

Tarea 4

EL JUEGO DE LA VIDA

Estructura de Datos
Universidad Nacional Autónoma de México
Facultad de Estudios Superiores Aragón

REALIZADO POR:

**Enrique Emiliano Cano
García**

Clase Array2D.

```
1 class Array2D: 3 usages  ± kno4
2     def __init__(self, ren, col):  ± kno4
3         self._data = [[None] * col for _ in range(ren)]
4         self._ren = ren
5         self._cols = col
6
7     def clear(self, dato): 2 usages  ± kno4
8         for row in range(self._ren):
9             for col in range(self._cols):
10                 self._data[row][col] = dato
11
12     def get_ren_size(self):  ± kno4
13         return self._ren
14
15     def get_col_size(self):  ± kno4
16         return self._cols
17
18     def set_item(self, ren, col, dato): 5 usages  ± kno4
19         if 0 <= ren < self._ren and 0 <= col < self._cols:
20             self._data[ren][col] = dato
21         else:
22             raise IndexError("Índice fuera de rango")
23
24     def get_item(self, ren, col): 2 usages  ± kno4
25         if 0 <= ren < self._ren and 0 <= col < self._cols:
26             return self._data[ren][col]
27         else:
28             raise IndexError("Índice fuera de rango")
29
30     def to_string(self): 1 usage  ± kno4
31         result = ""
32         for row in self._data:
33             result += " ".join(map(str, row)) + "\n"
34         return result
35
```

Juego de la vida

```
1 from Array2D import Array2D
2
3 class JuegoDeLaVida:
4     def __init__(self, rens, cols):
5         self.rejilla = Array2D(rens, cols)
6         self.rejilla.clear(0)
7         self.rens = rens
8         self.cols = cols
9
10    def configurar(self, lista_celulas_vivas):
11        """
12        lista_celulas_vivas debe ser una lista de tuplas [(ren, col), (ren, col), ...]
13        que especifican las posiciones de las células vivas.
14        """
15        for ren, col in lista_celulas_vivas:
16            self.rejilla.set_item(ren, col, 1) # Establece la célula como viva (1)
17
18    def get_num_vecinos_vivos(self, ren, col):
19        vecinos_vivos = 0
20        for i in range(ren - 1, ren + 2):
21            for j in range(col - 1, col + 2):
22                if (i == ren and j == col) or i < 0 or j < 0 or i >= self.rens or j >= self.cols:
23                    continue
24                if self.rejilla.get_item(i, j) == 1:
25                    vecinos_vivos += 1
26        return vecinos_vivos
27
28    def actualizar_generacion(self):
29        nueva_grid = Array2D(self.rens, self.cols)
30        nueva_grid.clear(0) # Inicializamos la nueva rejilla con todas las células muertas
31
32        for ren in range(self.rens):
33            for col in range(self.cols):
34                vecinos_vivos = self.get_num_vecinos_vivos(ren, col)
35                estado_actual = self.rejilla.get_item(ren, col)
36
37                # Aplicación de las reglas del juego
38                if estado_actual == 1: # Célula viva
39                    if vecinos_vivos == 2 or vecinos_vivos == 3:
40                        nueva_grid.set_item(ren, col, 1) # Sobrevive
41                    else:
42                        nueva_grid.set_item(ren, col, 0) # Muere
43                else: # Célula muerta
44                    if vecinos_vivos == 3:
45                        nueva_grid.set_item(ren, col, 1) # Nace una nueva célula
46                    else:
47                        nueva_grid.set_item(ren, col, 0) # Permanece muerta
48
49        # Actualizamos la rejilla con la nueva generación
50        self.rejilla = nueva_grid
51
52    def jugar(self, generaciones):
53        """
54        Juega el juego durante un número de generaciones.
55        """
56        for gen in range(generaciones):
57            print(f"Generación {gen + 1}:")
58            print(self.rejilla.to_string()) # Imprime el estado actual de la rejilla
59            self.actualizar_generacion()
60
```

Main

```
61 if __name__ == "__main__":
62     # Inicializar el juego con una rejilla de 5x5
63     juego = JuegoDeLaVida(10, 10)
64
65     celulas_vivas = [(2, 1), (2, 2), (2, 3), (3, 4), (8, 4), (7, 4), (3, 5)]
66     juego.configurar(celulas_vivas)
67
68     # Jugar durante 5 generaciones
69     juego.jugar(10)
70
```

Ejecución

Generación 1:

```
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 1 1 1 0 0 0 0 0 0
0 0 0 0 1 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

Generación 2:

```
0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0
0 0 1 1 1 0 0 0 0 0
0 0 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

Generación 3:

```
0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0
0 1 0 0 1 0 0 0 0 0
0 0 1 0 1 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

Generación 4:

```
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 1 1 0 0 0 0 0 0 0
0 0 1 0 1 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

Generación 5:

```
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 1 1 1 0 0 0 0 0 0
0 1 1 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

Generación 6:

```
0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0
0 1 0 1 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

[illegible][illegible]

A 10x10 grid of 100 small orange circles on a black background. The circles are arranged in 10 rows and 10 columns, with a small gap between each circle. The circles are a vibrant orange color and have a slightly textured appearance. The background is a solid black. The grid is centered in the image.

A 10x10 grid of 100 small, identical, stylized human figures standing in a uniform pattern. Each figure is a simple, dark silhouette with a distinct head, torso, and legs, standing on a small base. They are arranged in a perfectly aligned grid, with 10 figures in each row and 10 figures in each column. The figures are dark in color, contrasting with the light background.