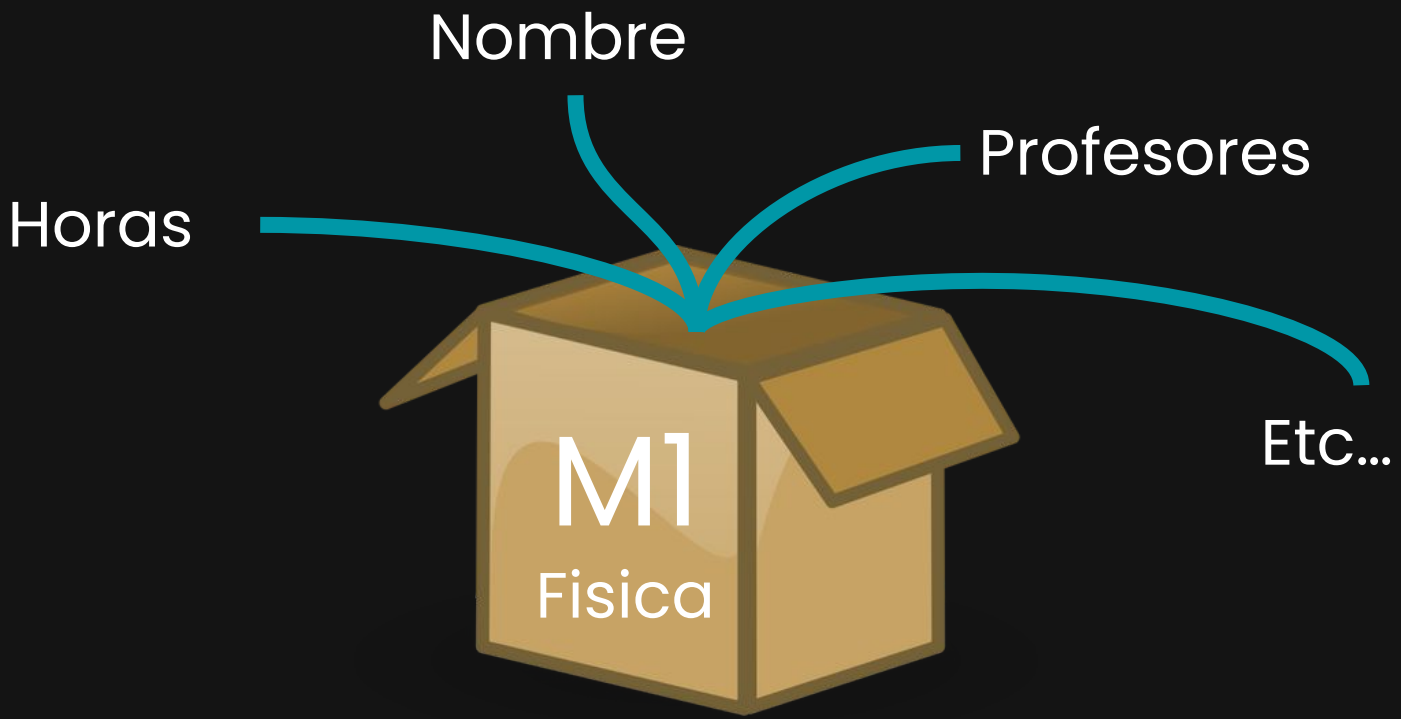
```
def abrir1(a, variables, btn):
    global ventana
    ventana = Toplevel(root)
    ventana.title("Turnos disponibles")
    ventana.config(bg='white')
    ventana.geometry("500x350")
    ventana.resizable(TRUE,TRUE)
    ventana.iconbitmap("LOGO.ico")
    main_frame1 = Frame(ventana)
    main_frame1.pack(fill=BOTH, expand=1)
    main_frame1.config(bg='white')
    my_canvas1 = Canvas(main_frame1)
    my_canvas1.pack(side=LEFT, fill=BOTH, expand=1)
    my_canvas1
    my_scrollba                                              as1.xview)
    my_scrollba
    my_canvas1
    my_canvas1                                               lregion=my_canvas1.bbox("all")))
    frame1=Fra
    anvas1                                                   00)
```

```
#Función para guardar la selección de casillas
def guardar_seleccion(z,btnb):
    global seleccion_anterior
    global seleccion_anterior_ind
    seleccion_anterior = [var.get() for var in variables]
    seleccion_anterior_ind[z] = var_ind.get()
    ventana.withdraw()  #Ocultar ventana al guardar selección
    cont = 0
    for j in range(z*6, z*6+6):
        if variables[j].get() == 1:
            cont += 1
    if cont >= 1:
        btnb.config(bg="lightsteelblue")
    else:
        btnb.config(bg="white")
```

Muchas combinaciones posibles




0000
8888


Nombre
Profesores
Horas
Etc...
M1
Fisica

M3
T1
T2

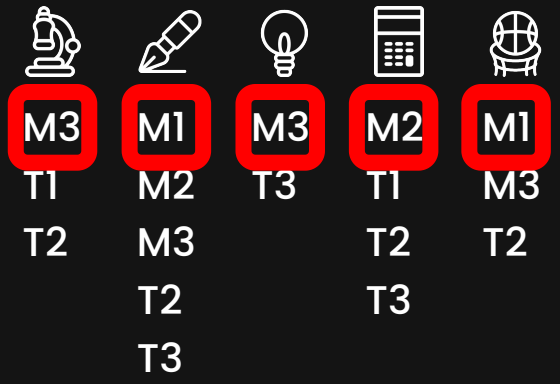
M1
M2
M3
T2
T3

M3
T3

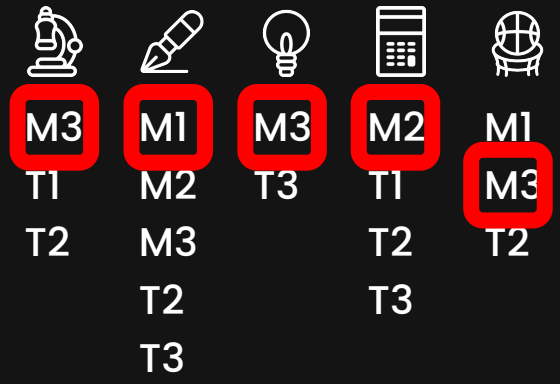
M2
T1
T2
T3

M1
M3
T2

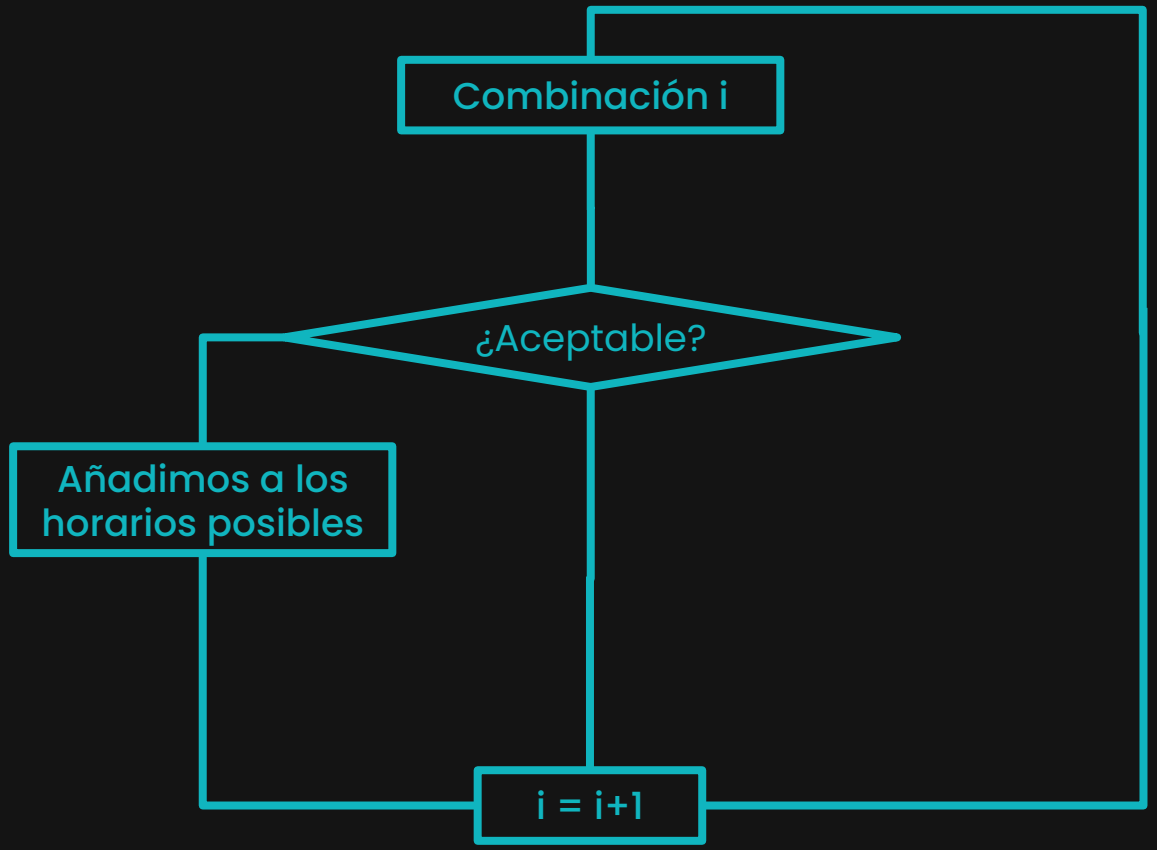
+ HORAS NO DISPONIBLES

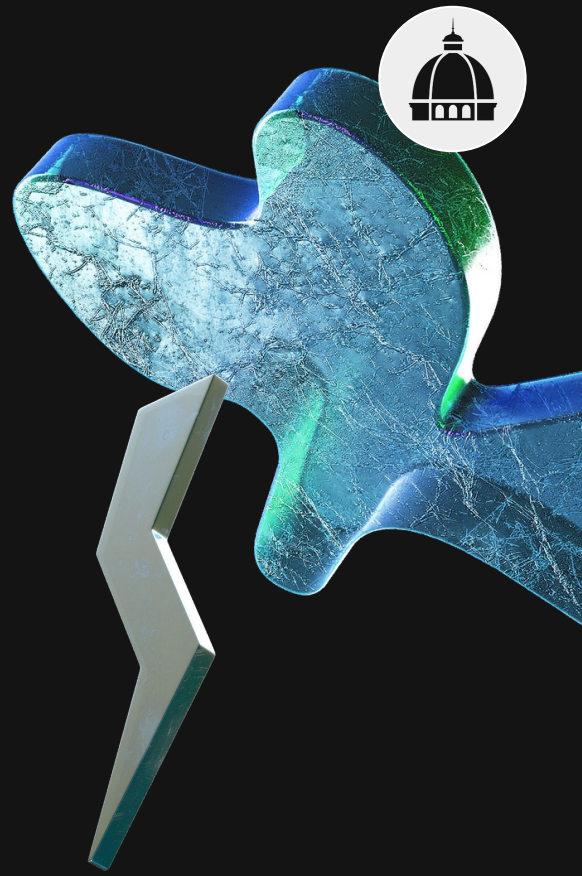

M3
T1
T2
M1
M2
M3
T2
T3
M3
T3
M2
T1
T2
T3
M1
M3
T2

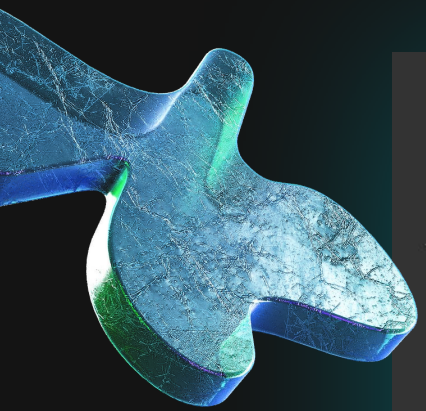
0

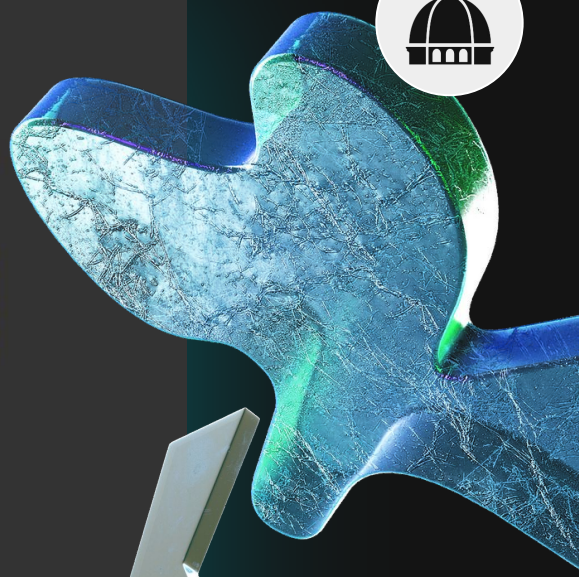

M3
T1
T2
M1
M2
M3
T2
T3
M3
T3
M2
T1
T2
T3
M1
M3
T2

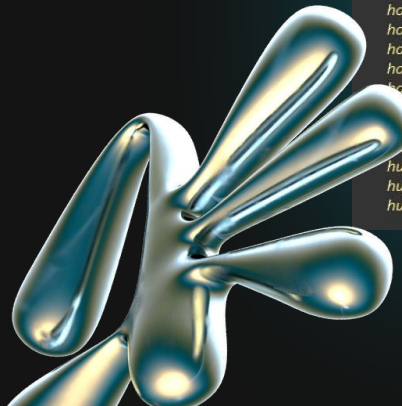
1


Combinación i
¿Aceptable?
Añadimos a los horarios posibles
i = i+1









```
for col_index in range(6):
    Grid.columnconfigure(frame, col_index, weight=3)
Grid.rowconfigure(frame, 13, weight=1)


#Aquí dibujamos los elementos basicos del horario
lunes = Label(frame, text = "Lunes", borderwidth=1, relief = "solid", font=("Century Gothic",10))
martes = Label(frame, text = "Martes", borderwidth=1, relief = "solid", font=("Century Gothic",10))
miercoles = Label(frame, text = "Miércoles", borderwidth=1, relief = "solid", font=("Century Gothic",10))
jueves = Label(frame, text = "Jueves", borderwidth=1, relief = "solid", font=("Century Gothic",10))
viernes = Label(frame, text = "Viernes", borderwidth=1, relief = "solid", font=("Century Gothic",10))
hora1 = Label(frame, text = "8:30 - 9:35", borderwidth=1, relief = "solid", font=("Century Gothic",10))
hora2 = Label(frame, text = "9:45 - 10:50", borderwidth=1, relief = "solid", font=("Century Gothic",10))
hora3 = Label(frame, text = "11:10 - 12:15", borderwidth=1, relief = "solid", font=("Century Gothic",10))
hora4 = Label(frame, text = "12:25 - 13:30", borderwidth=1, relief = "solid", font=("Century Gothic",10))
hora5 = Label(frame, text = "15:30 - 16:35", borderwidth=1, relief = "solid", font=("Century Gothic",10))
hora6 = Label(frame, text = "16:45 - 17:50", borderwidth=1, relief = "solid", font=("Century Gothic",10))
hora7 = Label(frame, text = "18:10 - 19:15", borderwidth=1, relief = "solid", font=("Century Gothic",10))
hora8 = Label(frame, text = "19:25 - 20:30", borderwidth=1, relief = "solid", font=("Century Gothic",10))

def hueco_fila(padre, fila, alto): #Esta función es para dejar huecos entra algunas filas
    vacio = Label(padre, height=alto)
    vacio.grid(row = fila, sticky= "news")

lunes.grid(row =0, column =1, sticky= "news")
martes.grid(row =0, column =2, sticky= "news")
miercoles.grid(row =0, column =3, sticky= "news")
jueves.grid(row =0, column =4, sticky= "news")
viernes.grid(row =0, column =5, sticky= "news")
hora1.grid(row =1, column =0, sticky= "news")
hora2.grid(row =2, column =0, sticky= "news")
hora3.grid(row =4, column =0, sticky= "news")
hora4.grid(row =5, column =0, sticky= "news")
hora5.grid(row =7, column =0, sticky= "news")
hora6.grid(row =8, column =0, sticky= "news")
a7.grid(row =10, column =0, sticky= "news")
a8.grid(row =11, column =0, sticky= "news")
eco_fila(frame,3,1)
hueco_fila(frame,6,1)
hueco_fila(frame,9,1)
hueco_fila(frame,12,1)
```

Leyenda

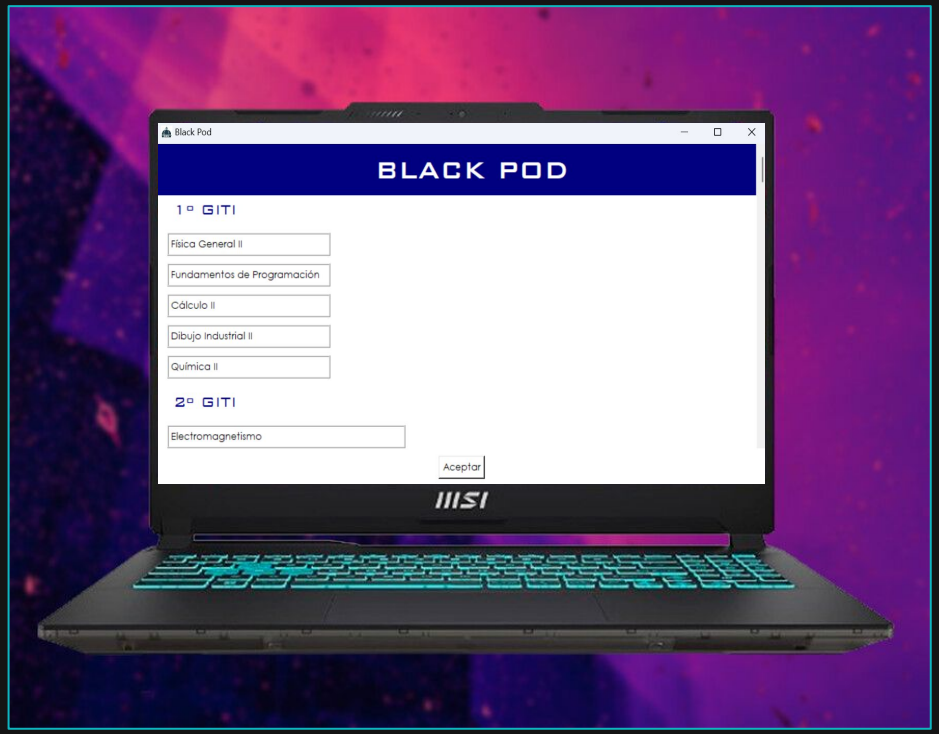
- Física General II: 55000006 - 1T3
- Fundamentos de Programación: 55000007 - 1T3
- Cálculo II: 55000008 - 1M2
- Dibujo Industrial II: 55000009 - 1M1
- Química II: 55000010 - 1M1

Posibles horarios

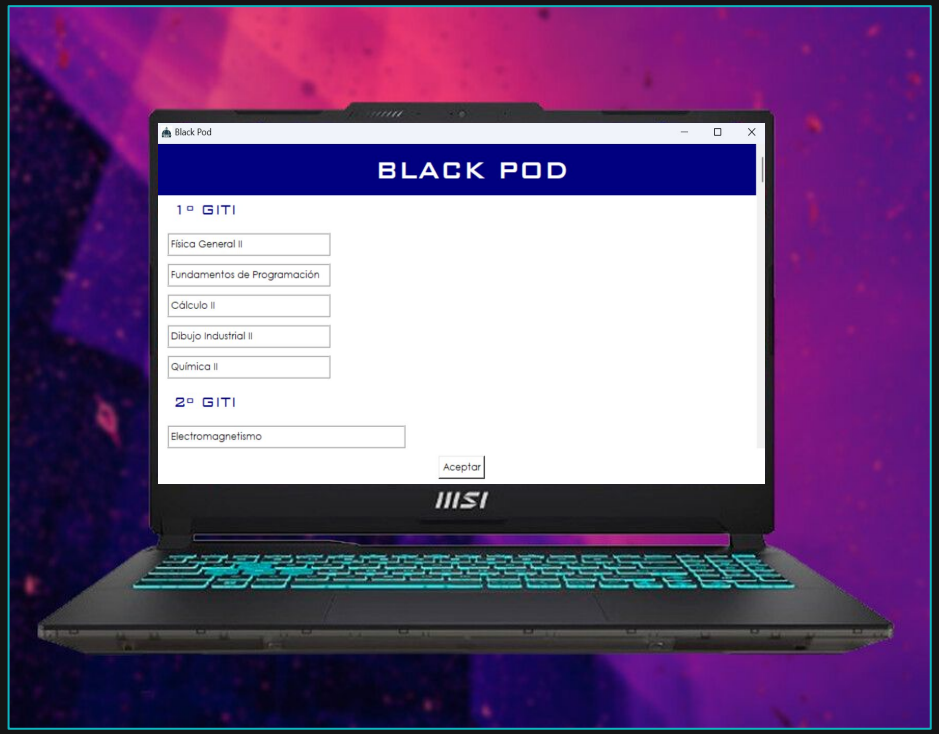
	Lunes	Martes	Miércoles	Jueves	Vier
8:30 - 9:35	55000010 - 1M1	55000008 - 1M2	55000008 - 1M2		
9:45 - 10:50	55000010 - 1M1	55000008 - 1M2	55000008 - 1M2		
11:10 - 12:15		55000010 - 1M1		55000009 - 1M1	55000009 - 1M1
12:25 - 13:30		55000010 - 1M1		55000009 - 1M1	55000009 - 1M1
15:30 - 16:35		55000007 - 1T3	55000007 - 1T3		
16:45 - 17:50		55000007 - 1T3	55000007 - 1T3		
18:10 - 19:15				55000006 - 1T3	55000006 - 1T3
19:25 - 20:30				55000006 - 1T3	55000006 - 1T3

Anterior | 34 / 1936 | Siguiente | Screenshot

CONCLUSIONES

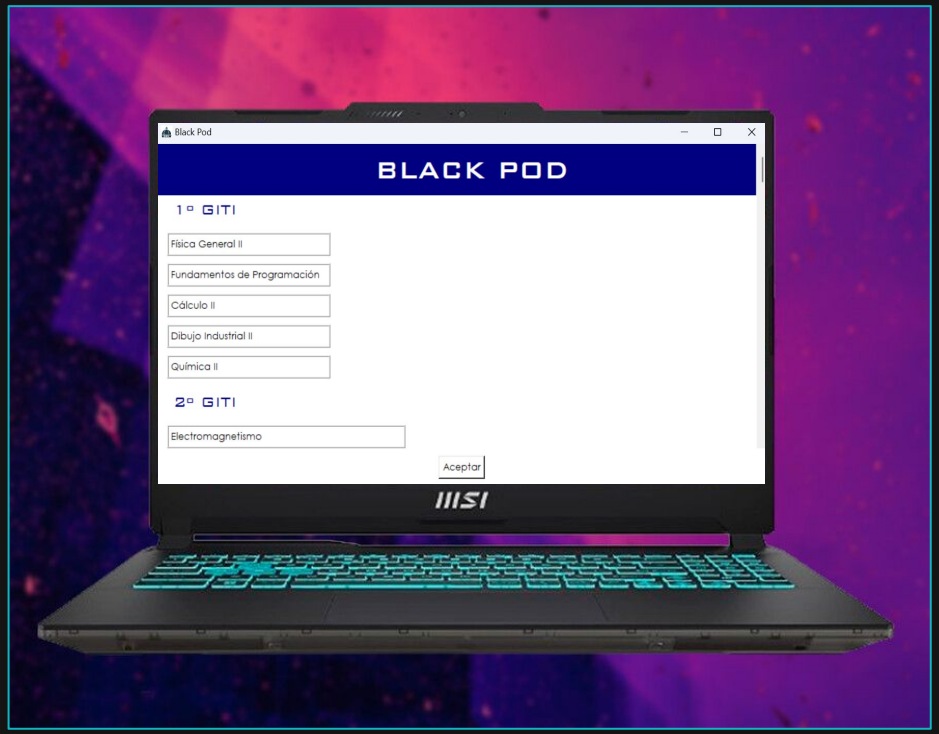


CONCLUSIONES



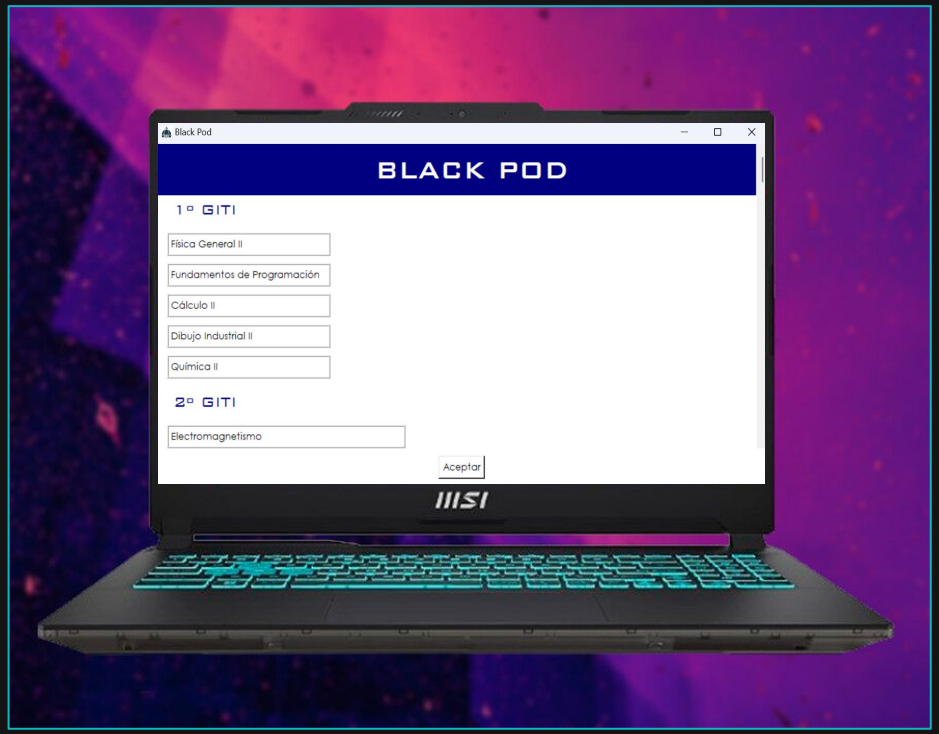
Objetivo “desarrollar una herramienta útil” cumplido.

CONCLUSIONES



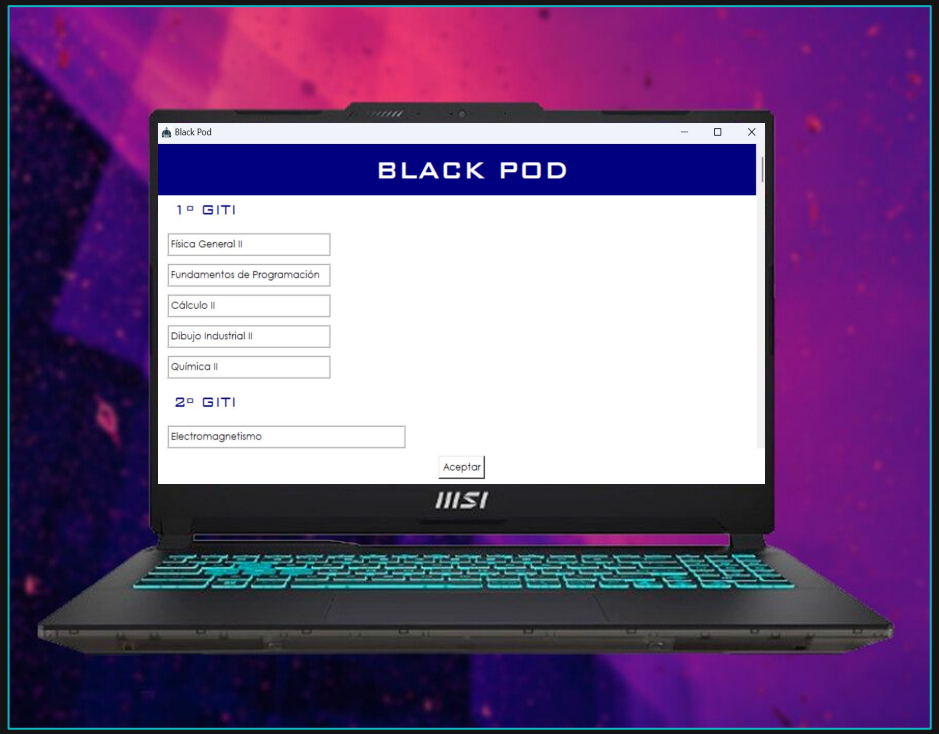
- Objetivo “desarrollar una herramienta útil” cumplido.
- Objetivos de la hoja de especificaciones cumplidos.

CONCLUSIONES



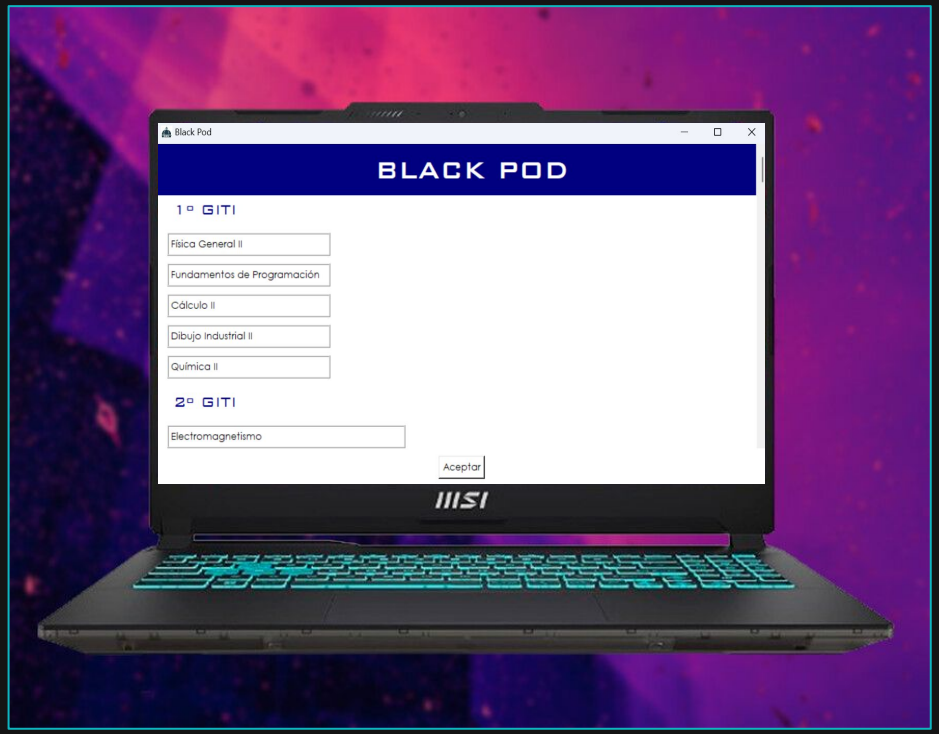
- Objetivo “desarrollar una herramienta útil” cumplido.
- Objetivos de la hoja de especificaciones cumplidos.
- Buen trabajo en equipo: organización, distribución de tareas y comunicación.

CONCLUSIONES



- Objetivo “desarrollar una herramienta útil” cumplido.
- Objetivos de la hoja de especificaciones cumplidos.
- Buen trabajo en equipo: organización, distribución de tareas y comunicación.
- Boost en nuestro aprendizaje sobre Python.

CONCLUSIONES



- Objetivo “desarrollar una herramienta útil” cumplido.
- Objetivos de la hoja de especificaciones cumplidos.
- Buen trabajo en equipo: organización, distribución de tareas y comunicación.
- Boost en nuestro aprendizaje sobre Python.
- Hemos disfrutado realizando el trabajo.

¡Muchas gracias por vuestra atención!

Keep calm and use Black POD.