

Informe de Laboratorio 04

Tema: Python

Estudiante	Escuela	Asignatura
Chara Condori Jean Carlo	Escuela Profesional de Ingeniería de Sistemas	Programación Web 2

Laboratorio	Tema	Duración
04	Python	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023-A	29 Mayo 2023	09 Junio 2023

1 Tarea

- Para resolver los siguientes ejercicios sólo está permitido usar ciclos, condicionales, definición de listas por comprensión, sublistas, map, join, (+), lambda, zip, append, pop, range.
- Implemente los métodos de la clase Picture. Se recomienda que implemente la clase picture por etapas, probando realizar los dibujos que se muestran en las siguientes preguntas.
- Usando únicamente los métodos de los objetos de la clase Picture dibuje las siguientes figuras (invoque a draw):

2 Resolucion

- REPOSITORIO GITHUB:
- https://github.com/JeanChara/pweb2_lab04
- Código clase picture:

```
1  from colors import *
2  class Picture:
3      def __init__(self, img):
4          self.img = img
5
6      def __eq__(self, other):
7          return self.img == other.img
8
9      def _invColor(self, color):
10         if color not in inverter:
11             return color
12         return inverter[color]
13
14     def verticalMirror(self):
15         """ Devuelve el espejo vertical de la imagen """
16         vertical = []
17         for i in range(len(self.img)):
18             vertical.append(self.img.pop())
19     return Picture(vertical)
20
21     def horizontalMirror(self):
22         """ Devuelve el espejo horizontal de la imagen """
23         horizontal = []
24         for i in range(len(self.img)):
25             textoInv = ""
26             for caracter in self.img[i]:
27                 textoInv = caracter + textoInv
28             horizontal.append(textoInv)
29
30     return Picture(horizontal)
31
32     def negative(self):
33         """ Devuelve un negativo de la imagen """
34         neg = []
35         for linea in self.img:
36             lineaNeg = ""
37             for caracter in linea:
38                 lineaNeg += ""+self._invColor(caracter)
39         neg.append(lineaNeg)
```

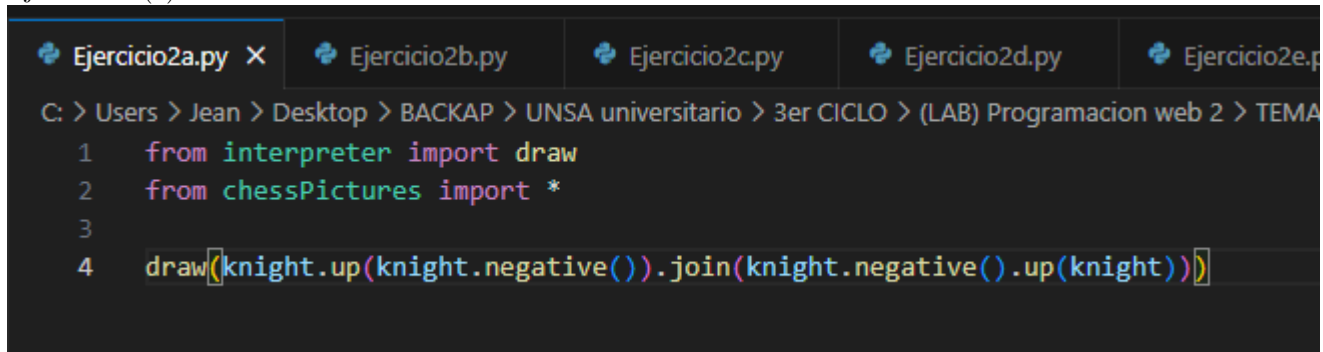
```
40     return Picture(neg)
41
42     def join(self, p):
43         """ Devuelve una nueva figura poniendo la figura del argumento
44         | al lado derecho de la figura actual """
45         nuevaFigura = []
46         for i in range(len(self.img)):
47             nuevaFigura.append(self.img[i] + p.img[i]) # concatenamos de fila en fila
48         return Picture(nuevaFigura)
49
50     def up(self, p):
51         """ Devuelve una nueva figura poniendo la figura p debajo de la figura actual """
52         nuevaFigura = []
53         for i in self.img:
54             nuevaFigura.append(i)
55         for j in p.img:
56             nuevaFigura.append(j)
57         return Picture(nuevaFigura)
58
59     def under(self, p):
60         """ Devuelve una nueva figura poniendo la figura p sobre la
61         | figura actual """
62         nuevaFigura = []
63
64         for i in range(len(self.img)):
65             aux = ""
66             for j in range(len(self.img[i])):
67                 if p.img[i][j] == " ":
68                     aux += self.img[i][j]
69                 else:
70                     aux += p.img[i][j]
71             nuevaFigura.append(aux)
72         return Picture(nuevaFigura)
73
74     def horizontalRepeat(self, n):
75         """ Devuelve una nueva figura repitiendo la figura actual al costado
76         | la cantidad de veces que indique el valor de n """
```

```

77     i = 0
78     nuevaFigura = self
79     while i < n-1:
80         nuevaFigura = nuevaFigura.join(self)
81         i+=1
82
83     return Picture(nuevaFigura.img)
84
85     def verticalRepeat(self, n):
86
87         i = 0
88         nuevaFigura = self
89         while i < n-1:
90             nuevaFigura = nuevaFigura.up(self)
91             i+=1
92
93         return Picture(nuevaFigura.img)
94
95     #Extra: Sólo para realmente viciosos
96     def rotate(self):
97         """Devuelve una figura rotada en 90 grados, puede ser en sentido horario
98         o antihorario"""
99         return Picture(None)
100
101

```

- Ejercicio2a (1):



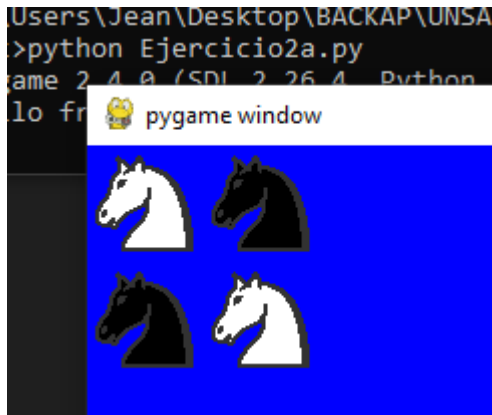
```

Ejercicio2a.py X  Ejercicio2b.py  Ejercicio2c.py  Ejercicio2d.py  Ejercicio2e.py
C: > Users > Jean > Desktop > BACKAP > UNSA universitario > 3er CICLO > (LAB) Programacion web 2 > TEMA
1  from interpreter import draw
2  from chessPictures import *
3
4  draw(knight.up(knight.negative()).join(knight.negative().up(knight)))

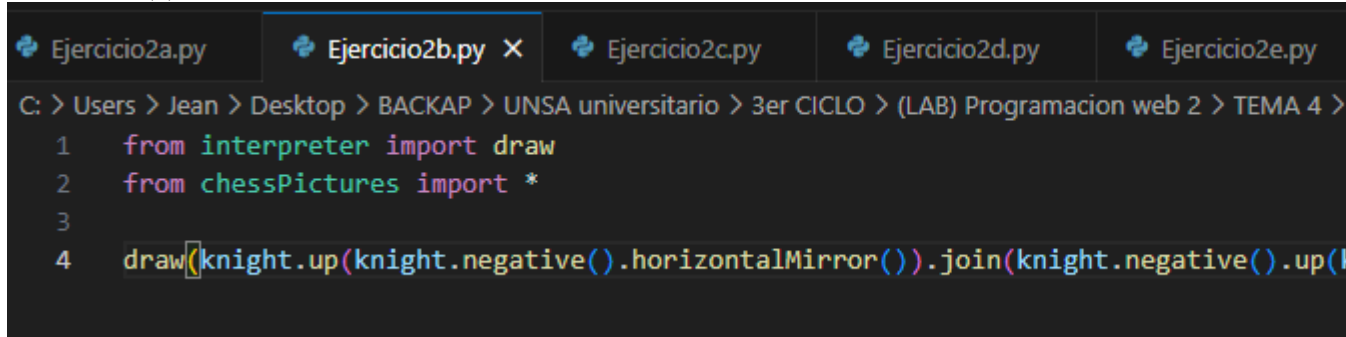
```

Utilizamos la función negative para hallar el color inverso de nuestra pieza, la función up para colocar nuestra pieza arriba de la otra, y la función join para unir estas piezas. Colocamos al caballo blanco encima de nuestro caballo negro, una vez realizado esto, la unimos con la otra columna(caballo negro sobre caballo blanco).

Ejecucion:

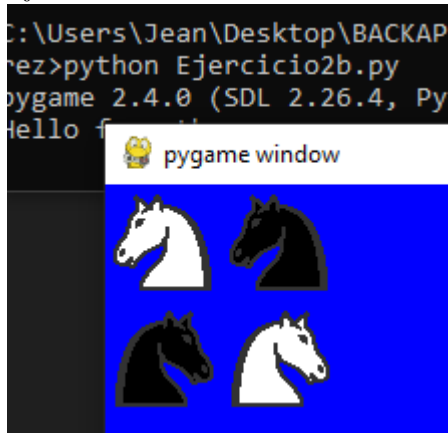


- Ejercicio2b (2):



En este ejercicio añadimos la función `horizontalMirror()`, la cual cambiara de dirección a nuestra pieza, haciendo que mire hacia el otro lado. Al igual que el ejercicio2a (1), creamos nuestros caballos, los colocamos en columnas y los unimos.

Ejecucion:




- Ejercicio2c (3):

```
Ejercicio2a.py Ejercicio2b.py Ejercicio2c.py X
C: > Users > Jean > Desktop > BACKAP > UNSA universitario > 3er C
1 from interpreter import draw
2 from chessPictures import *
3 print(knight)
4 ##draw(knight)
5 draw(queen.horizontalRepeat(4))
```

En este ejercicio creamos la función horizontalRepeat, la cual repetirá nuestra pieza un numero determinado de veces, esta función concatena a las piezas, haciendo que se coloquen una al costado de otra (izquierda a derecha).

Ejecucion:

```
\Users\Jean\Desktop\BACKAP\UNSA univers
z>python Ejercicio2c.py
game 2.4.0 (SDL 2.26.4, Python 3.11.3)
llo
ictu
pygame window
```




- Ejercicio2d (4):

```
Ejercicio2a.py Ejercicio2b.py Ejercicio2c.py Ejercicio2d.py X
C: > Users > Jean > Desktop > BACKAP > UNSA universitario > 3er CICLO > (LAB) Programacio
1 from interpreter import draw
2 from chessPictures import *
3
4 draw((square.join(square.negative())).horizontalRepeat(4))
```

En este ejercicio se nos pide crear una fila de un tablero de ajedrez, por lo que creamos los 2 primeros cuadrados (blanco y negro) y los repetimos 4 veces, para crear una fila con 8 elementos.

Ejecucion:

```
\Users\Jean\Desktop\BACKAP\UNSA universitario\3er CICLO\LAB)
z>python Ejercicio2d.py
game 2.4.0 (SDL 2.26.4, Python 3.11.3)
1
pygame window
```




- Ejercicio2e (5):

```
Ejercicio2c.py Ejercicio2d.py Ejercicio2e.py X Ejercicio2f.py
C: > Users > Jean > Desktop > BACKAP > UNSA universitario > 3er CICLO > (LAB) Programacion web
1 from interpreter import draw
2 from chessPictures import *
3
4 draw((square.negative().join(square)).horizontalRepeat(4))
```

Al igual que el anterior (Ejercicio2d), se nos pide crear una fila de cuadrados, solo que el orden de los cuadrados es diferente (negro y blanco).

Ejecucion:

```
>python Ejercicio2e.py
ame 2.4.0 (SDL 2.26.4, Python 3.11.3)
lo from the pygame community: https://www.pygame.org/contributors
pygame window
```

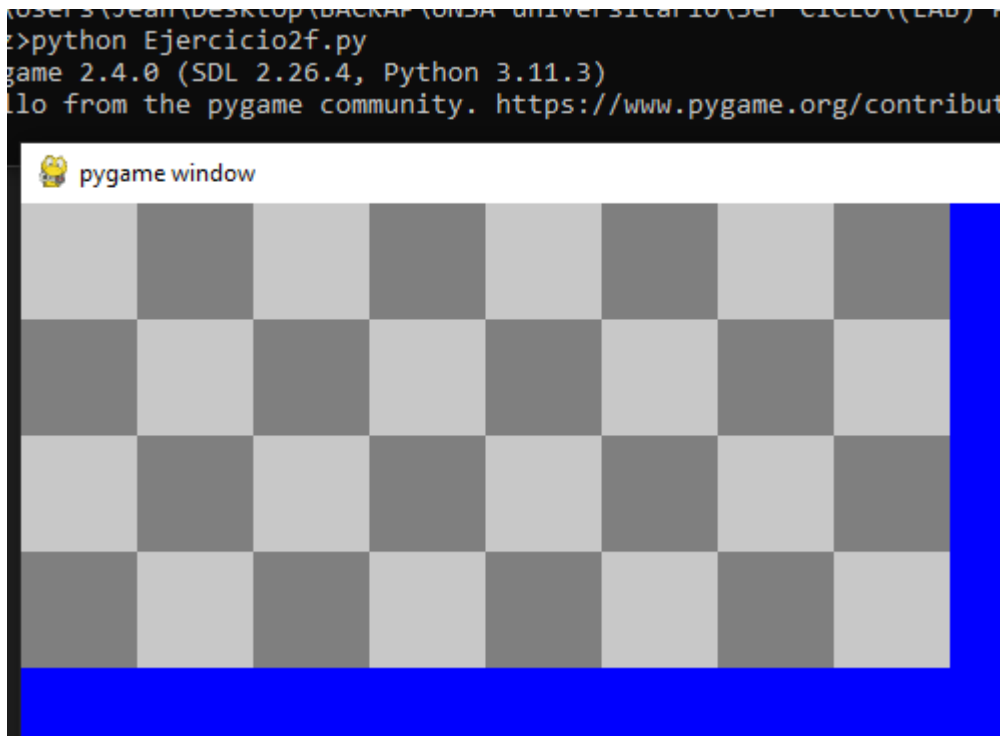


- Ejercicio2f (6):

```
Ejercicio2c.py Ejercicio2d.py Ejercicio2e.py Ejercicio2f.py X
C: > Users > Jean > Desktop > BACKAP > UNSA universitario > 3er CICLO > (LAB) Programacion web
1 from interpreter import draw
2 from chessPictures import *
3
4 fila = (square.join(square.negative())).horizontalRepeat(4)
5 draw((fila.up(fila.negative())).verticalRepeat(2))
```

Se crea la función verticalRepeat, la cual al igual que horizontalRepeat, creara varias veces un elemento o pieza, pero en columna, es decir, repetirá n veces el elemento hacia abajo. En este ejercicio se nos pide crear 4 filas intercaladas de cuadrados, por lo que creamos la variable fila, la cual almacenara una fila intercalada de cuadrados (iniciando con el blanco), luego, lo colocamos encima de nuestra fila con color negativo, utilizando la función verticalRepeat, lo haremos 2 veces para asi crear 4 filas, puesto que repite 2 veces la creación de 2 filas.

Ejecucion:



- Ejercicio2g (7):

```

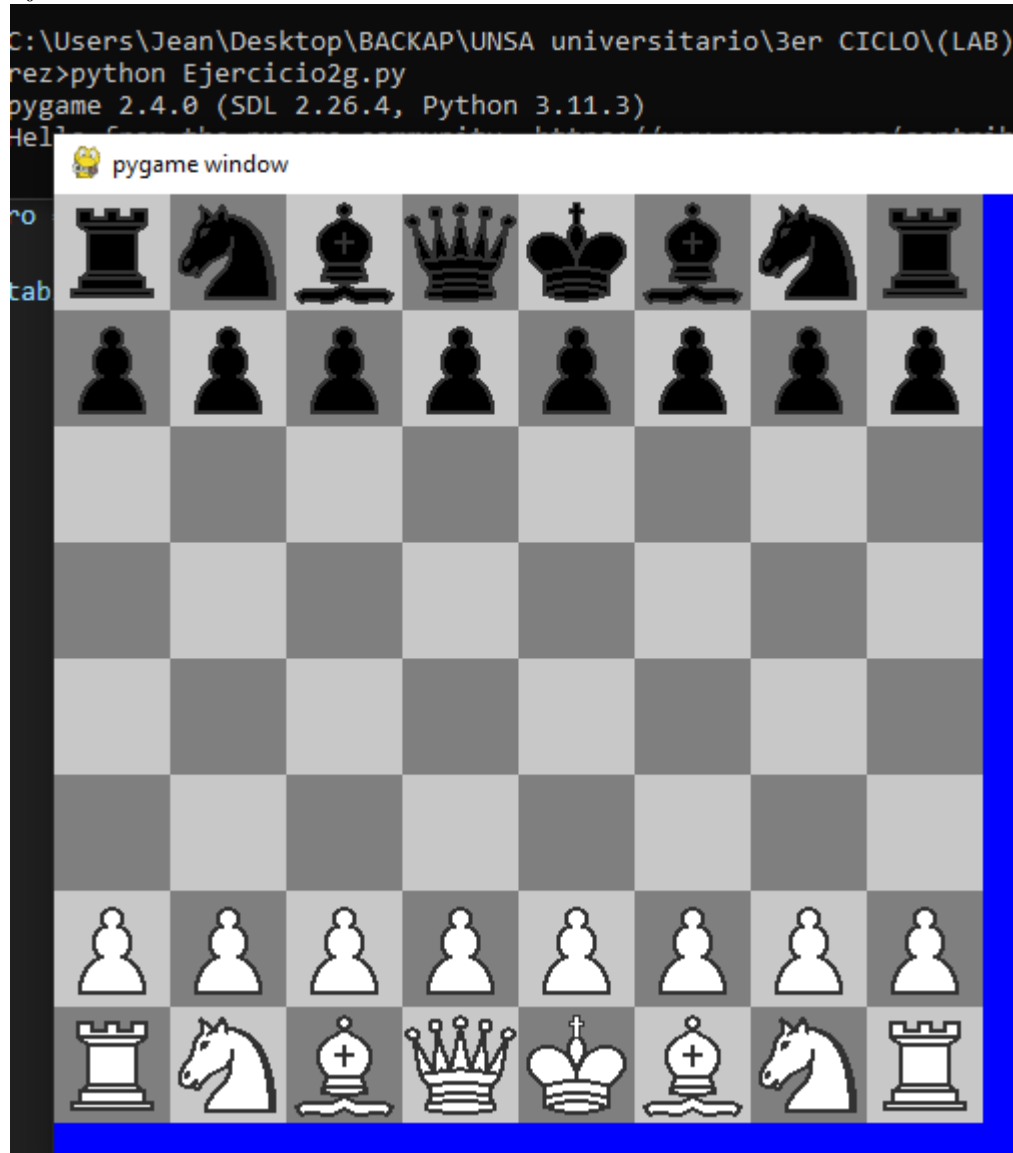
Ejercicio2c.py  Ejercicio2d.py  Ejercicio2e.py  Ejercicio2f.py  Ejercicio2g.py
C: > Users > Jean > Desktop > BACKAP > UNSA universitario > 3er CICLO > (LAB) Programacion web 2 > TEMA 4
1  from interpreter import draw
2  from chessPictures import *
3
4  #corregido
5  torres = square.negative().under(rock).join(square.under(knight)).join(square.under(knight))
6  peones = (square.under(pawn).join(square.negative().under(pawn))).horizontalRepeat(4)
7
8  fila = (square.join(square.negative())).horizontalRepeat(4)
9  espacio = (fila.up(fila.negative())).verticalRepeat(2)
10
11 fichas = (peones.up(torres))
12 fichasN = (torres.negative().up(peones.negative()))
13
14 tablero = (fichasN.up(espacio).up(fichas))
15
16 draw(tablero)

```

En este ejercicio se nos pide crear un tablero de ajedrez. Creamos la variable torres, la cual concatenara con join a todas las piezas en orden de juego, estas piezas se encuentran superpuestas a

los cuadros respectivos, intercalando de color. Seguidamente creamos la variable peones, la cual creara con horizontalRepeat la fila de peones. Creamos la variable fila, la cual almacenara una fila de cuadros, repitiéndose 4 veces para llegar a 8 elementos, luego en la variable espacio, la repetimos/ verticalmente 2 veces. Creamos las variables fichas y fichasN, las cuales almacenaran el orden de piezas para las fichas negras y blancas respectivamente, se utiliza la funcion up y negative para el orden y color. Finalmente, unimos las variables en la variable tablero, la cual colocara primero las fichas negras, luego el espacio y finalmente las fichas blancas.

Ejecucion:



3 Solucion del Cuestionario

- Explique: ¿Para qué sirve el directorio pycache?
- Almacena el código utilizado por Python para almacenar caché, se utiliza con el propósito de

aumentar el rendimiento de los scripts. Cuando se ejecuta un modulo mas de una vez, en lugar de compilar de nuevo, se busca el caché, en caso de encontrarse, se utiliza este mismo para ejecutarse de forma rápida.

4 Conclusiones

- Se utilizo módulos o “librerías” de Python (pygame) para la correcta realización de la práctica. Se ha practicado los temas en Python relacionados a objetos, métodos, funciones, arreglos (listas), ciclos, condicionales, entre otros aspectos básicos con el fin de reforzar nuestro conocimiento del mismo

5 Retroalimentacion general

- REPOSITORIO GITHUB: https://github.com/JeanChara/pweb2_lab04

6 Referencias y bibliografias

- <https://docs.python.org/3/library/array.html?highlight=pop#array.array.pop>
- <https://docs.python.org/3/tutorial/introduction.html#lists>
- <https://docs.python.org/3/glossary.html#term-function>
- <https://docs.python.org/es/3/tutorial/classes.html>