Informe de Laboratorio 05

Tema: Python

Nota

Estudiantes	Escuela	Asignatura
William Herderson Choquehuanca Berna wchoquehuancab@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Laboratorio de P.Web Semestre: III Código: 20233469

Laboratorio	Tema	Duración
05	Python	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - A	Del 20 de mayo 2024	Al 24 de mayo 2024

1. Actividades

- Cree un ambiente virtual para esta práctica.
- Instale Django en el ambiente virtual.
- Cree un directorio e inicialize un repositorio git en el y cree un proyecto github y enlacelos.
- Cree un archivo .gitignore según este repositorio https://github.com/django/django/blob/main/.gitignore
- Crear un proyecto en Django que maneje una tabla de Productos y una tabla de Ventas
- Crear las apps necesarias en Django
- Crear las vistas y formularios necesarios en Django para que se pueda ingresar una venta de varios productos.

2. Ejercicios Propuestos

- Implementa un Sistema en Django que maneje una tabla de Alumnos, una de Cursos y una de NotasAlumnosPorCurso y que permita ingresar a nuevos alumnos, nuevos cursos y finalmente permita ingresar las notas por curso. Laboratorio realizado en grupos de 2 estudiantes. Crear un unico proyecto y compartir github con el profesor (CarloCorrales010). No olvidar enviar video a flipgrip de manera personal.

3. Equipos, materiales y temas utilizados

- Sistema operativo de 64 bits, procesador basado en x64.
- Latex.
- git version 2.41.0.windows.1
- Lenguaje Python.
- IDE Visual Sudio Code.

4. URL Github, Video

- URL del Repositorio GitHub.
- `https://github.com/WilliamLawrence25/PWeb2/tree/main/Lab5`
- URL para el video flipgrid.
- `https://flip.com/s/QRG7DhggB2kt`

5. Capturas de los ejercicios propuestos

5.1. Ejercicio 1

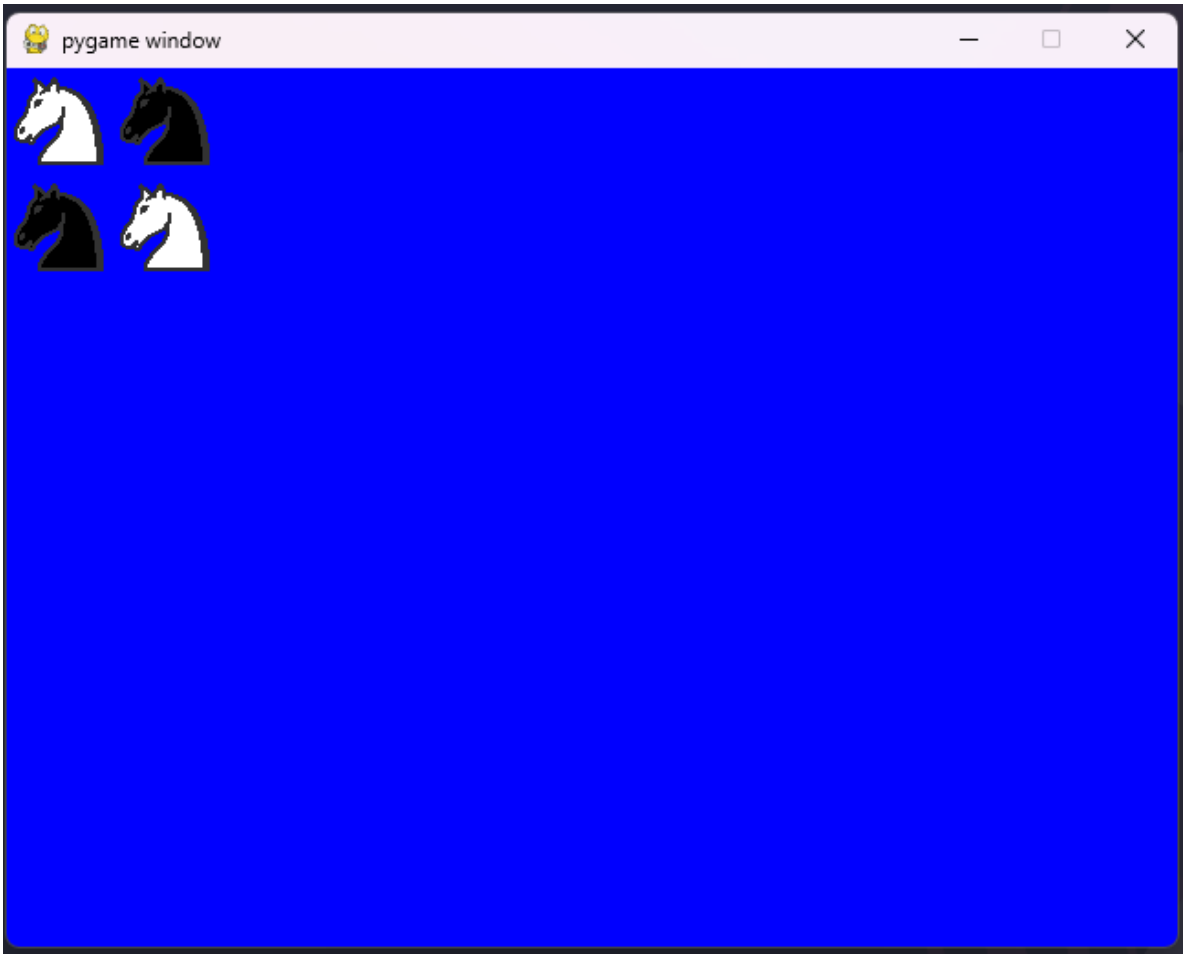
```python
from colors import *

class Picture:

  def __init__(self, img):
    self.img = img;

  def __eq__(self, other):
    return self.img == other.img

  def _invColor(self, color):
    if color not in inverter:
      return color
    return inverter[color]

  def verticalMirror(self):
    """ Devuelve el espejo vertical de la imagen """
    vertical = []
    vertical = [line[::-1] for line in self.img]
    return Picture(vertical)

  def horizontalMirror(self):
    """ Devuelve el espejo horizontal de la imagen """
    horizontal = self.img[::-1]
    return Picture(horizontal)

  def negative(self):
    """ Devuelve un negativo de la imagen """
    negative = [''.join(self._invColor(char) for char in line) for line in self.img]
    return Picture(negative)

  def join(self, p):
    """ Devuelve una nueva figura poniendo la figura del argumento
        al lado derecho de la figura actual """
    join_img = [self_line + p_line for self_line, p_line in zip(self.img, p.img)]
    return Picture(join_img)

  def up(self, p):
    return Picture(self.img + p.img)
    return Picture(self.img + p.img)

  def under(self, p):
    """ Devuelve una nueva figura poniendo la figura p sobre la
        figura actual """
    max_height = max(len(self.img), len(p.img))
    max_width = max(max(len(line) for line in self.img), max(len(line) for line in p.img))

    new_img_self = [line.ljust(max_width) for line in self.img] + [' ' * max_width] * (max_height - len(self.img))
    new_img_p = [line.ljust(max_width) for line in p.img] + [' ' * max_width] * (max_height - len(p.img))

    combined_img = []
    for line_self, line_p in zip(new_img_self, new_img_p):
        combined_line = ''.join(char_p if char_p != ' ' else char_self for char_self, char_p in zip(line_self, line_p))
        combined_img.append(combined_line)

    return Picture(combined_img)

  def horizontalRepeat(self, n):
    """ Devuelve una nueva figura repitiendo la figura actual al costado
        la cantidad de veces que indique el valor de n """
    repeat_img = [line * n for line in self.img]
    return Picture(repeat_img)

  def verticalRepeat(self, n):
    repeat_img = self.img * n
    return Picture(repeat_img)

  #Extra: Sólo para realmente viciosos
  def rotate(self):
    """Devuelve una figura rotada en 90 grados, puede ser en sentido horario
    o antihorario"""
    rotated = [''.join(row) for row in zip(*self.img)]
    rotated = rotated[::-1]
    return Picture(rotated)

```

5.2. Ejercicio 2a

- Python

```python
from interpreter import draw
from chessPictures import *

fila1 = knight.join(knight.negative())
fila2 = fila1.negative()
todo = fila1.up(fila2)
draw (todo)
```

- Tablero

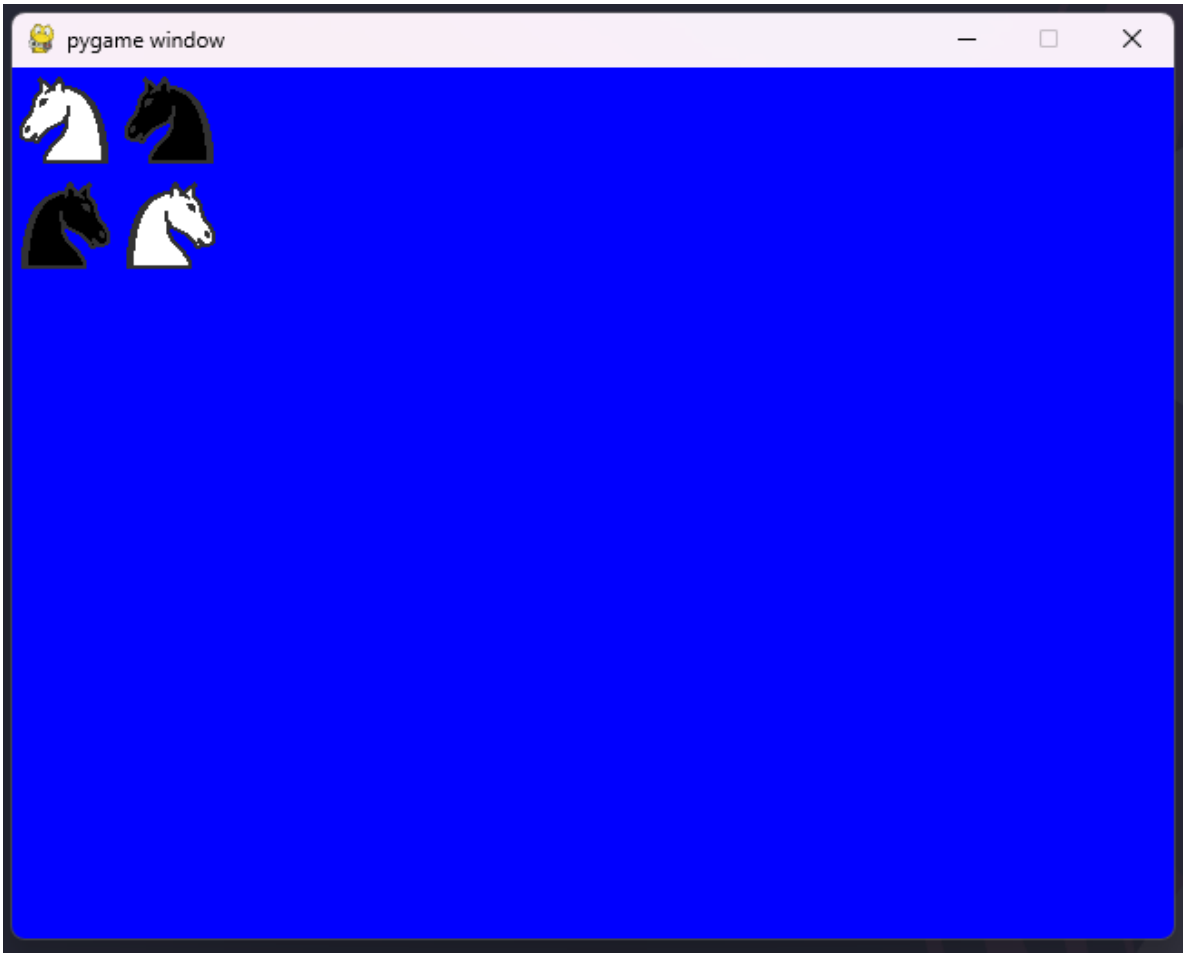


5.3. Ejercicio 2b

- Python

```python
from interpreter import draw
from chessPictures import *

fila1 = knight.join(knight.negative())
fila2 = (fila1).verticalMirror()
todo = fila1.up(fila2)
draw (todo)
```

- Tablero

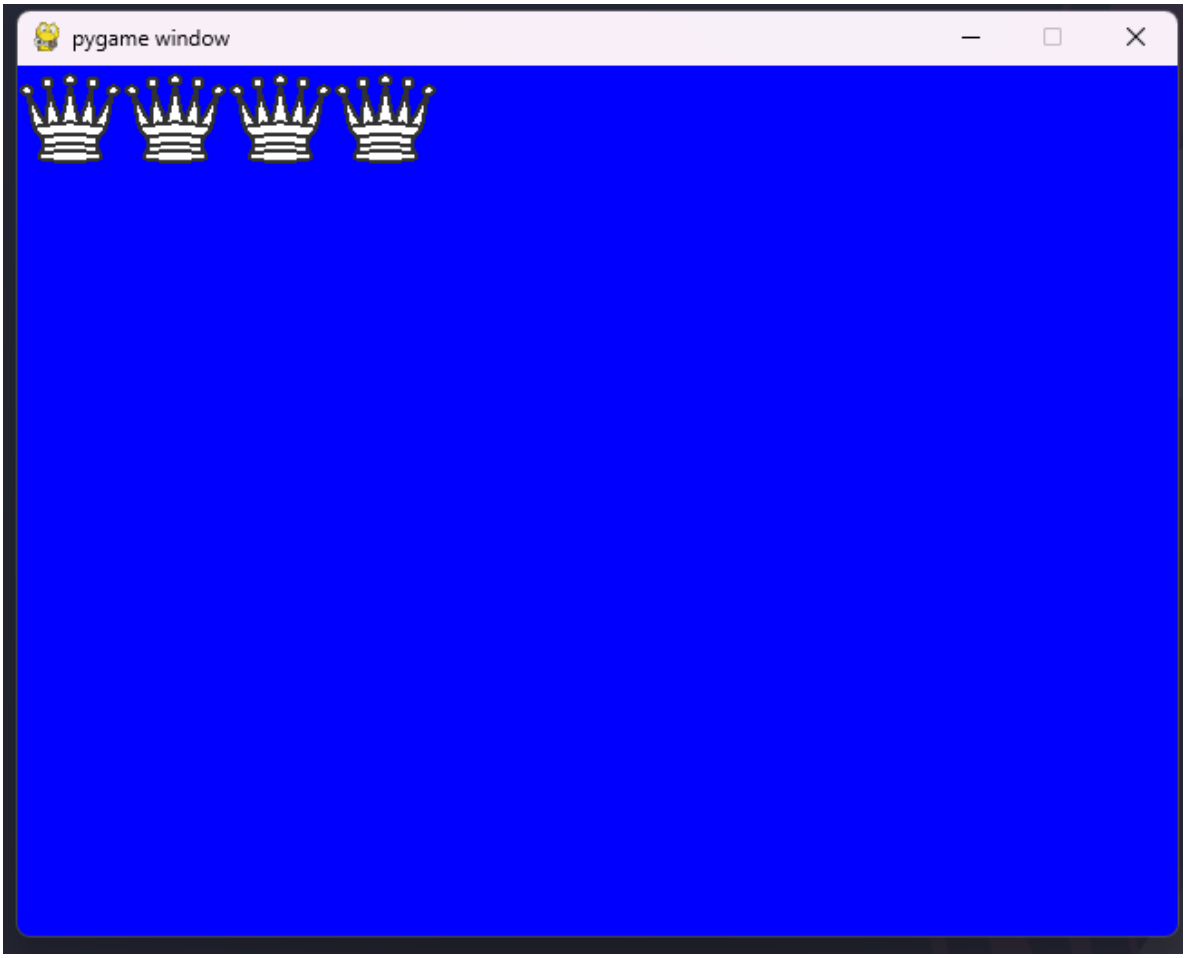


5.4. Ejercicio 2c

- Python

```python
from interpreter import draw
from chessPictures import *

draw (queen.horizontalRepeat(4))
```

- Tablero

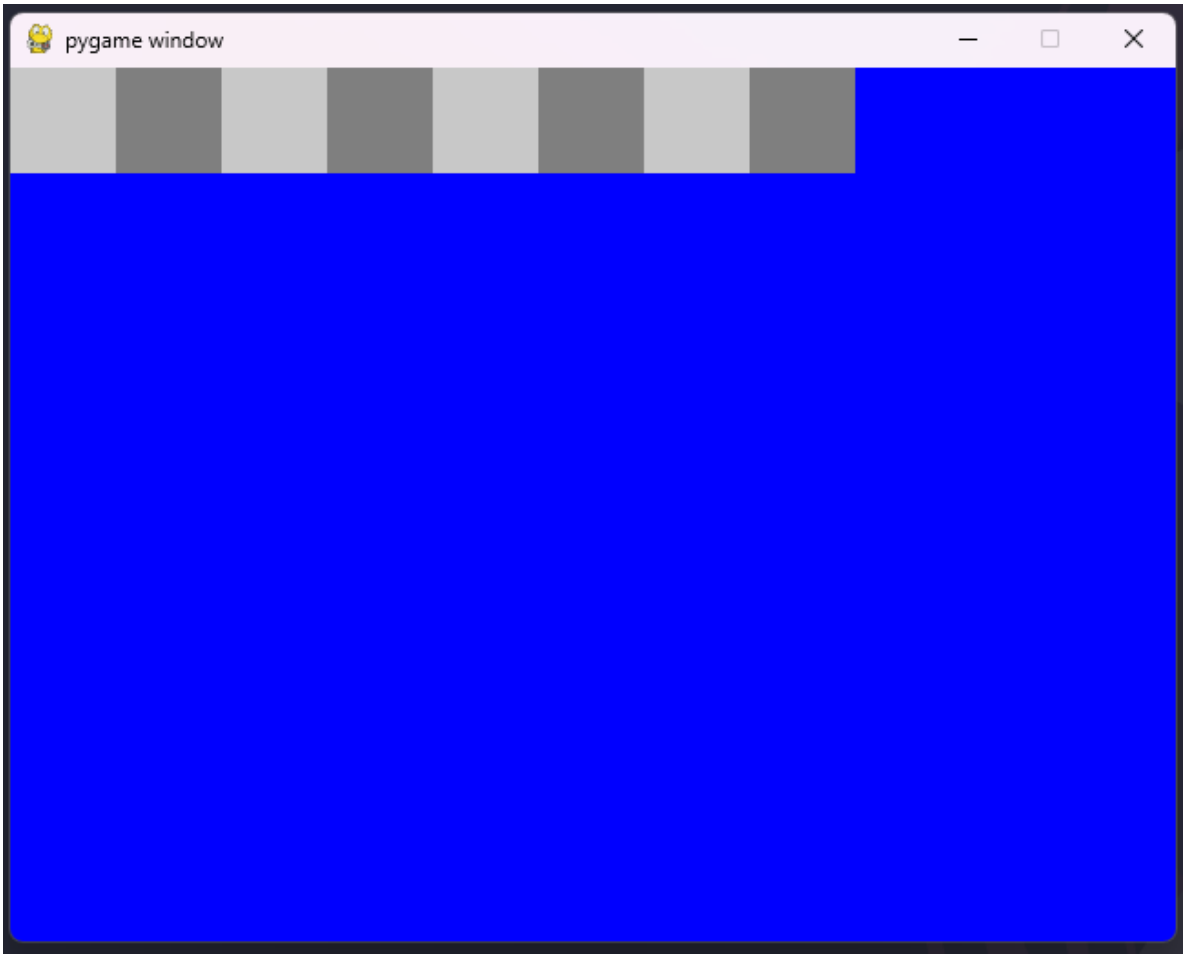


5.5. Ejercicio 2d

- Python

```python
from interpreter import draw
from chessPictures import *

draw (square.join(square.negative()).horizontalRepeat(4))
```

- Tablero

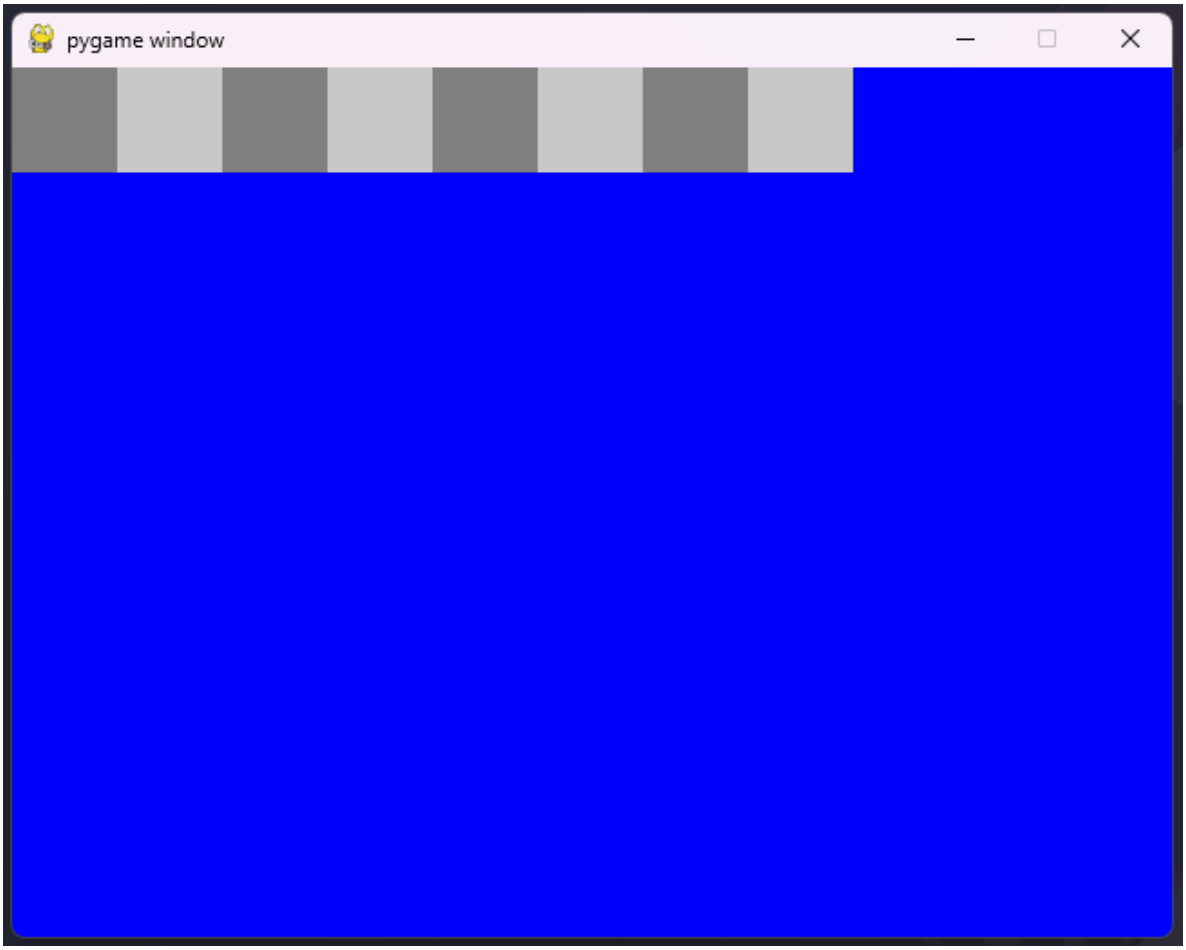


5.6. Ejercicio 2e

- Python

```python
from interpreter import draw
from chessPictures import *

draw (square.negative().join(square).horizontalRepeat(4))
```

- Tablero

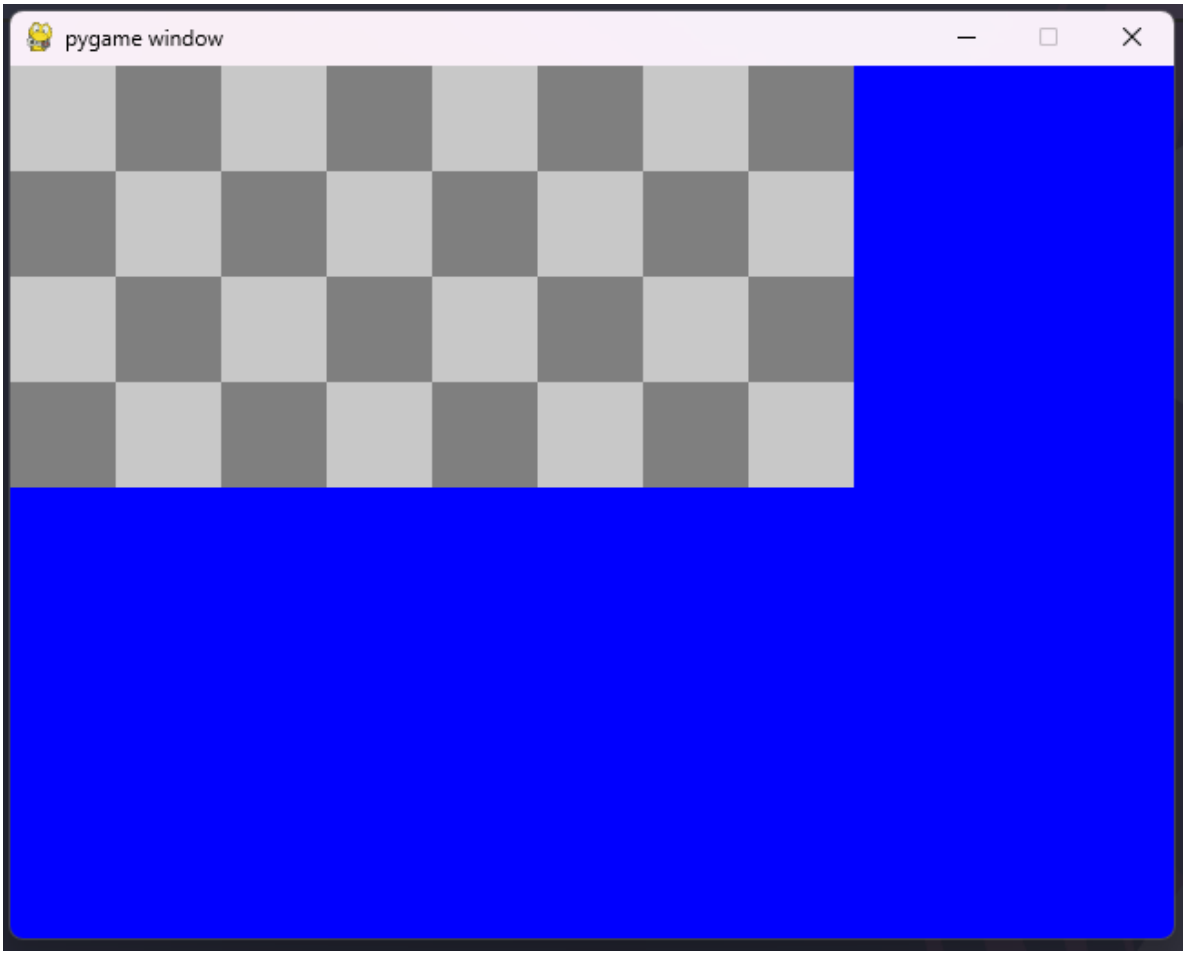


5.7. Ejercicio 2f

- Python

```python
from interpreter import draw
from chessPictures import *

type1 = (square.join(square.negative()).horizontalRepeat(4))
type2 = (square.negative().join(square).horizontalRepeat(4))
total = type1.up(type2).verticalRepeat(2)

draw (total)
```

- Tablero

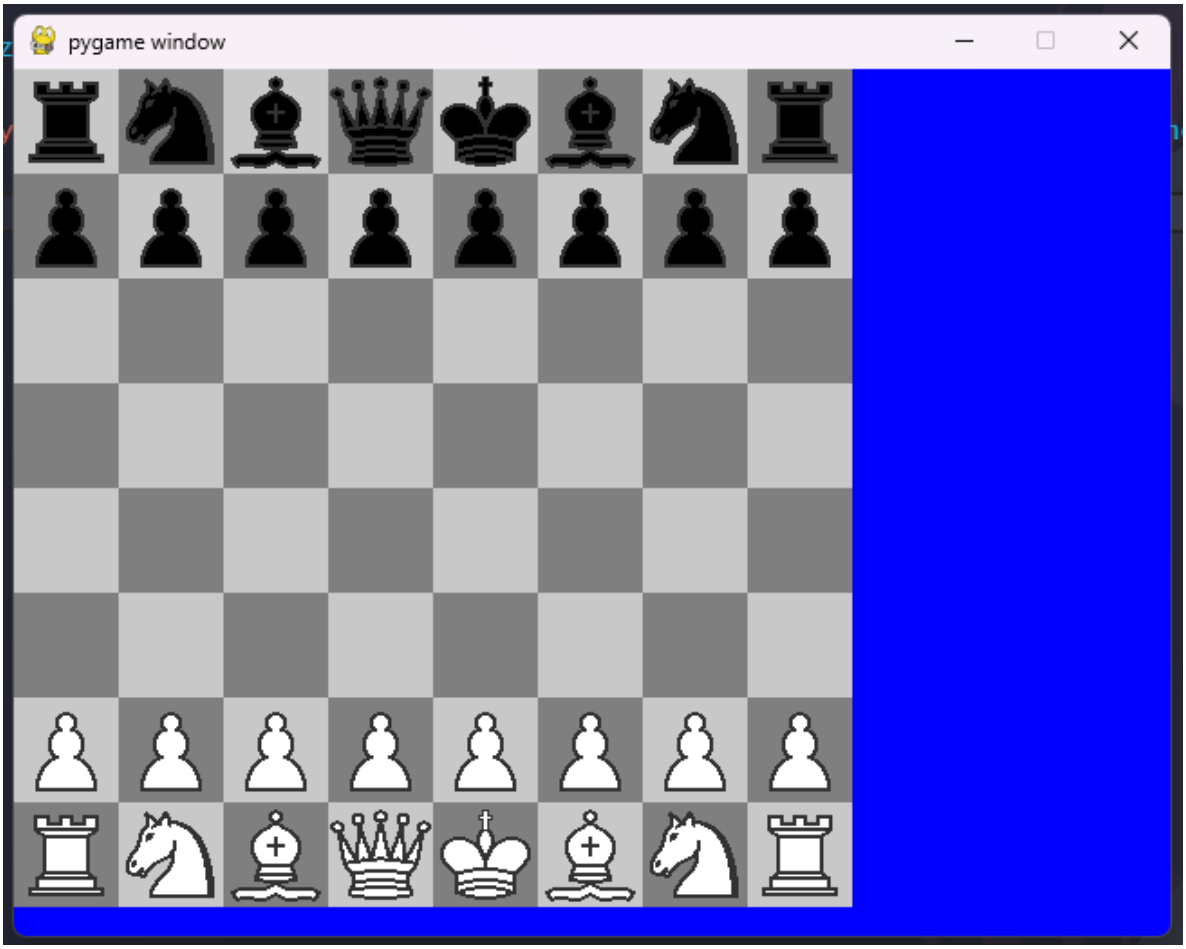


5.8. Ejercicio 2g

- Python

```python
from interpreter import draw
from chessPictures import *

type1 = (square.join(square.negative()).horizontalRepeat(4))
type2 = (square.negative().join(square).horizontalRepeat(4))

chess1s = type2.under(rock.join(knight).join(bishop).join(queen).join(king).join(bishop).join(knight).join(rock)).negative()
chess2s = type1.under(pawn.horizontalRepeat(8)).negative()

table = chess1s.up(chess2s.up(type1.up(type2).verticalRepeat(2))).up(chess2s.negative()).up(chess1s.negative())

draw (table)
```

- Tablero


pygame window

6. Referencias

- https://github.com/rescobedoq/pw2/tree/main/labs/lab04/Tarea-del-Ajedrez