
	UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA	 ABET
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLE-001	Página: 1

## INFORME DE LABORATORIO

### (formato estudiante)

INFORMACIÓN BÁSICA					
<b>ASIGNATURA:</b>	Programacion Web 2				
<b>TÍTULO DE LA PRÁCTICA:</b>	Python				
<b>NÚMERO DE PRÁCTICA:</b>	05	<b>AÑO LECTIVO:</b>	2024-A	<b>NRO. SEMESTRE:</b>	///
<b>FECHA DE PRESENTACIÓN</b>	01/06/2024	<b>HORA DE PRESENTACIÓN</b>	12:30		
<b>INTEGRANTE (s):</b>				<b>NOTA:</b>	
Tapia Huamantuma Antony Juan					
<b>DOCENTE(s):</b> Lino Pinto					

SOLUCIÓN Y RESULTADOS
<p><b>I. SOLUCIÓN DE EJERCICIOS/PROBLEMAS</b></p> <p><i>Chespictures.py:</i></p> <pre style="background-color: #2e3436; color: #eeeeec; padding: 10px;">from piezas import * from picture import *  bishop = Picture(BISHOP) king = Picture(KING) knight = Picture(KNIGHT) pawn = Picture(PAWN) queen = Picture(QUEEN) rock = Picture(ROCK) square = Picture(SQUARE)</pre> <p><i>colors.py</i></p> <pre style="background-color: #2e3436; color: #eeeeec; padding: 10px;">WHITE = (255, 255, 255) BLACK = (0, 0, 0) LIGHTGRAY = (200, 200, 200) GRAY = (127, 127, 127) DARKGRAY = (50, 50, 50) BLUE = (0, 0, 255)  color = {     '_': LIGHTGRAY,</pre>

```
'=': GRAY,  
'.': WHITE,  
'@': BLACK,  
'#': DARKGRAY,  
' ': BLUE,  
}  
inverter = {  
    '_': '=',  
    '=': '_',  
    '.': '@',  
    '@': '.',  
}
```

#### *Interpreter.py*

```
import pygame, sys  
from pygame.locals import *  
from colors import *  
  
def parseLine(DISPLAY, y, s):  
    x = 0  
    for c in s:  
        pygame.draw.line(DISPLAY, color[c], (x, y), (x, y))  
        x += 1  
  
def draw(picture):  
    try:  
        img = picture.img  
    except:  
        img = picture  
    pygame.init()  
  
    DISPLAY=pygame.display.set_mode((640, 480))  
    DISPLAY.fill(BLUE)  
  
    n = len(img)  
    for i in range(0, n):  
        parseLine(DISPLAY, i, img[i])  
  
    while True:  
        for event in pygame.event.get():  
            if event.type==QUIT:  
                pygame.quit()  
                sys.exit()  
            pygame.display.update()
```

#### *picture.py*

```
from colors import *
```

```
class Picture:
    def __init__(self, imagen):
        self.imagen = imagen

    def __eq__(self, otro):
        return self.imagen == otro.imagen

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

    def verticalMirror(self):
        arriba = []
        for valor in self.imagen:
            arriba.append(valor[::-1])
        return Picture(arriba)

    def negative(self):
        Otraimagen = []
        for valor in self.img:
            row = []
            for char in valor:
                row.append(self._invColor(char))
            Otraimagen.append(row)
        return Picture(Otraimagen)

    def join(self, p):
        Otraimagen = []
        for variable, valor in enumerate(self.img):
            Otraimagen.append(list(valor) + list(p.img[variable]))
        return Picture(Otraimagen)

    def up(self, p):
        Otraimagen = []
        for valor in p.img:
            Otraimagen.append(valor[::-1])
        for valor in self.img:
            Otraimagen.append(valor[::-1])
        return Picture(Otraimagen)

    def under(self, p):
        Otraimagen = []
        for valor in self.img:
```

```
def verticalRepeat(self, n):
    auxiliar = self
    for _ in range(n-1):
        auxiliar = auxiliar.up(self)
    return auxiliar
```

SQUARE = [

[illegible]

BISHOP = [

[illegible]

[illegible]

KNIGHT = [





[illegible]

```

"
"
"
"
"
"
"
]
QUEEN = [
"
"
"
"
"
"
"      ##
"      #####
"      #####   ###.###   #####
"      #####   ##...##   #####
"      ##...##   ##...##   ##...##
"      ##...##   #####   ##...##
"      ##      ##...##   #####   ##...##      ##
"      #####   #####   ##      #####   #####
"      ###.###   #####   ####   #####   ###.###
"      ##...##   ##      ####   ##      ##...##
"      ###.###   ##      ####   ##      ##...##
"      #####   ##      ####   ##      #####
"      #####   ####   #####   ####   #####
"      ##      #####   ##.##   #####   ##
"      #####   #####   ##.##   #####   #####
"      #####   #####   ##.##   #####   #####
"      #####   ##.##   ##.##   ##.##   #####
"      #####   ##.##   ###.###   ##.##   #####
"      ##.##   ###.###   ##...##   ###.###   ##.##
"      ##.###   ##...##   ##...##   ##...##   ##.##
"      ##...##   ##...##   ##...##   ##...##   ##...##
"      ##.###   ##...##   ###...###   ##...##   ##.##
"      ###.###   ##...##   ##...##   ##...##   ##...##
"      ##...##   ##...#####...#####...##   ##
"      ##...#####...#####...#####...##
"      ##...#####...#####...#####...##
"      ##...#####...##.###...###.###.##.###...##
"      ###...###.#####.#####.###...##
"      ##.#####...#####.##
"      #####...#####
"      ###.....##
"      ##.....##
"      ##.....##

```

[illegible]

## #Ejercicio a

```
tab = knight
tab = Picture.join(tab, Picture.negative(knight))
tab = Picture.up(Picture.negative(tab), tab)
draw(tab)

#Ejercicio b

tab = knight
tab = Picture.join(tab, Picture.negative(knight))
tab = Picture.up(Picture.verticalMirror(tab), tab)
draw(tab)

#Ejercicio c

tab = queen
tab = Picture.horizontalRepeat(queen, 4)
draw(tab)

#Ejercicio d

tab = square
tab = Picture.join(tab, Picture.negative(square))
tab = Picture.horizontalRepeat(tab, 4)
draw(tab)

#Ejercicio e

tab = square
tab = Picture.join(tab, Picture.negative(square))
tab = Picture.negative(Picture.horizontalRepeat(tab, 4))
draw(tab)

#Ejercicio f

tab1 = square
tab1 = Picture.join(tab1, Picture.negative(square))
tab1 = (Picture.horizontalRepeat(tab1, 4))
tab2 = Picture.negative(tab1)
tab3 = Picture.verticalRepeat(Picture.up(tab2, tab1), 2)
draw(tab3)
```

#Ejercicio g

```
filaInit = Picture.under(rock, Picture.negative(square))

filaInit = Picture.join(filaInit, Picture.under(knight, square))
filaInit = Picture.join(filaInit, Picture.under(bishop, Picture.negative(square)))
filaInit = Picture.join(filaInit, Picture.under(queen, square))
filaInit = Picture.join(filaInit, Picture.under(king, Picture.negative(square)))
filaInit = Picture.join(filaInit, Picture.under(bishop, square))
filaInit = Picture.join(filaInit, Picture.under(knight, Picture.negative(square)))
filaInit = Picture.join(filaInit, Picture.under(rock, square))

filaNegro = Picture.negative(filaInit)

filaPeon = Picture.under(pawn, square)
filaPeon = Picture.join(filaPeon, Picture.under(pawn, Picture.negative(square)))
filaPeon = Picture.horizontalRepeat(filaPeon, 4)

tab1 = square
tab1 = Picture.join(tab1, Picture.negative(square))
tab1 = (Picture.horizontalRepeat(tab1, 4))
tab2 = Picture.negative(tab1)
tab3 = Picture.verticalRepeat(Picture.up(tab2, tab1), 2)

tablero = Picture.up(Picture.negative(filaPeon), filaNegro)
tablero = Picture.up(tab3, tablero)
tablero = Picture.up(filaPeon, tablero)
tablero = Picture.up(filaInit, tablero)

draw(tablero)
```

*Resolucion problemas:*

a.



b.



c.



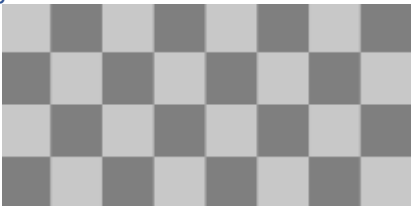
d.



e.





f.



g.





	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;"><b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p><b>Aprobación:</b> 2022/03/01</p>	<p><b>Código:</b> GUIA-PRLE-001</p>	<p><b>Página:</b> 17</p>

## REFERENCIAS Y BIBLIOGRAFÍA

- [https://www.w3schools.com/python/python\\_reference.asp](https://www.w3schools.com/python/python_reference.asp)
- <https://docs.python.org/3/tutorial/>

Contenido y demostración		Puntos		Checklist Estudiante	
				Profesor	
1. GitHub	Repositorio se pudo clonar y se evidencia la estructura adecuada para revisar los entregables. (Se descontará puntos por error u omisión)	4			
2. Commits	Hay porciones de código fuente asociado a los commits planificados con explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	2			
3. Ejecución	Se incluyen comandos para ejecuciones y pruebas del código fuente explicadas gradualmente que permitirían replicar el proyecto. (Se descontará puntos por cada omisión)	2			
4. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	1			
7. Ortografía	El documento no muestra errores ortográficos. (Se descontará puntos por error encontrado)	2			
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente con explicaciones puntuales pero precisas, agregando diagramas generados a partir del código fuente y refleja un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4			
Total		16			

