

Práctica 10: Listas multidimensionales

1. Objetivos y planificación
2. Abstracción
3. Crear baraja
4. Contar parejas
5. Crear columnas
6. Mostrar columnas
7. Mover cartas
8. Trabajo en casa

Índice

1. Objetivos y planificación
2. Abstracción
3. Crear baraja
4. Contar parejas
5. Crear columnas
6. Mostrar columnas
7. Mover cartas
8. Trabajo en casa

Objetivos y planificación

Objetivos

- Abstracciones con listas paralelas.
- Abstracciones con listas multidimensionales.
- Construir, recorrer y manipular estructuras anidadas.

Planificación

Desarrollaremos:

- Un sencillo juego de solitario con cartas

Juego

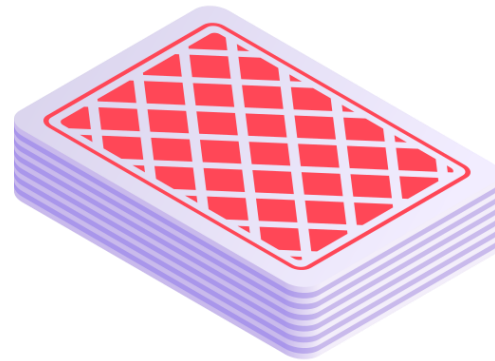
Disponemos de una baraja con cartas de poker:

- Las cartas tendrán palo, número y color.



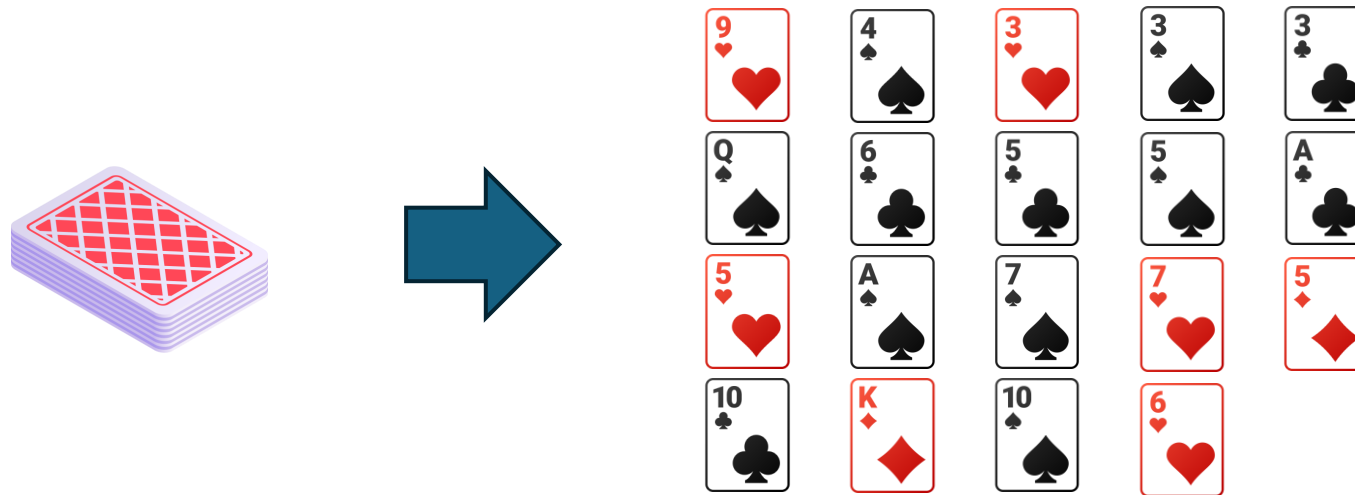
Juego

Al inicio del juego se creará un mazo de cartas barajadas aleatoriamente.



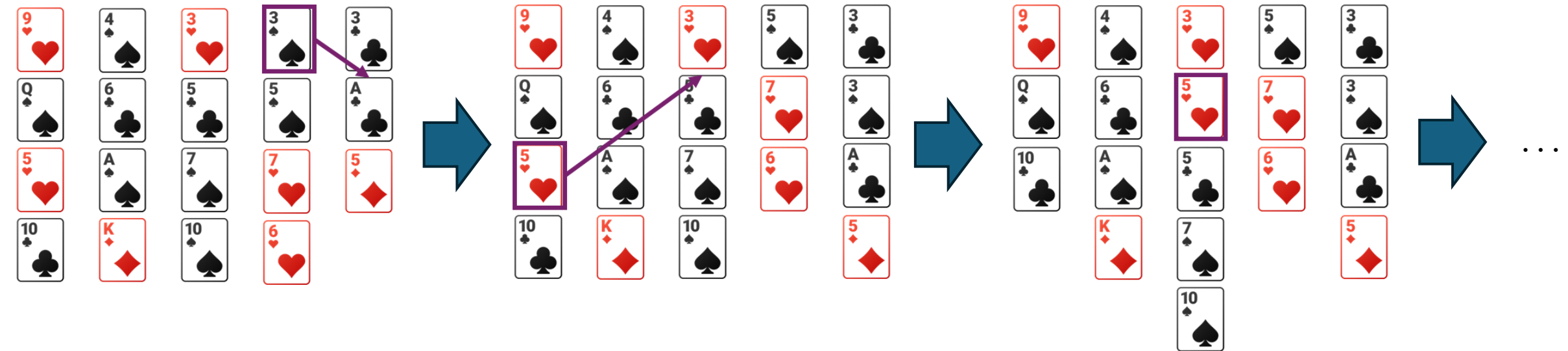
Juego

A continuación, dividiremos el mazo en “n” columnas de cartas que serán visibles para el jugador.



Juego

El jugador tendrá 3 movimientos para recolocar cartas como prefiera.

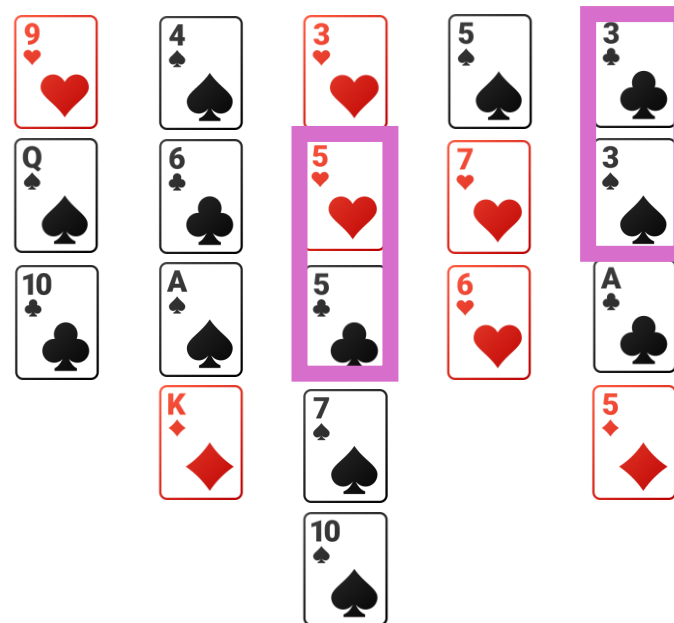


Juego

Tras finalizar los movimientos, se calculará el nº de parejas (mismo número) de cada columna.

- Más adelante podremos añadir más criterios de puntuación (cartas del mismo color, mismo palo, escaleras, etc...)

El objetivo es utilizar los 3 movimientos para maximizar la puntuación.



Juego

Descarga del campus virtual el ejecutable del juego y
pruébalo.

Planificación

- Crearemos funciones en el siguiente orden:
 1. Crear la baraja.
 2. Contar parejas en una lista.
 3. Crear las columnas a partir de la baraja.
 4. Mostrar las columnas.
 5. Mover cartas.
- Lo integraremos todo en un bucle de juego.

1. Objetivos y planificación
2. Abstracción
3. Crear baraja
4. Contar parejas
5. Crear columnas
6. Mostrar columnas
7. Mover cartas
8. Trabajo en casa

Abstracción

Abstracciones

Necesitamos representar tres conceptos:

- Los tipos de cartas disponibles en el juego y su información
 - Nombre, número, color, palo
- El mazo de juego barajado aleatoriamente.
- las columnas del tablero una vez repartidos.

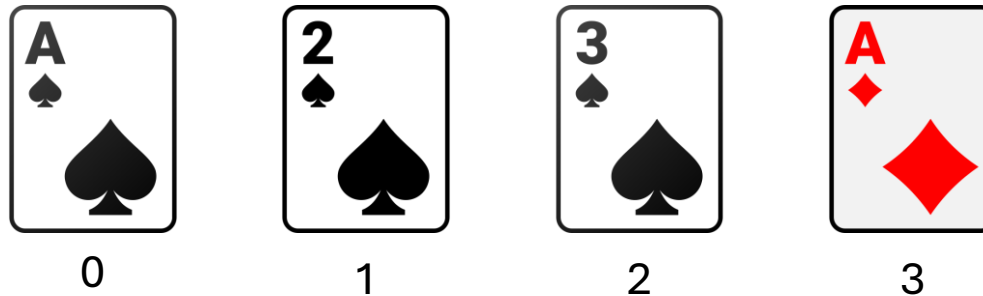
Información de las cartas

Utilizaremos tuplas paralelas para representar los tipos de cartas disponibles.

- Una tupla con los ids (nombres) de las cartas.
- Una tupla con los números.
- Una tupla con los palos.
- Una tupla con los colores.

Información de las cartas

Los índices de todas las tuplas están relacionados.



```
card_names = ("1_of_spades", "2_of_spades", "3_of_spades", "1_of_diamonds", ...)
```

```
card_numbers = (1, 2, 3, 1, ...)
```

```
card_suits = ("spades", "spades", "spades", "diamonds", ...)
```

```
card_colors = ("black", "black", "black", "red", ...)
```


Información de las cartas

Los índices de todas las tuplas están relacionados.

- Para obtener el nombre, número, palo o color de una carta, accedemos al mismo índice en cada tupla. Puedes descargar el fichero `solitaire_cards.py` con todas las tuplas ya definidas.

```
card_names = ("1_of_spades", "2_of_spades", "3_of_spades", "1_of_diamonds", ...)
card_numbers = (1, 2, 3, 1, ...)
card_suits = ("spades", "spades", "spades", "diamonds", ...)
card_colors = ("black", "black", "black", "red", ...)
```



Información de las cartas

¿Por qué tuplas y no listas?

La información de las cartas no cambia durante la ejecución.

Dado que las listas son mutables y las tuplas son inmutables, es mejor usar tuplas cuando los valores, el orden y el tamaño no se van a modificar.

Mazo de juego barajado

Para representar el mazo de juego, utilizaremos una nueva lista con los nombres de las cartas

- Igual que en la sesión anterior.

```
shuffled_deck = ["3_of_spades", "2_of_diamonds", "4_of_clubs", "1_of_hearts", ...]
```

Mazo de juego barajado

¿Necesitamos modificar el orden de las listas paralelas para que coincidan con el orden del mazo?

```
shuffled_deck = ["3_of_spades", "2_of_diamonds", "4_of_clubs", "1_of_hearts", ...]
```

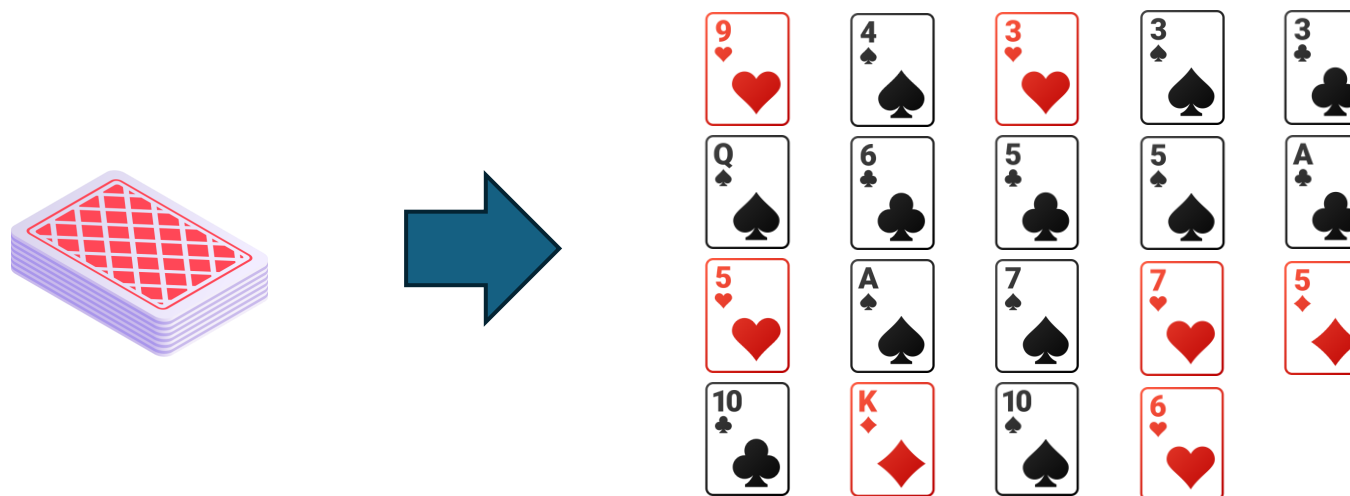
Mazo de juego barajado

Gracias a las listas paralelas, durante el juego podemos trabajar solo con listas de nombres:

- Las 4 listas paralelas no se modifican, permanecen en el mismo orden.
- Cuando necesitemos saber el número, palo, o color de una carta del mazo:
 1. Usamos el método `index` de la lista paralela `card_names` pasándole el nombre de la carta del mazo a consultar. Nos devolverá su índice.
 2. Usamos ese índice para acceder al resto de listas paralelas.

Columnas de cartas

Partiendo del mazo de cartas barajado, lo dividiremos para crear las columnas de cartas.

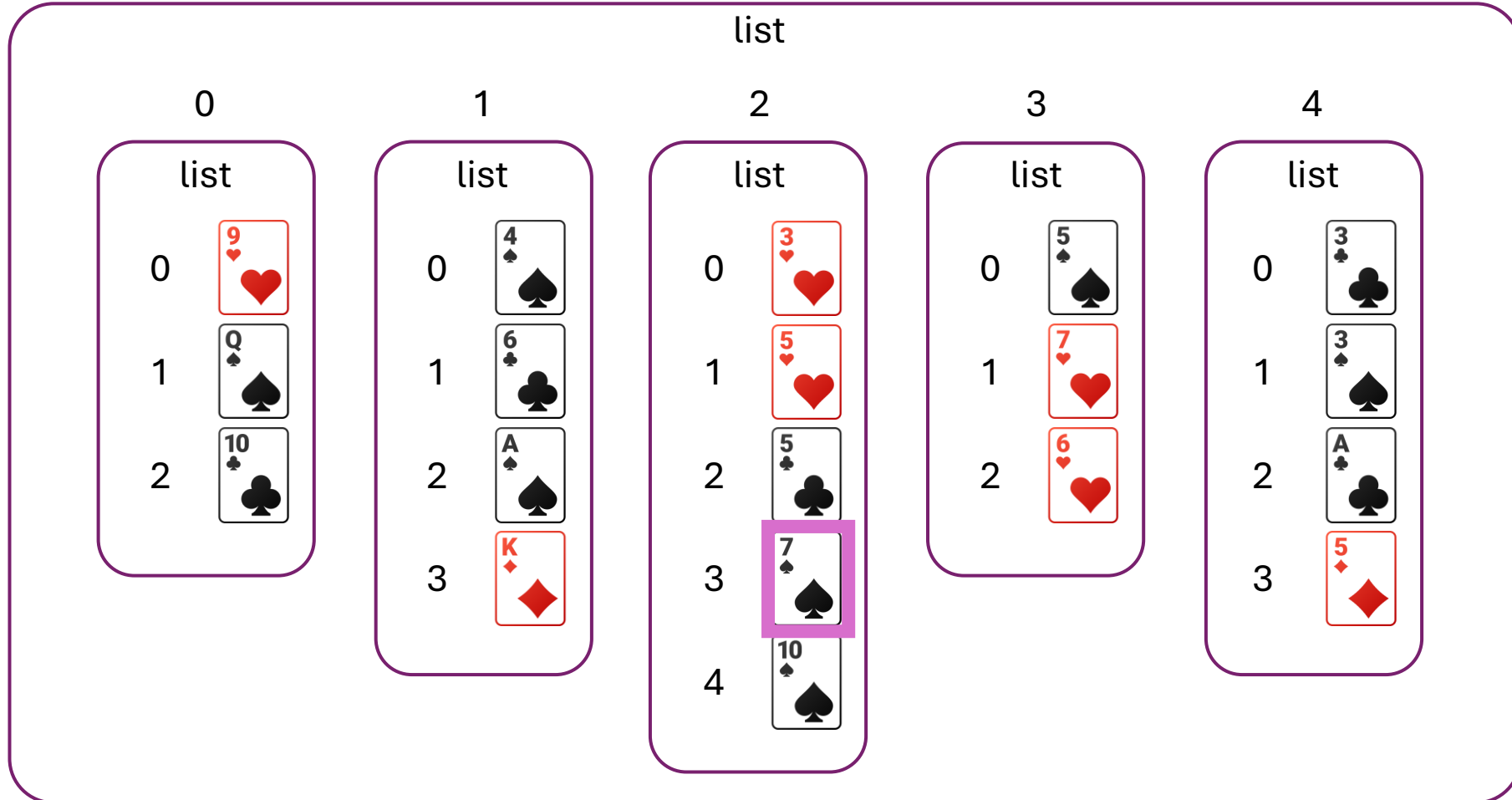


Columnas de cartas

Para representar las columnas, usaremos una lista multidimensional (una lista de listas).

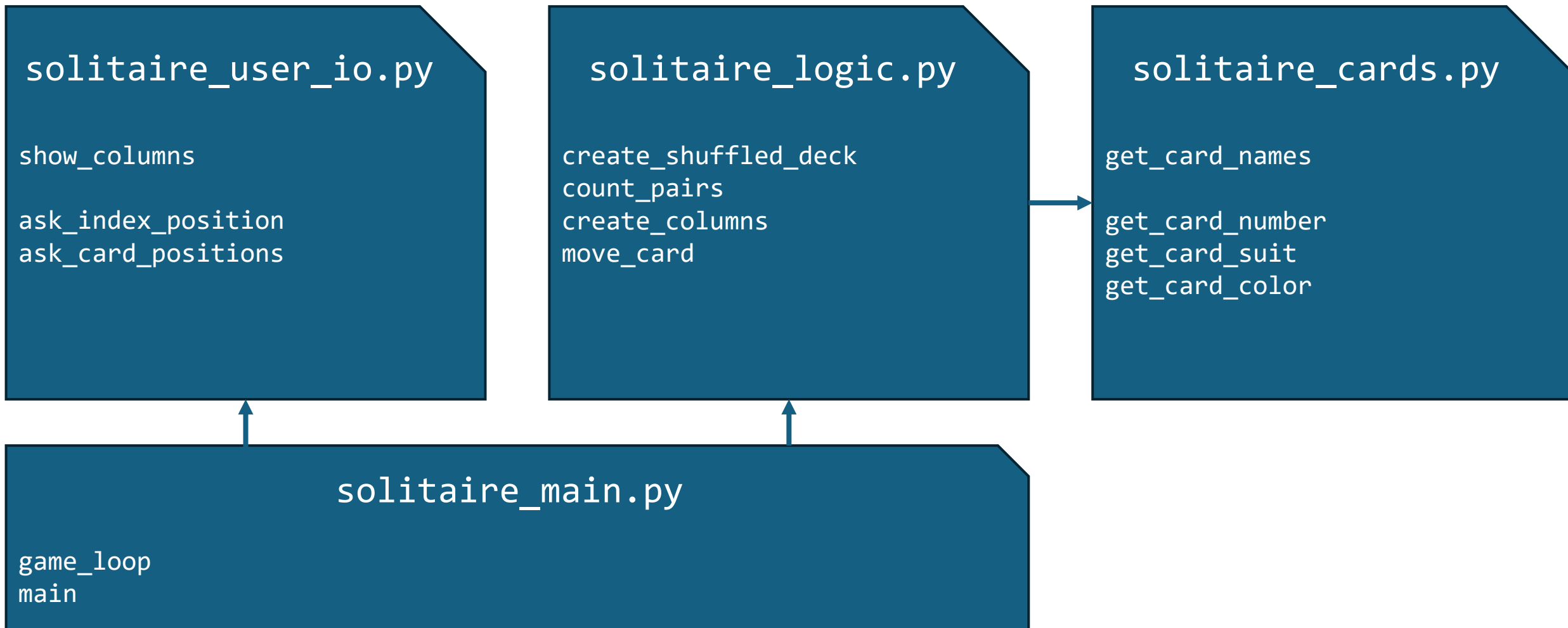
- Cada columna será una lista con nombres de cartas.
- El tablero será una lista de listas (columnas).

Columnas de cartas



Índice: [2][3]

Organización del código



Descarga código inicial

Descarga el código inicial del campus virtual. Incluye:

- `solitaire_cards.py`: Tuplas con las cartas de juego.
- `solitaire_user_io.py`: Funciones para imprimir el tablero de juego.
- `test_examples.py`: Ejemplos que podrás usar en los tests.

1. Objetivos y planificación
2. Abstracción
3. Crear baraja
4. Contar parejas
5. Crear columnas
6. Mostrar columnas
7. Mover cartas
8. Trabajo en casa

Crear baraja

Crear baraja

En el módulo `solitaire_cards.py` crea la función `create_shuffled_deck`.

- Recibe la tupla con los nombres de cartas de juego.
 - La tupla es inmutable. Necesitas crear una lista a partir de ella. Puedes usar la función `list()`.
- Devuelve una nueva lista con los nombres ordenados de forma aleatoria. Puedes importar el módulo `random` y usar su función `shuffle`.

1. Objetivos y planificación
2. Abstracción
3. Crear baraja
4. Contar parejas
5. Crear columnas
6. Mostrar columnas
7. Mover cartas
8. Trabajo en casa

Contar parejas

Contar parejas

Crea el módulo `solitaire_logic.py` y en él crea la función `count_pairs`:

- Consideraremos como pareja dos cartas en orden consecutivo que tengan el mismo número.

Contar parejas

La lista de cartas que pasaremos a la función solo tendrá nombres de cartas.

- ¿Cómo podemos saber el número de una carta?
- ¿Qué otros parámetros deberíamos pasarle?

Una vez lo tengas, claro, crea la función y pruébala con doctests.

Acceso a información de cartas

Si más adelante queremos crear más funciones que accedan a los números, palos o colores de las cartas, deberemos repetir parte del código que ya hemos creado en la función `count_pairs`.

Vamos a crear funciones para acceder a la información más fácilmente.

Acceso a información de cartas

En el modulo `solitaire_cards.py` crea las funciones `get_card_number`, `get_card_suit` y `get_card_color`.

- Reciben el nombre de la carta.
- Devuelven la característica correspondiente.

Crea la función `get_card_names` que devuelva una **copia** de la lista de nombres.

Puedes acceder a las listas paralelas como constantes, sin necesidad de pasarlas como parámetros.

Contar parejas

Importa el modulo `solitaire_cards.py` en el modulo `solitaire_logic.py` y

- Modifica la función `count_pairs` para que utilice la función `get_card_number`
- Modifica la función `create_shuffled_deck` para que obtenga la lista de nombres a través de la función `get_card_names` en vez de recibirla como parámetro.

1. Objetivos y planificación
2. Abstracción
3. Crear baraja
4. Contar parejas
5. **Crear columnas**
6. Mostrar columnas
7. Mover cartas
8. Trabajo en casa

Crear columnas

Crear columnas

En el modulo `solitaire_logic.py` crea la función `create_columns`

- Recibe el mazo barajado y el número de columnas.
- Devuelve una nueva lista de listas con las columnas.

Crear columnas

Consejos:

- Crea una lista vacía para el tablero.
- Obtén el tamaño de las columnas (el último podrá tener menos cartas).
- Crea un bucle:
 - Utiliza el operador de slicing con el número de la iteración actual y el tamaño de las columnas para encontrar las posiciones de inicio y final del corte.
 - Inserta la lista obtenida (la columna) en la lista del tablero.
 - Con cada iteración, las posiciones de inicio y fin para el corte deben cambiar.

1. Objetivos y planificación
2. Abstracción
3. Crear baraja
4. Contar parejas
5. Crear columnas
6. **Mostrar columnas**
7. Mover cartas
8. Trabajo en casa

Mostrar columnas

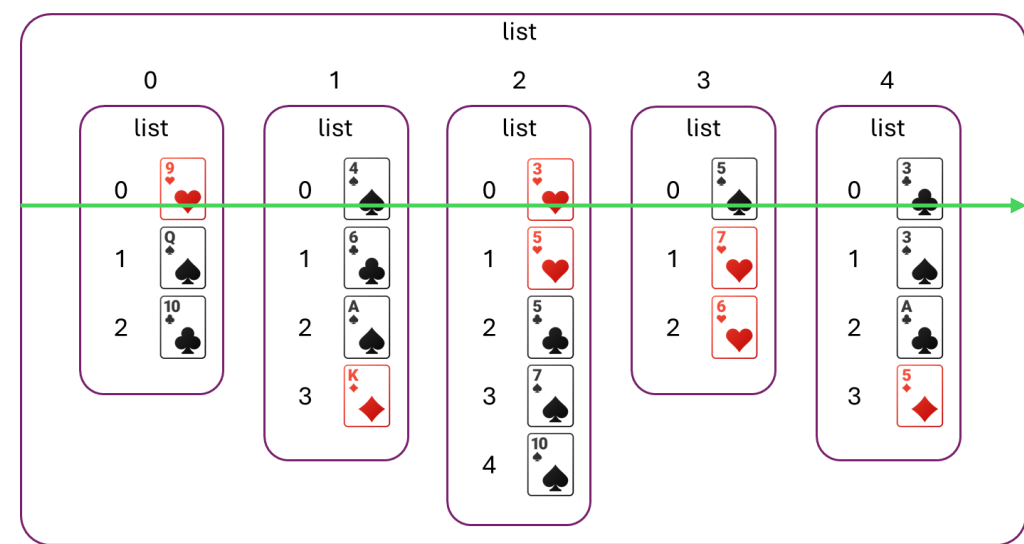
Mostrar columnas

El modulo `solitaire_user_io.py`. contiene la función `show_columns`:

- Recibe la lista de listas de columnas.
- Muestra en consola las cartas en el siguiente formato:
 - Cada carta sigue el formato [índice de la columna][índice de la carta] nombre.

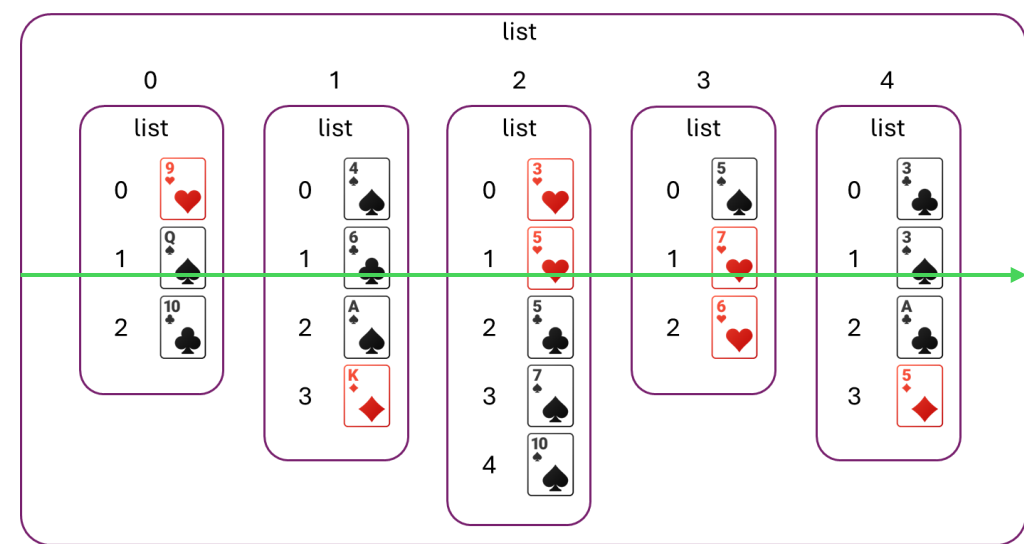
```
[0][0] 9_of_hearts [1][0] 4_of_spades [2][0] 3_of_hearts [3][0] 5_of_spades [4][0] 3_of_clubs
[0][1] queen_of_spades [1][1] 6_of_clubs [2][1] 5_of_hearts [3][1] 7_of_hearts [4][1] 3_of_spades
[0][2] 10_of_clubs [1][2] ace_of_spades [2][2] 5_of_clubs [3][2] 6_of_hearts [4][2] ace_of_clubs
[1][3] king_of_diamonds [2][3] 7_of_spades [4][3] 5_of_diamonds
[2][4] 10_of_spades
```

Funcionamiento de show_columns



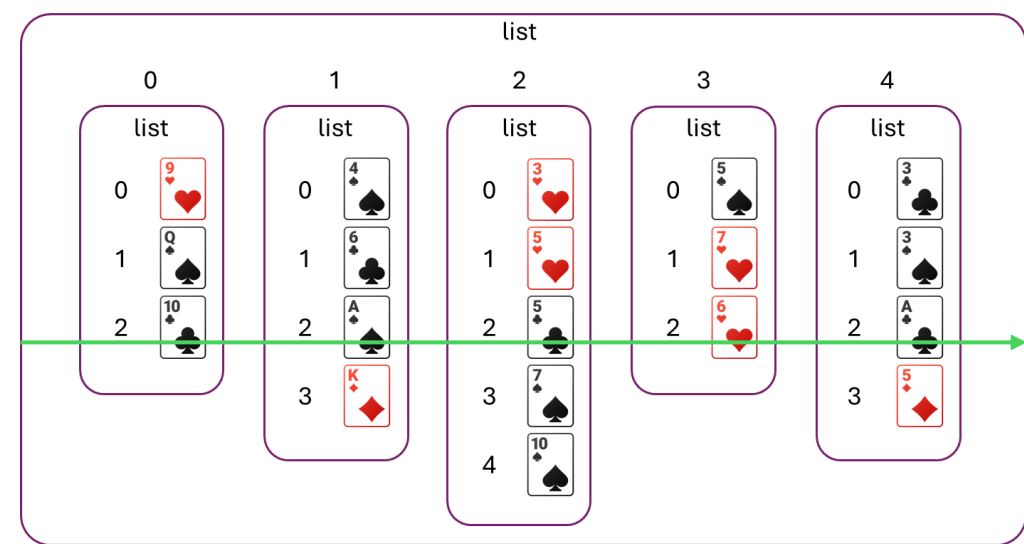
```
[0][0] 9_of_hearts [1][0] 4_of_spades [2][0] 3_of_hearts [3][0] 5_of_spades [4][0] 3_of_clubs
```


Funcionamiento de show_columns



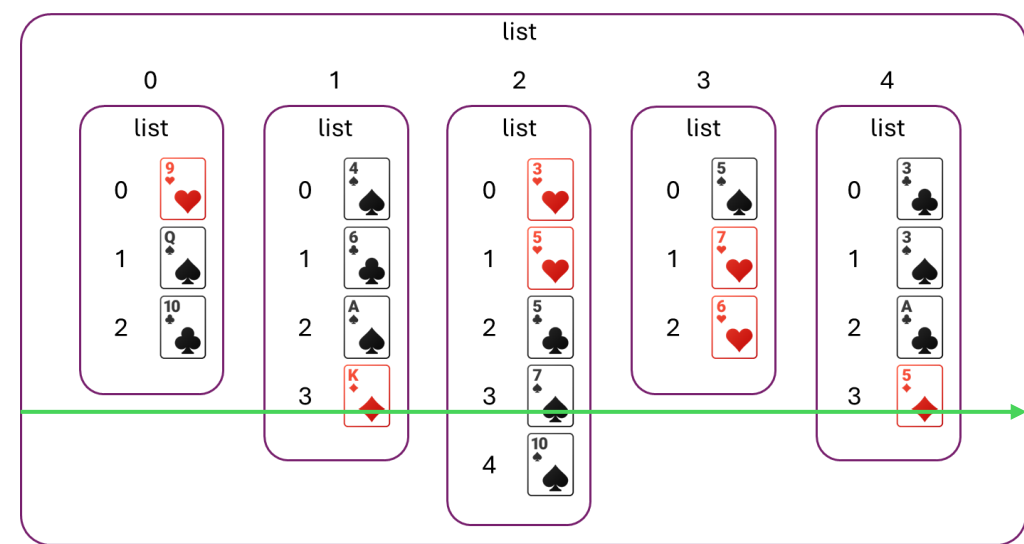
```
[0][0] 9_of_hearts    [1][0] 4_of_spades    [2][0] 3_of_hearts    [3][0] 5_of_spades    [4][0] 3_of_clubs  
[0][1] queen_of_spades [1][1] 6_of_clubs    [2][1] 5_of_hearts    [3][1] 7_of_hearts    [4][1] 3_of_spades
```

Funcionamiento de show_columns



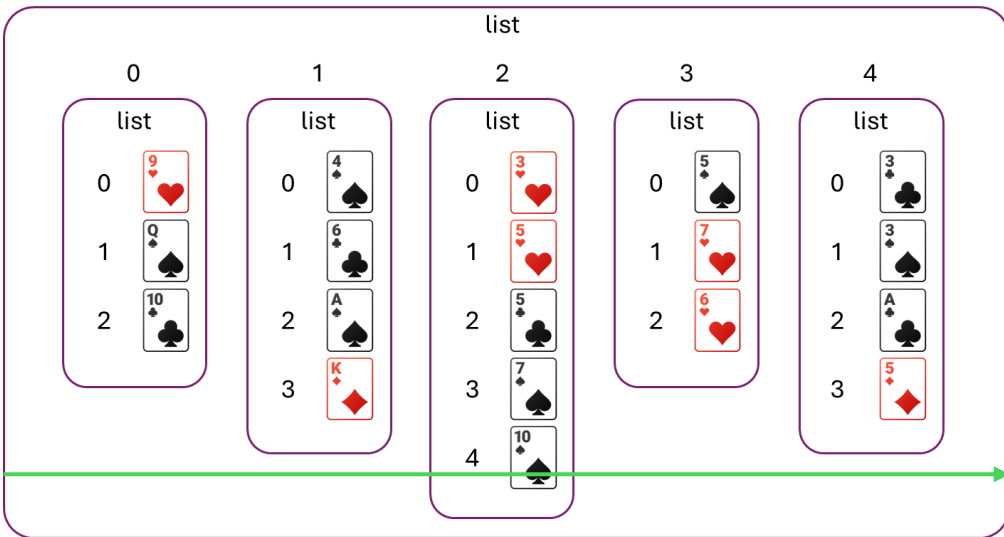
```
[0][0] 9_of_hearts    [1][0] 4_of_spades    [2][0] 3_of_hearts    [3][0] 5_of_spades    [4][0] 3_of_clubs  
[0][1] queen_of_spades [1][1] 6_of_clubs    [2][1] 5_of_hearts    [3][1] 7_of_hearts    [4][1] 3_of_spades  
[0][2] 10_of_clubs    [1][2] ace_of_spades [2][2] 5_of_clubs    [3][2] 6_of_hearts    [4][2] ace_of_clubs
```

Funcionamiento de show_columns



```
[0][0] 9_of_hearts [1][0] 4_of_spades [2][0] 3_of_hearts [3][0] 5_of_spades [4][0] 3_of_clubs
[0][1] queen_of_spades [1][1] 6_of_clubs [2][1] 5_of_hearts [3][1] 7_of_hearts [4][1] 3_of_spades
[0][2] 10_of_clubs [1][2] ace_of_spades [2][2] 5_of_clubs [3][2] 6_of_hearts [4][2] ace_of_clubs
[1][3] king_of_diamonds [2][3] 7_of_spades [4][3] 5_of_diamonds
```

Funcionamiento de show_columns



```
[0][0] 9_of_hearts [1][0] 4_of_spades [2][0] 3_of_hearts [3][0] 5_of_spades [4][0] 3_of_clubs
[0][1] queen_of_spades [1][1] 6_of_clubs [2][1] 5_of_hearts [3][1] 7_of_hearts [4][1] 3_of_spades
[0][2] 10_of_clubs [1][2] ace_of_spades [2][2] 5_of_clubs [3][2] 6_of_hearts [4][2] ace_of_clubs
[1][3] king_of_diamonds [2][3] 7_of_spades [4][3] 5_of_diamonds
[2][4] 10_of_spades
```

Funcionamiento de show_columns

Funcionamiento:

- Encuentra cuál es la columna con más cartas (puede no ser la primera).
- Realiza un bucle que vaya desde el índice 0 hasta el tamaño de columna máximo que se ha encontrado.
 - Para cada índice, crea un string vacío.
 - Visita cada columna y obtiene sus contenidos como un string.
 - Si el índice existe en la lista, hay una carta, se obtienen los índices y el nombre de la carta.
 - Si el índice no existe en la lista, no hay carta, se genera un string vacío (“”).
 - Imprime el string resultante.
 - Continúa con el siguiente índice de carta (fila).

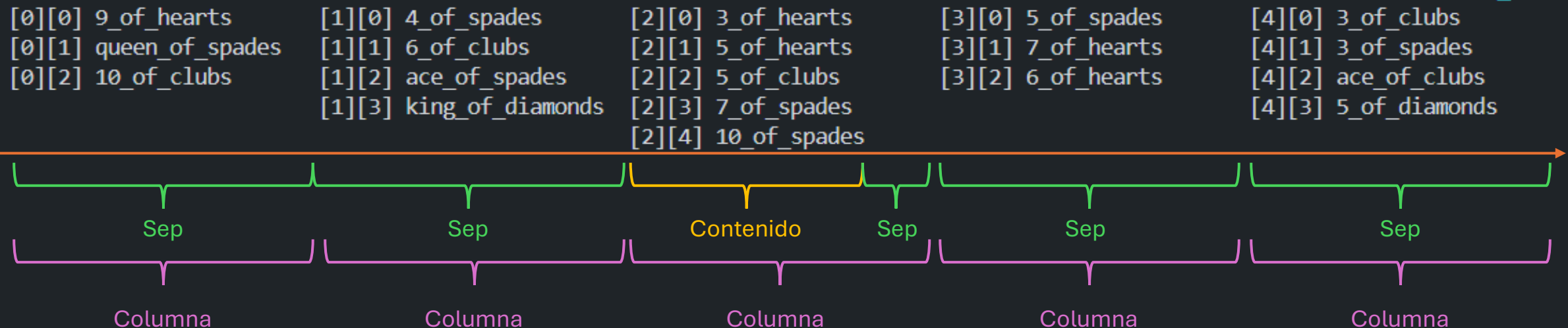
Funcionamiento de show_columns

- Para alinear el contenido, el número de caracteres de la columna será siempre constante.
- Al contenido de la columna se le añaden un número de espacios en blanco determinados hasta completar el número de caracteres.
- Esto se hace con el método `ljust` de la clase `string`.

Columna		Columna		Columna		Columna		Columna	
Contenido	Sep	Contenido	Sep	Contenido	Sep	Contenido	Sep	Contenido	Sep
[0][0] 9 of hearts		[1][0] 4 of spades		[2][0] 3 of hearts		[3][0] 5 of spades		[4][0] 3 of clubs	
[0][1] queen_of_spades		[1][1] 6_of_clubs		[2][1] 5_of_hearts		[3][1] 7_of_hearts		[4][1] 3_of_spades	
[0][2] 10_of_clubs		[1][2] ace_of_spades		[2][2] 5_of_clubs		[3][2] 6_of_hearts		[4][2] ace_of_clubs	
		[1][3] king_of_diamonds		[2][3] 7_of_spades				[4][3] 5_of_diamonds	
				[2][4] 10_of_spades					

Funcionamiento de show_columns

Cuando en una fila hay columnas sin carta, ljust rellenará con espacios en blanco la columna entera.



1. Objetivos y planificación
2. Abstracción
3. Crear baraja
4. Contar parejas
5. Crear columnas
6. Mostrar columnas
7. Mover cartas
8. Trabajo en casa

Mover cartas

Mover cartas

En el módulo `solitaire_logic.py` crea la función `move_card`:

- Recibe la lista de columnas.
- Recibe dos tuplas.
 - 2 índices (columna y fila) de la carta a mover.
 - 2 índices (columna y fila) de la posición de destino.
- Devuelve una nueva lista de columnas actualizada.
 - Debes hacer copias de cada columna, no solo de la lista que los contiene. Si no mutarás las columnas originales.

Puedes utilizar las funciones `pop` e `insert`.

Mover cartas

Doctests:

- Para realizar pruebas con listas multidimensionales, podemos declarar variables globales en el main que sean accedidas desde los tests.
- Esto evita tener que escribirlo cada vez en los tests.

Mover cartas

Puedes usar el siguiente ejemplo

```
if __name__ == "__main__":  
    import doctest  
  
    test_columns = [  
        ['1_of_hearts', '2_of_hearts'],  
        ['3_of_spades', '4_of_spades'],  
        ['5_of_clubs'],  
        ['2_of_diamonds', '3_of_diamonds', '4_of_diamonds']  
    ]  
  
    doctest.testmod(extraglobs={"columns": test_columns}, verbose=True)
```

Solicitar índices

En el modulo `solitaire_user_io.py` crea la función **`ask_indexes`** que solicite al usuario dos enteros (columna y fila) y devuelva una tupla con ambos.

Solicitar movimientos

En el módulo `solitaire_user_io.py` crea la función `ask_move` que solicite al usuario los índices de la carta a mover, y los índices de destino y devuelva una tupla formada por dos tuplas (la de origen y la de destino).

- Utiliza la función `ask_indexes` dentro de esta.

Bucle de juego

En el módulo `solitaire_main.py` crea la función `game_loop`

- Crea una baraja nueva.
- Crea las columnas.
- Realiza 3 movimientos:
 - Muestra las columnas.
 - Solicita al usuario un movimiento.
- Calcula la puntuación de cada columna y los suma, mostrando el resultado.

1. Objetivos y planificación
2. Abstracción
3. Crear baraja
4. Contar parejas
5. Crear columnas
6. Mostrar columnas
7. Mover cartas
8. Trabajo en casa

Trabajo en casa

Trabajo en casa

Modifica `ask_move` para que también reciba la lista multidimensional de columnas como argumento.

Se utilizará para verificar que los índices introducidos existen en cada momento (ten en cuenta que los índices disponibles pueden cambiar con cada movimiento de carta).

Si no existen, se solicitará de nuevo la coordenada hasta que sea correcta.

Trabajo en casa

- Crea funciones que reciban una lista de nombres de cartas y calculen el número de:
 - Poker.
 - 4 cartas con el mismo número consecutivas.
 - Escalera.
 - Secuencia de 5 cartas con números consecutivos.
 - Escalera de color.
 - Secuencia de 5 cartas con números consecutivos del mismo palo.
 - Color
 - Secuencia de 5 cartas consecutivas del mismo color.

Trabajo en casa

- Aplícalas al bucle de juego con las siguientes puntuaciones:
 - Parejas: 1 punto
 - Poker: 5 puntos
 - Escalera: 4 puntos
 - Escalera de color: 7 puntos
 - Color: 3 puntos
- Las mismas cartas pueden puntuar para distintas combinaciones.
 - P ej, un poker implica tener también 2 parejas.