



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

INFORME DE LABORATORIO

(formato estudiante)

INFORMACIÓN BÁSICA						
ASIGNATURA:	Programación Web 2					
TÍTULO DE LA PRÁCTICA:	Laboratorio 4: Ajedrez - Python					
NÚMERO DE PRÁCTICA:	04	AÑO LECTIVO:	2024-B		NRO. SEMESTRE:	II
FECHA DE PRESENTACIÓN	18/10/2024	HORA DE PRESENTACIÓN				
INTEGRANTE (s) - Choquehuanca Bedoya Brayan Denilson					NOTA (0-20)	Nota colocada por el docente
DOCENTE(s): Lino Pinto Oppe						

RESULTADOS Y PRUEBAS
<p>I. EJERCICIOS RESUELTOS:</p> <p>Clase Picture:</p> <ul style="list-style-type: none"> verticalMirror(): <ol style="list-style-type: none"> Devuelve el espejo vertical de la imagen En este método iteramos sobre la imagen dada como atributo. Crearemos una imagen vacía(lista) y le daremos los valores de la imagen entregada,pero de manera inversa para esto utilizaremos la expresión slice en cada string de la imagen , -1 indicará que comenzará desde el último elemento y el primer: indicará al siguiente de esta manera se invertirán todos los caracteres del string para su posterior agregación a nueva imagen. Este método lo usaremos en el ejercicio 2b

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 2</p>

```
def verticalMirror(self):
    """ Devuelve el espejo vertical de la imagen """
    vertical = []
    for value in self.img:
        vertical.append(value[::-1])

    return Picture(vertical)
```

- **horizontalMirror:**

1. En este método iteramos sobre la imagen dada como atributo
2. Crearemos una imagen vacía(lista) y le daremos los valores de la imagen entregada,pero de manera invertida para iteramos con el método reversed lo que nos dará el último elemento al primero , lo que permitirá que la imagen se muestre de cabeza debido a que la nueva imagen tendrá como primer elemento el último string de la imagen original y así hasta el primer elemento de la imagen original.

```
def horizontalMirror(self):
    """ Devuelve el espejo horizontal de la imagen """
    horizontal = []
    for value in reversed(self.img):
        horizontal.append(value[::-1])
    return Picture(horizontal)
```

- **negative()**

1. Devuelve el negativo de la imagen
2. En este método cambiaremos el color de la imagen dada con su inverso
3. Crearemos una imagen vacía(lista) , luego iteraremos sobre la imagen y agregaremos a la nueva imagen los strings invertidos de la imagen original , esto sera posible gracias a el motodo invColorqueusaeldiccionarioinverterdelacalseColor, este diccionario llave un elemento que puede cambiar de color , solo seran 4 los elementos que tendran inversa invColo rretornara el inverso del caracter y lo agregara a la nueva imagen de esta manera se obtendra el inverso de cada pieza del ajedrez
4. Este método se utiliza mucho en varios ejercicios destacando su uso en el ejercicio 2g.

```
def negative(self):  
    """ Devuelve el negativo de la imagen """  
    nuevaImagen = []  
    for value in self.img:  
        row = []  
        for caracter in value:  
            row.append(self._invColor(caracter))  
        nuevaImagen.append(row)  
    return Picture(nuevaImagen)
```


- **join()**

1. pone la imagen al lado derecho usando el argumento p
2. En este método crearemos una imagen que tendrá a su costado otra imagen ,algo parecido a una concatenación
3. Crearemos una imagen vacía, iteramos sobre la imagen dada ,en este caso nos pasaran dos imágenes pero como el tamaño es el mismo solo usaremos index como índice de la segunda imagen luego simplemente concatenamos los strings de las dos imágenes brindadas osea , el primer string de la primera imagen se concatenan con el primer string de la segunda imagen ,su concatenación será agregada al primer elemento de la nueva imagen y así con todos los elementos, de esta manera se podrá colocar una imagen al costado de otra.
4. Este método se utiliza en algunos otros métodos como horizontalRepeat.

```
def join(self, p):  
    """ pone la imagen al lado derecho usando el argumento p """  
    nuevaImagen = []  
  
    for index, value in enumerate(self.img):  
        nuevaImagen.append(list(value) + list(p.img[index]))  
  
    return Picture(nuevaImagen)
```

- **up()**

1. En este método crearemos una imagen que tendrá encima a otra imagen
2. Crearemos una imagen vacía, iteramos respectivamente con las imágenes dadas, en la iteración de la primera imagen agregaremos los elemento a la imagen vacía creada ,luego iteramos sobre la segunda imagen y seguiremos añadiendo estos elementos a la imagen anteriormente vacía lo que tendremos será una imagen con el doble de longitud de una imagen normal lo que dará la impresión de que una pieza está encima de la otra.

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 4</p>

3. Este método se utiliza en algunos otros métodos como verticalRepeat.

```
def up(self, p):
    """Devuelve a la figura encima de la actual """
    nuevaImagen = []
    #copiar todo p a nuevaImagen
    for value in p.img:
        nuevaImagen.append(value[:,1])

    for value in self.img:
        nuevaImagen.append(value[:,1])
    return Picture(nuevaImagen)
```


- **under():**

1. Devuelve una nueva imagen y la sobrepone encima de la actual
2. En este método crearemos una imagen que sobrepondra otra imagen dando una simulación como un cuadrado del ajedrez con su pieza encima
3. Crearemos una imagen vacía , e iteramos sobre la imagen que tiene la figura de una pieza, luego haremos una iteración anidada para consultar todos los caracteres de la nueva imagen dentro del bucle anidado utilizaremos una condicional para saber que caracteres de la nueva imagen se encuentran vacíos (" ") ,y caso que esté vacío colocaremos el carácter de la otra imagen que será un square de esta manera se colocara la pieza encima de su cuadrado de ajedrez
4. Este método se utilizará en el último ejercicio 2g.

```
def under(self, p):
    """ Devuelve una nueva y la pone encima de la actual"""
    nuevaImagen = []
    for value in self.img:
        nuevaImagen.append(list(value))

    for i, value in enumerate(p.img):
        for j, caracter in enumerate(value):
            if(nuevaImagen[i][j] == ' '):
                nuevaImagen[i][j] = caracter
```

- **horizontalRepeat():**

	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLE-001	Página: 5


1. Devuelve una nueva figura repitiendo la figura actual al costado la cantidad de veces que indique el valor de n
2. En este método crearemos una imagen que tendrá “n” veces a la misma imagen a su lado
3. Crearemos una imagen que tendrá lo mismo que la imagen principal sobre todo para que nuestra imagen original no quede modificada , luego iteramos n-1 veces debido a que utilizaremos el método join dentro de este bucle , por lo que el trabajo de duplicar n veces la imagen lo hará join ,sin embargo como en la primera iteración genera 2 imagenes se tendrá que iterar n-1 veces , de esta manera colocaremos n veces la imagen al costado de la misma imagen
4. Este método se utilizará en el ejercicio 2c.

```
def horizontalRepeat(self, n):
    """ Devuelve una nueva figura repitiendo la figura actual al costado
        la cantidad de veces que indique el valor de n """
    aux = self
    for _ in range(n-1):
        aux = aux.join(self)
    return aux
```

- **verticalRepeat():**

1. Devuelve una nueva figura repitiendo la figura actual debajo, la cantidad de veces que indique el valor de n
2. En este método crearemos una imagen que tendrá n veces a la misma imagen encima ^ Este método se comportara de la misma manera que en anterior método , con la diferencia que dentro de las iteraciones se llamará al método up en vez del “join” lo que permitirá superponer n veces la imagen .
3. Este método se utiliza en varios ejercicios.

```
def verticalRepeat(self, n):
    """Devuelve una nueva figura repitiendo la figura actual debajo, la cantidad
    de veces que indique el valor de n"""
    aux = self
    for _ in range(n-1):
        aux = aux.up(self)
    return aux
```

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 6</p>

- **Ejercicio2a.py**

```
from chessPictures import *
from interpreter import draw


tab = knight
#llama a la picture creando la primera columna
tab = Picture.join(tab, Picture.negative(knight))
#llama ala segunda columna
draw(Picture.up(Picture.negative(tab), tab))
```



- **Ejercicio2b.py**

```
from chessPictures import *
from interpreter import draw


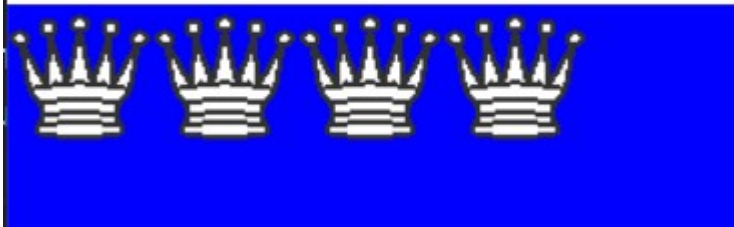
tab = knight
tab = Picture.join(tab, Picture.negative(knight))
tab = Picture.up(Picture.verticalMirror(tab), tab)
draw(tab)
```

 pygame window

- **Ejercicio2c.py**

```
from interpreter import draw
from chessPictures import *

draw(king.horizontalRepeat(4))
```



 pygame window

- **Ejercicio2d.py**

```
from interpreter import draw
from chessPictures import *

eb = Picture(SQUARE)
en = eb.negativo()

draw(eb.join(en).horizontalRepeat(4))
```

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 8</p>



- **Ejercicio2e.py**

```
from interpreter import draw
from chessPictures import *

eb = Picture(SQUARE)
en = eb.negative()



draw(en.join(eb).horizontalRepeat(4))
```

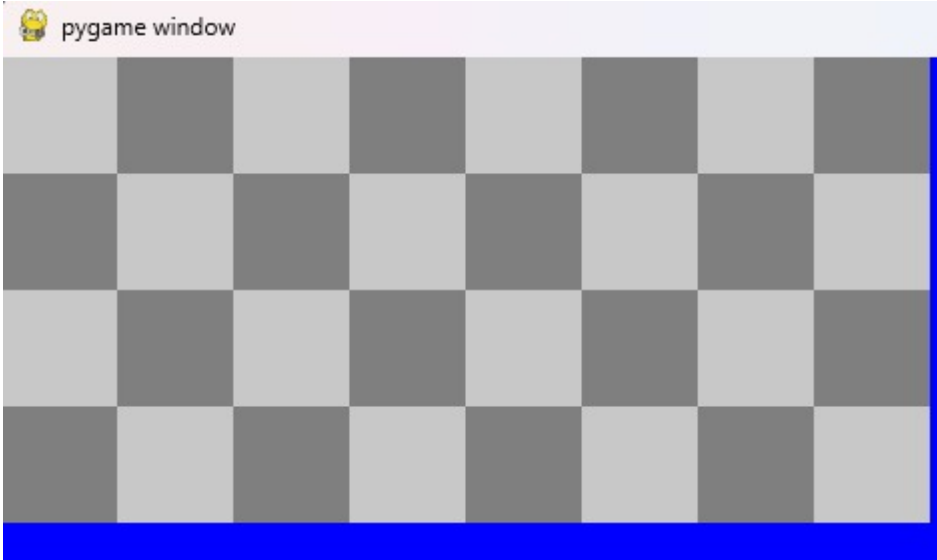


- **Ejercicio2f.py**

```
from interpreter import draw
from chessPictures import *

eb = Picture(SQUARE)
en = eb.negative()
fila1 = eb.join(en).horizontalRepeat(4)
fila2 = en.join(eb).horizontalRepeat(4)
imagen = fila2.up(fila1).verticalRepeat(2)
draw(imagen)
```


	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 9</p>



- **Ejercicio2g.py**

```
from chessPictures import *
from interpreter import draw
from picture import Picture
from colors import *

#Ejercicio g tableor completo

filaInicial = Picture.under(rock, Picture.negative(square))

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

filaNegro = Picture.negative(filaInicial)


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

```
filaPeon = Picture.horizontalRepeat(filaPeon, 4)

tabla1 = square
tabla1 = Picture.join(tabla1, Picture.negative(square))
tabla1 = (Picture.horizontalRepeat(tabla1, 4))
tabla2 = Picture.negative(tabla1)
tabla3 = Picture.verticalRepeat(Picture.up(tabla2, tabla1), 2)

tablero = Picture.up(Picture.negative(filaPeon), filaNegro)
tablero = Picture.up(tabla3, tablero)
tablero = Picture.up(filaPeon, tablero)
tablero = Picture.up(filaInicial, tablero)

draw(tablero)
```

 pygame window



• z

CONCLUSIONES

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 12</p>

METODOLOGÍA DE TRABAJO

METODOLOGÍA DE TRABAJO

1. **Definición del Problema:** Se identificaron los objetivos y metas de la práctica.
2. **Investigación:** Se revisaron materiales y documentación relevante.
3. **Planificación:** Se elaboró un plan de trabajo con tareas y plazos asignados.
4. **Desarrollo:** Se implementó el código, realizando pruebas unitarias y documentando el proceso.
5. **Pruebas:** Se llevaron a cabo pruebas exhaustivas para asegurar el funcionamiento correcto.
6. **Evaluación:** Se revisó el trabajo en grupo y se solicitó retroalimentación.
7. **Presentación:** Se preparó el informe final y se presentaron los resultados.

REFERENCIAS Y BIBLIOGRAFÍA

Guides.github.com. 2021. GitHub Guides. [online] Available at: [Accessed 10 April 2021].

RÚBRICA PARA EL CONTENIDO DEL INFORME Y DEMOSTRACIÓN

El alumno debe marcar o dejar en blanco en celdas de la columna Checklist si cumplió con el ítem correspondiente.

Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos los ítems.

El alumno debe autocalificarse en la columna Estudiante de acuerdo a la siguiente tabla:

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	x	x	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4		x	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2		x	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2		x	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2		x	
6. Fechas	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	2		x	
7. Ortografía	El documento no muestra errores ortográficos.	2		x	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4			
TOTAL		20		18	

Nivel

Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 14</p>