

INFORME DE LABORATORIO


INFORMACIÓN BÁSICA					
ASIGNATURA:	Programación web 2 -Laboratorio Grupo A				
TÍTULO DE LA PRÁCTICA:	Lab5-Python				
NÚMERO DE PRÁCTICA:	05	AÑO LECTIVO:	2025 - A	NRO. SEMESTRE:	III
FECHA DE PRESENTACIÓN	11/04/2025	HORA DE PRESENTACIÓN	10:00:00 PM		
INTEGRANTE <ul style="list-style-type: none"> Romina Giuliana Camargo Hilachoque 				NOTA:	
DOCENTE: Carlo José Luis Corrales Delgado					

DESARROLLO
<p>Introducción:</p> <p>El presente informe expone el desarrollo del Laboratorio N.º 5 del curso de Programación Web II, cuyo objetivo fue aplicar principios fundamentales de programación usando el lenguaje Python, con énfasis en la programación orientada a objetos y el tratamiento funcional de estructuras de datos. En este laboratorio se trabajó con representaciones gráficas basadas en listas de cadenas de texto, implementando una clase denominada Picture, la cual encapsula diversas transformaciones visuales como espejos, negativos, repeticiones y superposiciones. Para la representación visual de los resultados se empleó la librería pygame, ejecutada en un entorno virtual configurado con virtualenv. La actividad se desarrolló en el contexto de una simulación gráfica del ajedrez, permitiendo reforzar el diseño modular del código, la reutilización de métodos y la claridad en la separación entre datos y vista.</p>

Commits on May 11, 2025

crenado un ejercicio extra para probar el metodo rotate

romich1307 committed 6 hours ago

61588e8  <>

implementando rotate a pesar de que no lo usaremos


romich1307 committed 6 hours ago

c07ae13  <>

Commits on May 10, 2025


añadiendo README

romich1307 committed yesterday

1c568ee  <>


Arreglando los comentarios

romich1307 committed yesterday

8f6ad9c  <>

arreglando ejercicio 7 porque no mostraba bien las michas ni el tablero

romich1307 committed yesterday

b81c1c8  <>

Realizando el dibujo del ejercicio7

romich1307 committed yesterday

7f726cf  <>


Arreglando el ejercicio6

romich1307 committed yesterday

19d1948  <>


Realizando el dibujo del ejercicio6

romich1307 committed yesterday

39d3167  <>


Realizando el dibujo del ejercicio5

romich1307 committed yesterday

4694521  <>


arreglando dibujo del ejercicio4

romich1307 committed yesterday

80f3287  <>


Realizando el dibujo del tercer ejercicio

romich1307 committed yesterday

3f88a5d  <>


arreglando dibujo del ejercicio2

romich1307 committed yesterday

608bb25  <>

Realizando el dibujo del segundo ejercicio

romich1307 committed yesterday

4678a59  <>


Realizando el dibujo del primer ejercicio

romich1307 committed yesterday

895bbfb  <>


Ejercicio 2a: unir dos piezas rock con join()

romich1307 committed yesterday

2695e13  <>


Implementado método verticalRepeat en Picture

romich1307 committed yesterday

ff6d75d  <>

Implementado método horizontalRepeasen Picture

romich1307 committed yesterday

91dc652  <>

Implementado método under en Picture

romich1307 committed yesterday

521da87  <>

Implementado método up en Picture

romich1307 committed yesterday

#97711f  <>


Implementado método join en Picture

romich1307 committed yesterday

3c19acc  <>

Implementado método negative en Picture

romich1307 committed yesterday

3a568d7  <>

Implementado método verticalMirror en Picture

romich1307 committed yesterday

8246358  <>


descargando pygame

romich1307 committed yesterday

c2e968f  <>

Subiendo tarea de ajedrez en LAB5

romich1307 committed yesterday

19558e3  <>

Objetivos

- Practicar principios fundamentales de programación usando Python.
- Implementar una separación clara entre el modelo de datos (listas de strings) y la vista (dibujos gráficos).
- Aplicar conceptos de programación orientada a objetos y operaciones funcionales sobre estructuras.

Temas Abordados

- Listas y sublistas
- Ciclos y comprensión de listas
- Programación orientada a objetos (POO)
- Uso de entornos virtuales y paquetes con pip

Sistema y Herramientas Utilizadas

- **Sistema Operativo:** Windows 10 (64 bits)
- **Editor:** Visual Studio Code
- **Entorno Virtual:** creado con virtualenv
- **Librería Gráfica:** Pygame
- **Repositorio Base:**
<https://github.com/rescobedoq/pw2/tree/main/labs/lab04/Tarea-del-Ajedrez>

Marco Teórico

Python es un lenguaje interpretado que permite manipular estructuras de datos de manera eficiente. En este laboratorio se usó una arquitectura basada en clases para definir piezas de ajedrez representadas como listas de strings. Se aplicaron principios de programación funcional (como map, join, zip) y orientada a objetos. El uso de entornos virtuales permitió aislar dependencias con pip, facilitando la instalación de paquetes como pygame para la visualización.

Desarrollo

Se implementó la clase Picture que representa una figura gráfica basada en una lista de strings (img). Los métodos desarrollados permiten modificar visualmente las piezas:

- verticalMirror()
- horizontalMirror()
- negative()
- join(picture)
- up(picture)
- under(picture)
- horizontalRepeat(n)
- verticalRepeat(n)

Estos métodos fueron usados en archivos como ejercicio2a.py a ejercicio2g.py, donde se generaron combinaciones de piezas y finalmente el tablero completo.


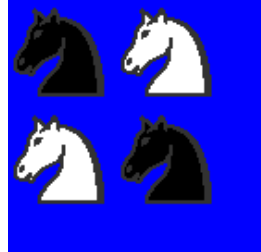
Desarrollo por Ejercicios

a. Caballos enfrentados (blanco-negro)

Métodos utilizados: join(), verticalMirror(), up()

Se toma un caballo blanco y se le aplica verticalMirror() para obtener el negro. Luego se usa join() para unirlos en una fila y up() para colocar una copia invertida debajo.

```
1  from interpreter import draw
2  from chessPictures import *
3
4  fila1 = knight.join(knight.negative())
5  fila2 = knight.negative().join(knight)
6  imagen = fila1.up(fila2)
7
8  draw(imagen)
```

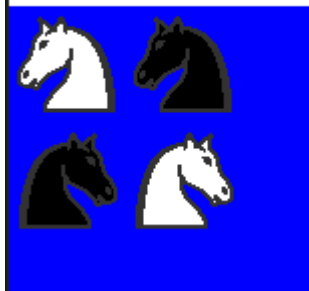
 pygame window

b. Reflejo horizontal de los caballos

Métodos utilizados: horizontalMirror(), join(), up()

Se realiza un espejo horizontal a los caballos, haciendo que se intercambien arriba-abajo. Se usa join() para formar una fila, luego up() para superponer el reflejo.

```
1 from interpreter import draw
2 from chessPictures import *
3
4 fila1 = knight.negative().verticalMirror().join(knight.verticalMirror())
5 fila2 = knight.join(knight.negative())
6 cuadro = fila1.up(fila2)
7
8 draw(cuadro)
```


 pygame window

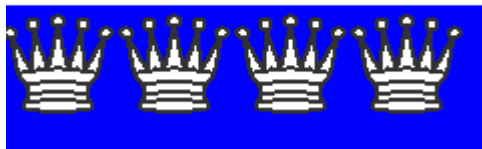
c. Repetición horizontal de la reina

Método utilizado: `negative()`, `horizontalRepeat(n)`

La reina se transforma en negativa para contrastar y luego se repite 4 veces de forma horizontal.

```
1  from interpreter import draw
2  from chessPictures import *
3
4  imagen = queen.horizontalRepeat(4)
5  draw(imagen)
6
```

 pygame window




d. Una fila del tablero de ajedrez

Métodos utilizados: `join()`, `horizontalRepeat(n)`

Se alternan `square` y `square.negative()` para crear un patrón de casillas negras y blancas.

```
1  from interpreter import draw
2  from chessPictures import *
3
4  fila = square.join(square.negative()).horizontalRepeat(4)
5  draw(fila)
```


 pygame window



e. La siguiente fila del tablero**Método utilizado:** up()


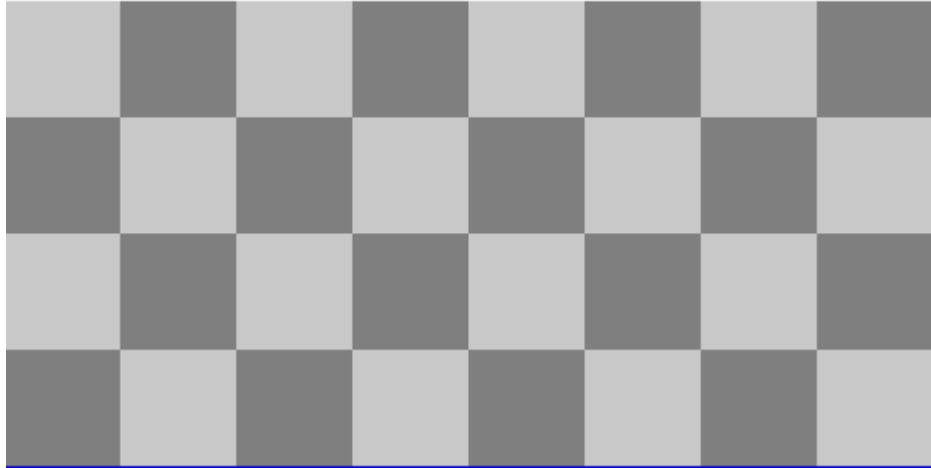
Se alternan square.negative() y square para crear el otro patrón de casillas negras y blancas.

```
1  ✓ from interpreter import draw
2    from chessPictures import *
3
4    fila = square.negative().join(square).horizontalRepeat(4)
5    draw(fila)
```

 pygame window**f. Tablero (4x8)****Método utilizado:** verticalRepeat(n)

Se repite el patrón de dos filas del tablero 2 veces verticalmente.

```
1  from interpreter import draw
2  from chessPictures import *
3
4  fila1 = square.join(square.negative()).horizontalRepeat(4)
5  fila2 = square.negative().join(square).horizontalRepeat(4)
6  tablero = fila2.up(fila1).verticalRepeat(2)
7  draw(tablero)
```

 pygame window


g. Tablero completo (8x8)


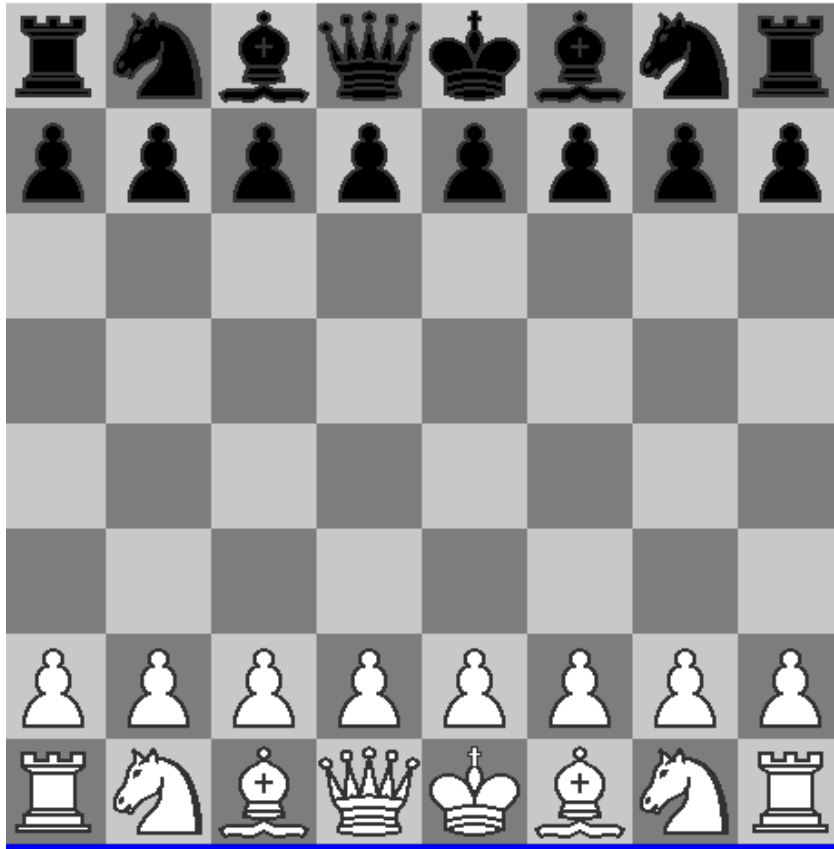
Método utilizado: verticalRepeat(4) ,join, up, under

Construir un tablero de ajedrez de 8x8 casillas que incluya todas las piezas en su ubicación inicial (tanto blancas como negras), utilizando métodos de repetición (horizontalRepeat, verticalRepeat) y superposición (up, under).

```

1  from interpreter import draw
2  from chessPictures import *
3
4  # Crear el tablero (cuadro gris claro y oscuro)
5  fila1 = square.join(square.negative()).horizontalRepeat(4)
6  fila2 = square.negative().join(square).horizontalRepeat(4)
7  tablero = fila2.up(fila1).verticalRepeat(4)
8
9  # Fila de piezas negras
10 fila_negras = rock.negative().join(knight.negative()).join(bishop.negative()).join(queen.negative()).join(king.negative()).join(bishop.negative()).join(knight.negative()).
11
12 # Peones negros
13 peones_negros = pawn.negative().horizontalRepeat(8)
14
15 # Peones blancos
16 peones_blancos = pawn.horizontalRepeat(8)
17
18 # Fila de piezas blancas
19 fila_blancas = rock.join(knight).join(bishop).join(queen).join(king).join(bishop).join(knight).join(rock)
20
21 # Creando las dos filas vacías que usaremos
22 espacio1 = square.join(square.negative()).horizontalRepeat(4).verticalRepeat(1)
23 espacio2 = square.negative().join(square).horizontalRepeat(4).verticalRepeat(1)
24
25 # Unir las piezas en orden correcto: blancas abajo, negras arriba
26 piezas = fila_blancas.up(peones_blancos).up(espacio2).up(espacio1).up(espacio2).up(espacio1).up(peones_negros).up(fila_negras)
27
28 # Superponer las piezas sobre el tablero
29 imagen = tablero.under(piezas)
30
31 draw(imagen)

```


 pygame window

Repositorio Final

El código completo y los commits están disponibles en el siguiente repositorio:

https://github.com/romich1307/PWEB2-LAB_A/tree/main/LAB5

Conclusiones

- Se logró implementar con éxito transformaciones visuales sobre listas de strings.
- La librería pygame permitió representar gráficamente el estado de las piezas.
- El uso de GitHub facilitó el control de versiones y respaldó el trabajo realizado.

REFERENCIAS EN FORMATO APA7

- https://www.w3schools.com/python/python_reference.asp
- <https://docs.python.org/3/tutorial/>

RETROALIMENTACIÓN