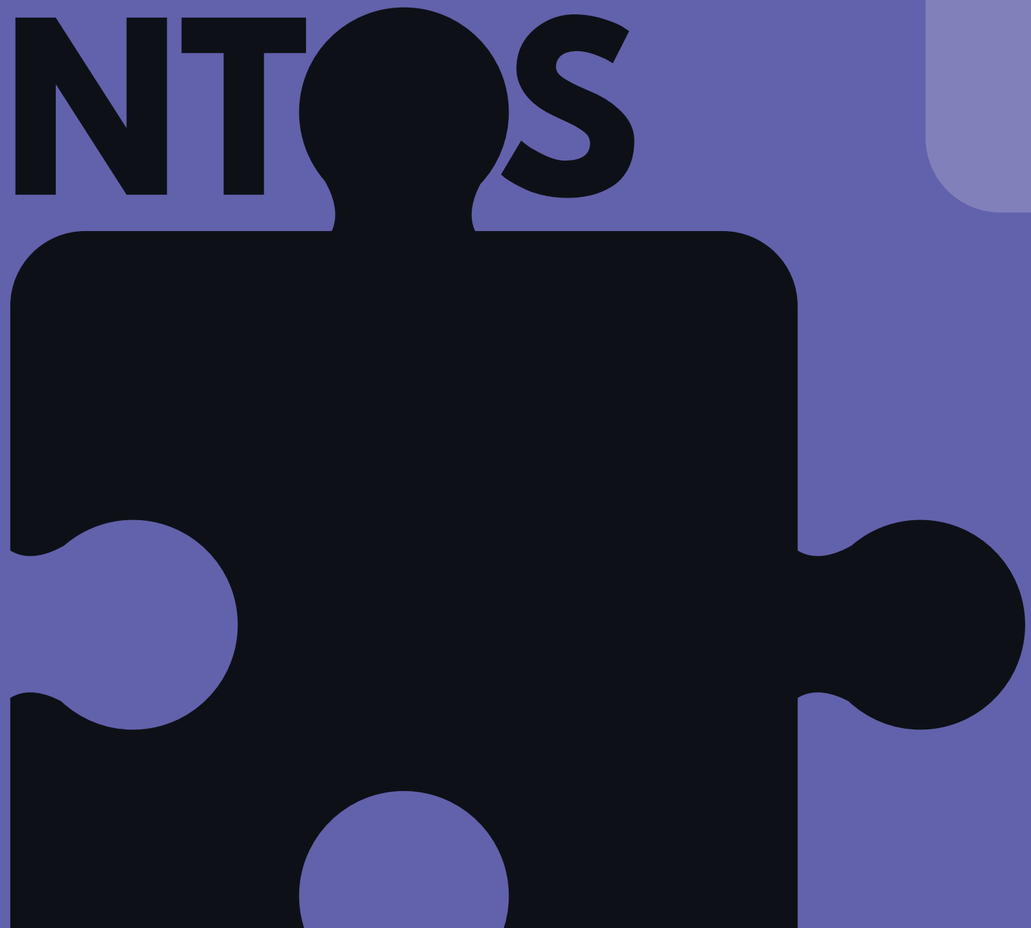
MINIPROYECTO 2

PUZZLE WORD

JUEGO EDUCATIVO BASADO EN PERMUTACIONES PARA MEJORAR LA CONSTRUCCIÓN DE ORACIONES COHERENTES

Asignatura: Matemática Discreta
Alumna: Rojas Castañeda, Ruth Camila
Docente: Rodríguez Rodríguez, Ciro

TEORÍA BÁSICA DE CONJUNTOS



Todos recordaremos las ***operaciones*** *básicas con* ***conjuntos***

Todos recordaremos las ***operaciones*** *básicas con* ***conjuntos***

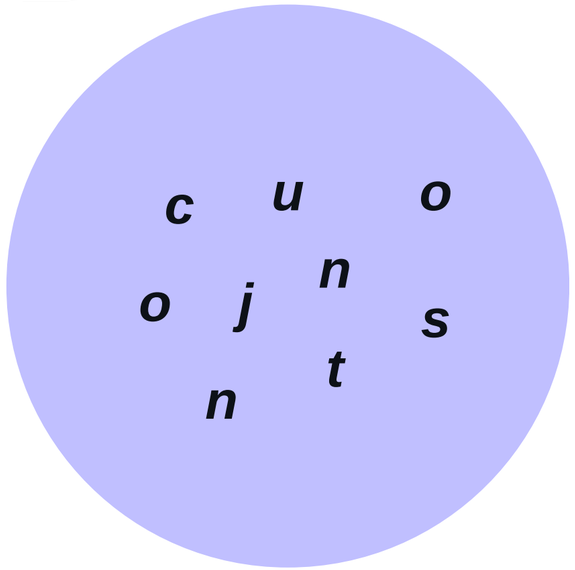
operaciones ***conjuntos***

o p e r a c i o n e s

c o n j u n t o s

A

B

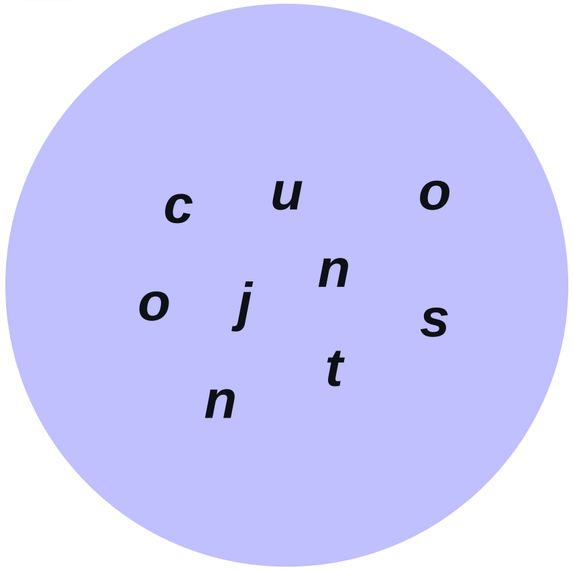


Estos son dos

CONJUNTOS

A

B

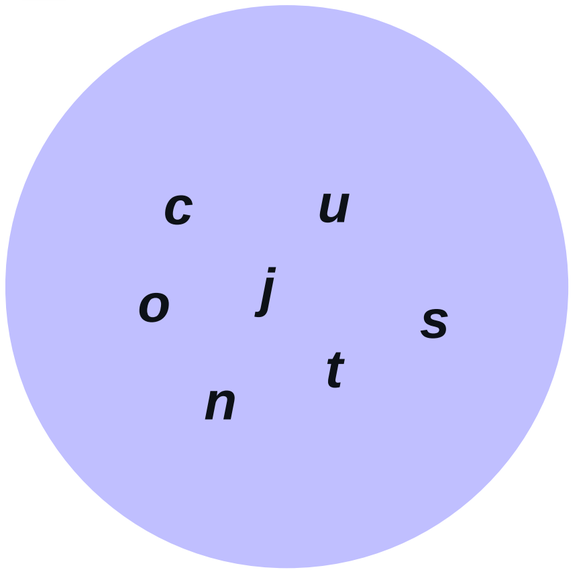


CONJUNTOS

*Claro que tenemos que eliminar todos los **elementos** que se **repiten** primero*

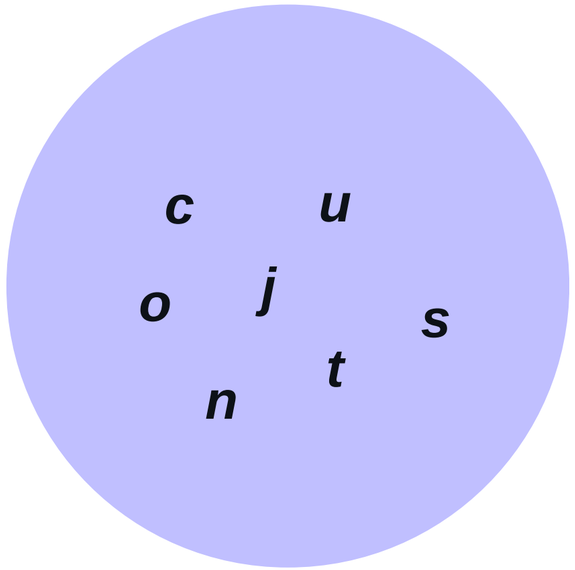
A

B



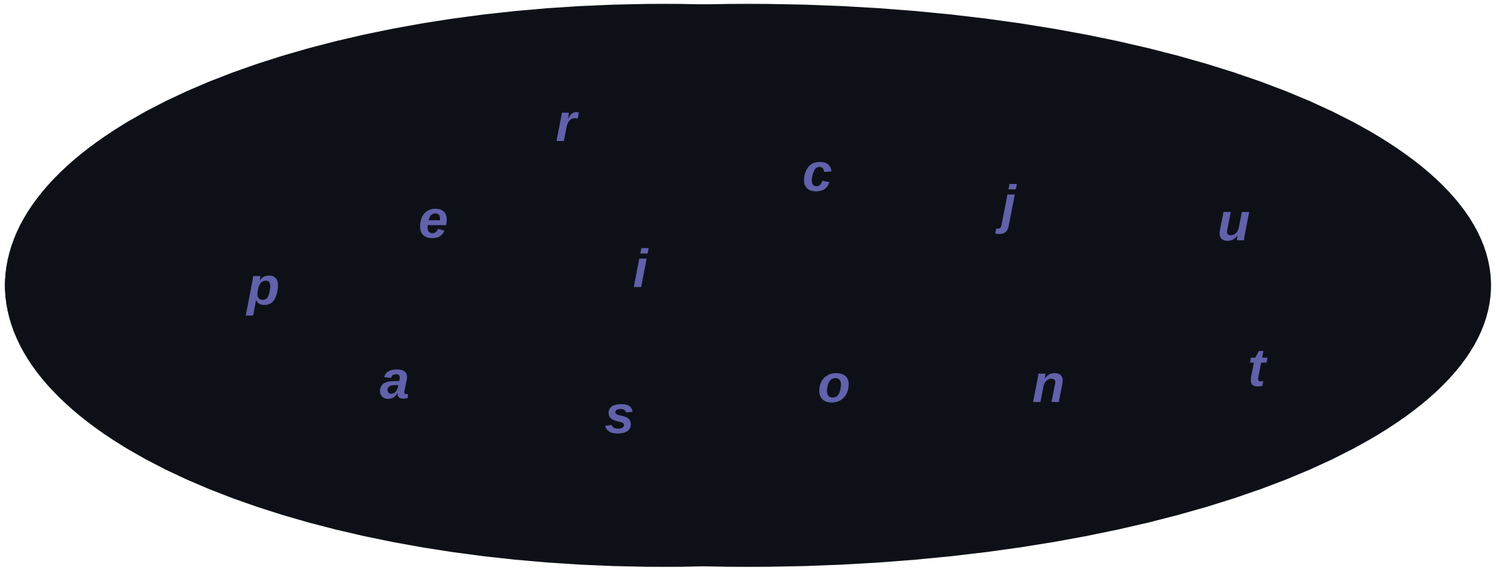
A

B



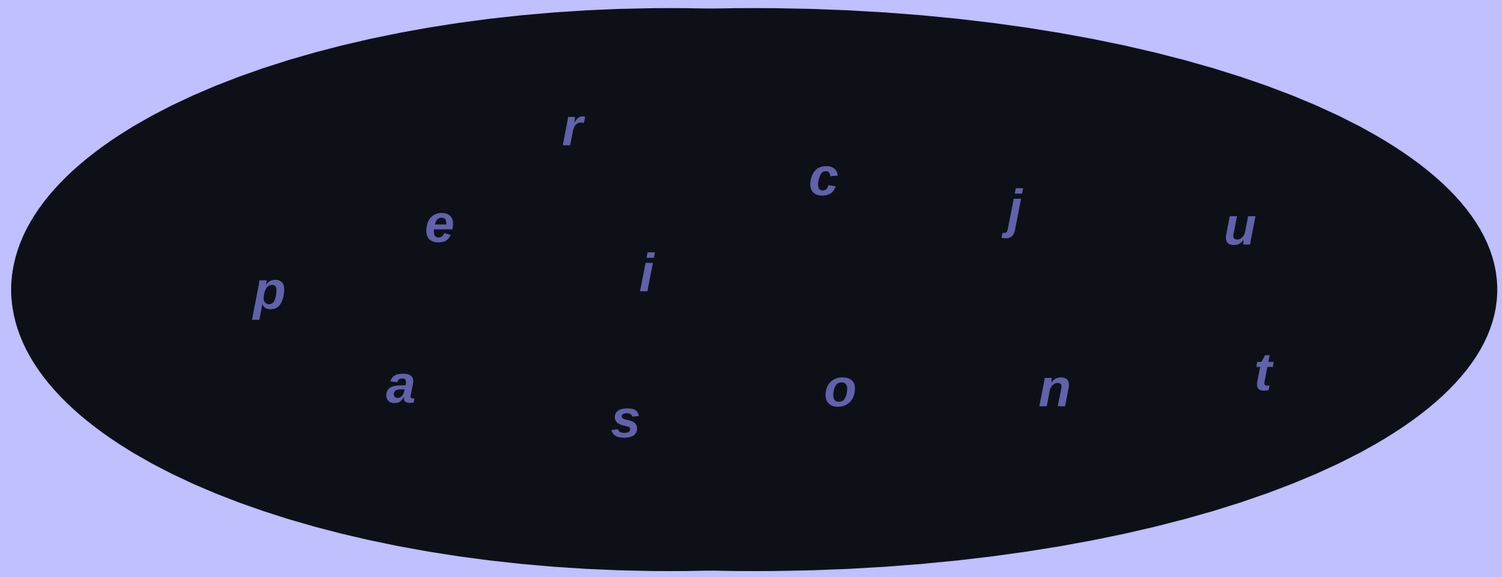
Ahora bien, recordemos:

A U B



UNIÓN

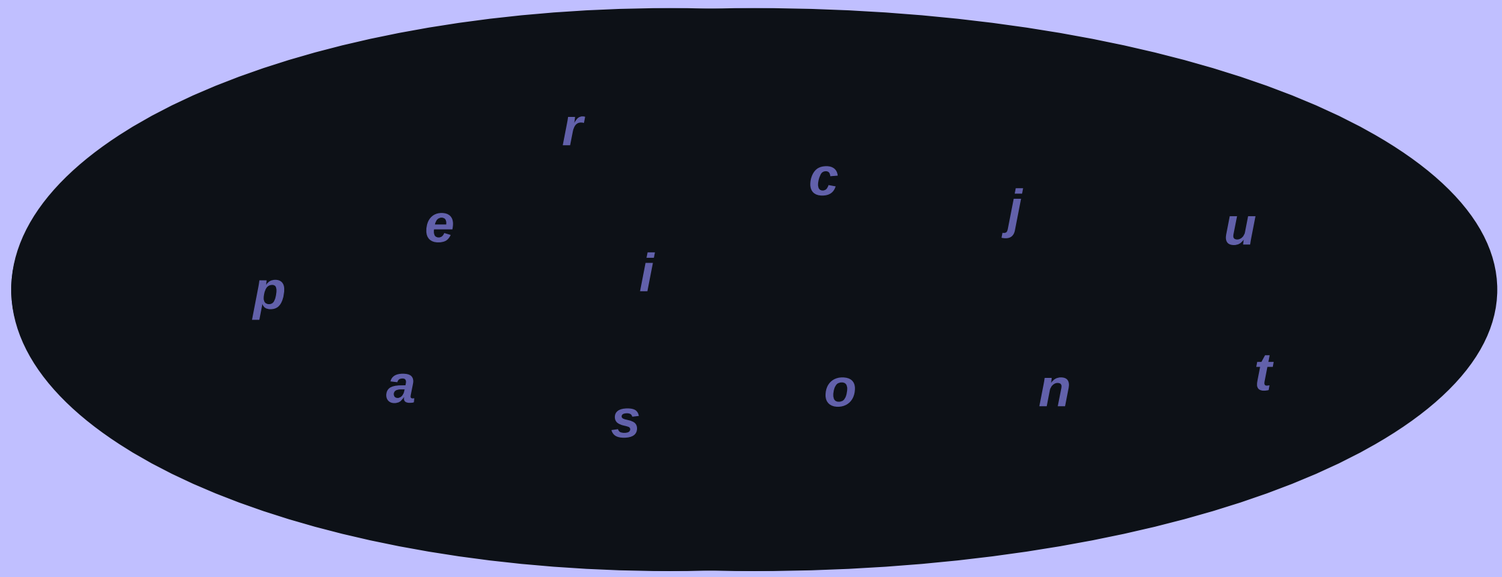
A U B



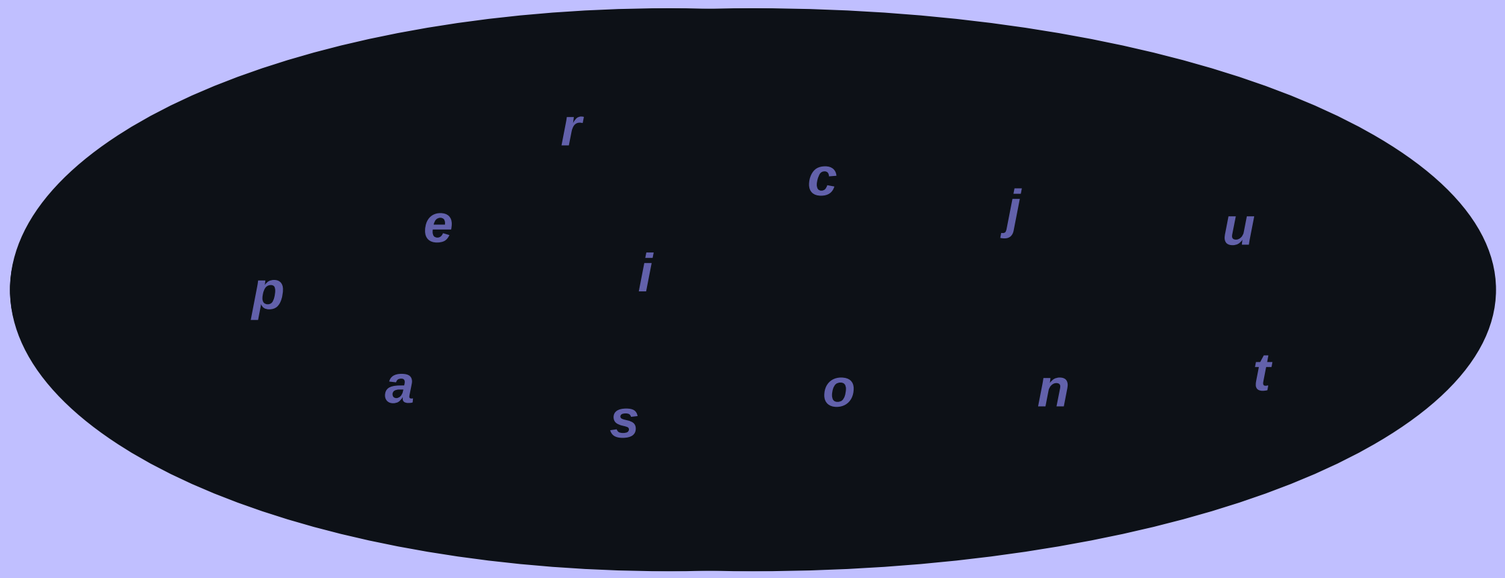
UNIÓN

$A \cup B$

Es un nuevo conjunto que contiene todos los elementos que pertenecen a al menos uno de los conjuntos A o B.

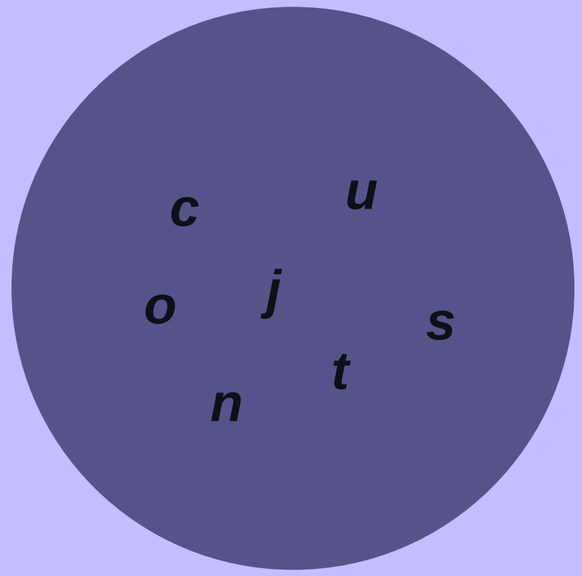


A U B



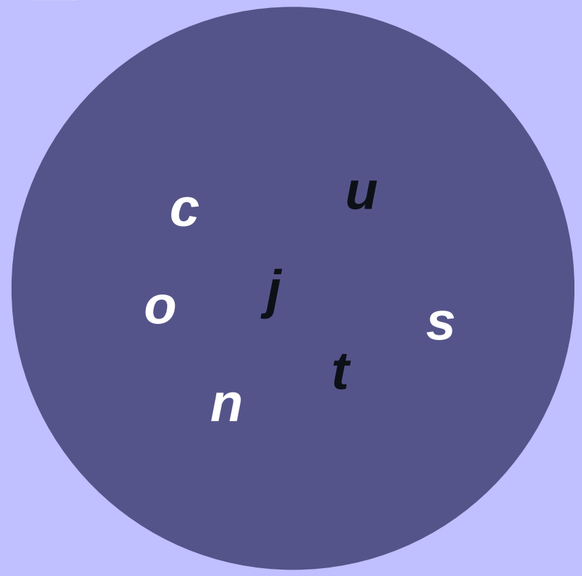
A

B

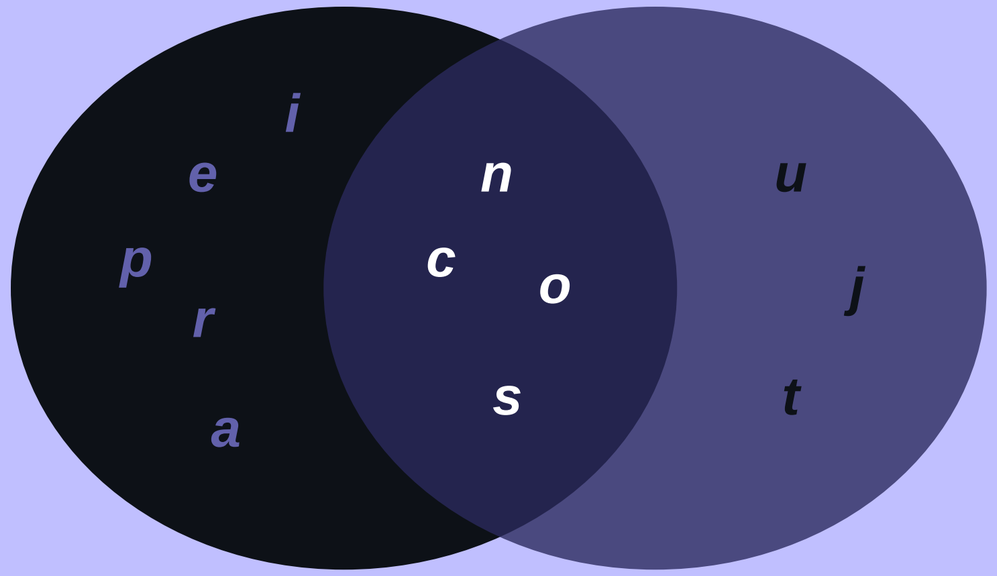


A

B

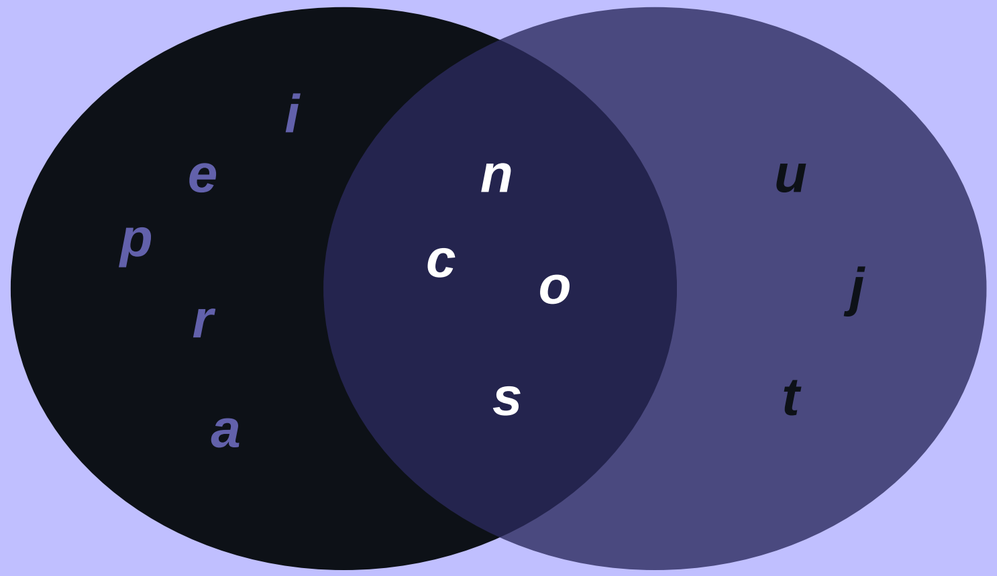


A ∩ B



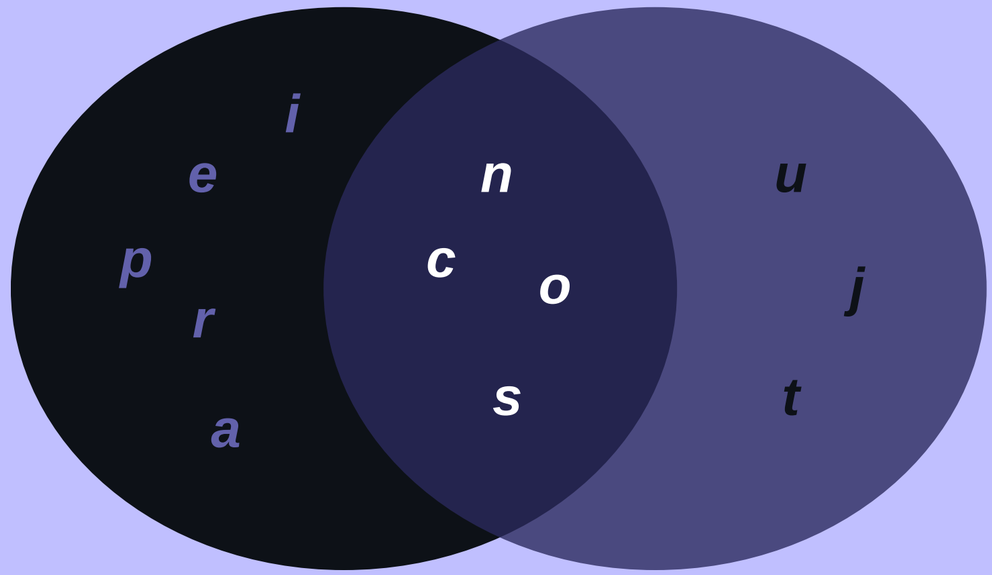
INTERSECCIÓN

$A \cap B$

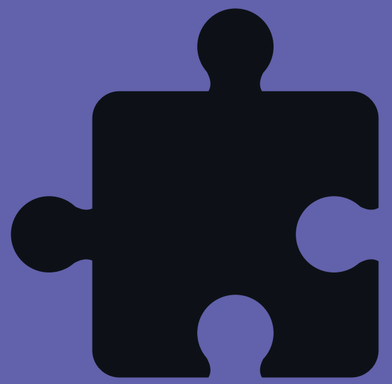


O INTERSECCIÓN

A ∩ B



La intersección recopila todos los elementos comunes a ambos conjuntos.



A - B

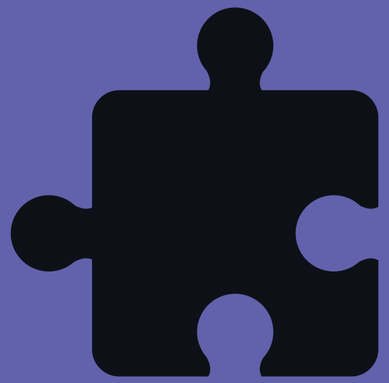
RESTA

A - B

RESTA

A - B

Es un nuevo conjunto que contiene todos los elementos que pertenecen a A pero no a B.



EN PYTHON, UNA LISTA ES UN CONJUNTO

*Una **estructura de datos** que se utiliza para almacenar un **conjunto** ordenado de elementos. Ej: **mi_lista = [1, 2, 3, 4, 5]***

ASÍ QUE ....

¿CÓMO SE APLICAN EN PUZZLEWORD LOS CONCEPTOS DE CONJUNTOS VISTOS?

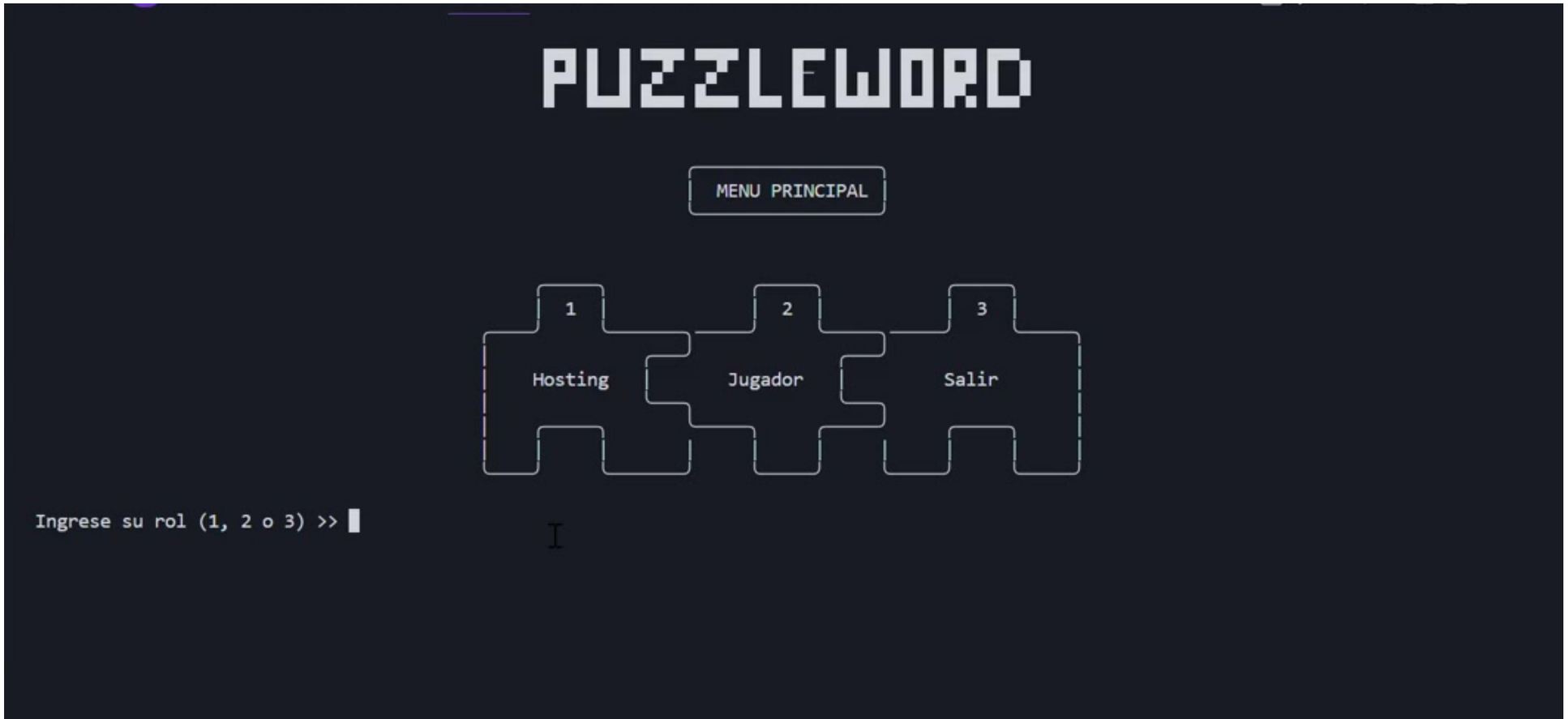
PUZZLEWORD

PUZZLEWORD

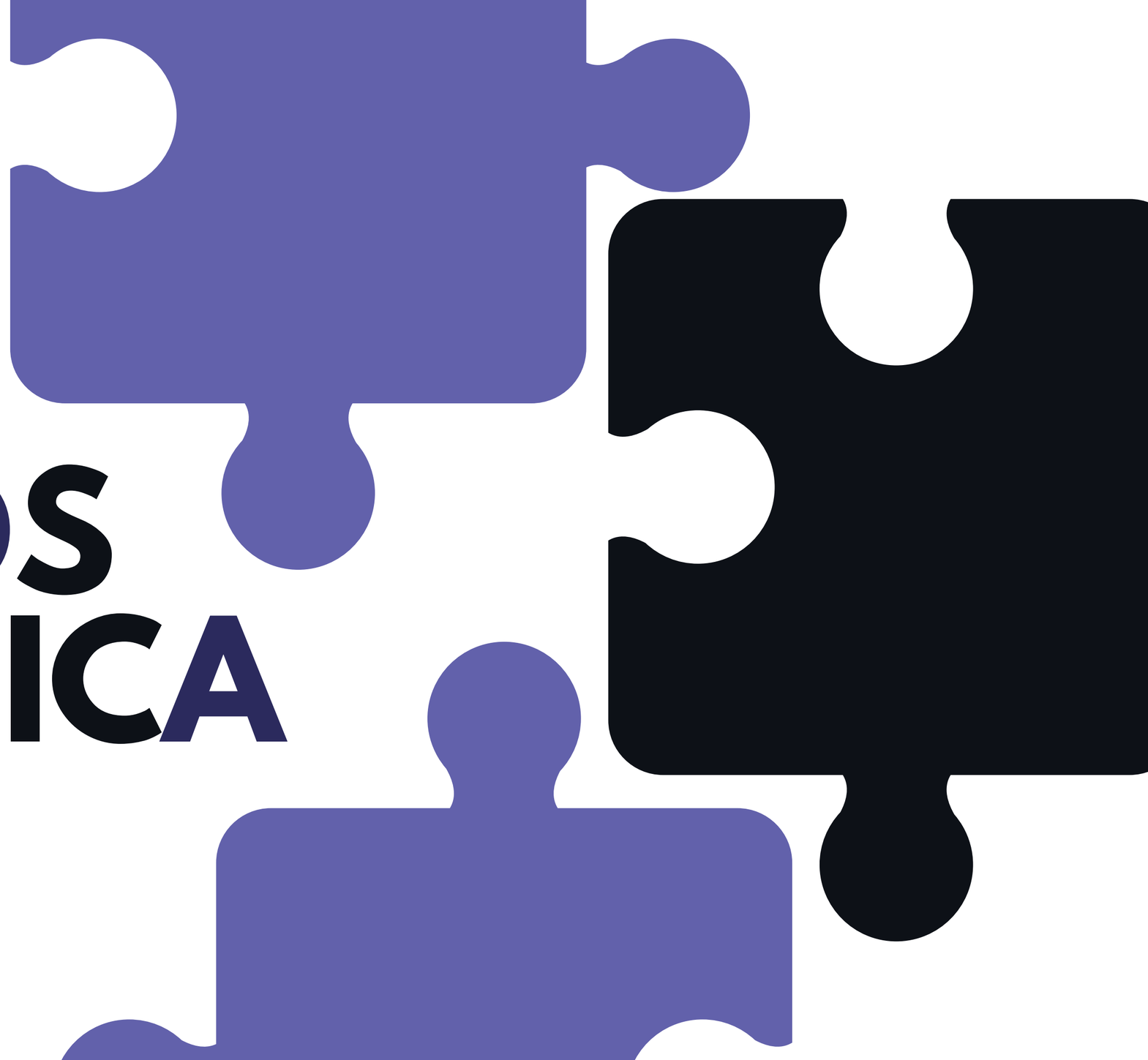
Es un juego de palabras en el que el jugador debe resolver desafíos relacionados con la formación de oraciones a partir de un conjunto de palabras desordenadas

PUZZLEWORD

Es un juego de palabras en el que el jugador debe resolver desafíos relacionados con la formación de oraciones a partir de un conjunto de palabras desordenadas



VEAMOS LA LÓGICA

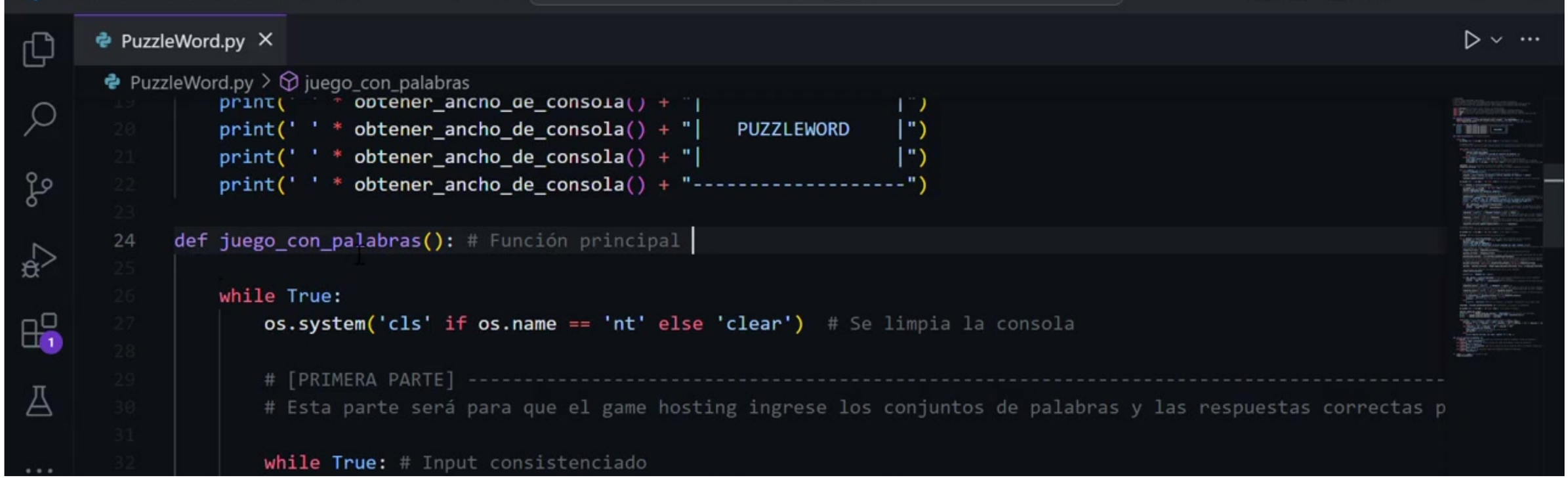


```python
def centrar_texto_en_pantalla(texto):
    espacios_en_blanco = int((os.get_terminal_size().columns - len(texto)) / 2)
    print(' ' * espacios_en_blanco + texto)

def imprimir_nombre_del_juego(): # Función que muestra el nombre del juego
    centrar_texto_en_pantalla("
    centrar_texto_en_pantalla("
    centrar_texto_en_pantalla("

def menu_principal(): # Muestra el menú principal del juego

    conjuntos = [] # Se inicializa la lista vacía llamada "conjuntos"
    respuestas_correctas = [] # Se inicializa la lista vacía llamada "respuestas_correctas"

    while True:

        os.system('cls' if os.name == 'nt' else 'clear') # Se limpia la consola
```

PUZZLEWORD

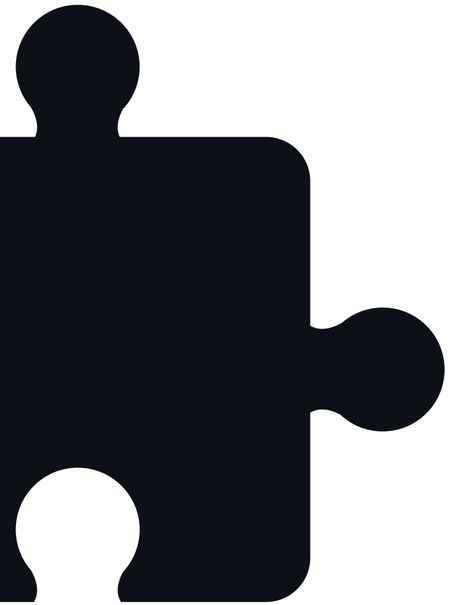
Tenemos 5 funciones principales, dentro de las cuales, *centrar_texto_en_pantalla*, *imprimir_nombre_del_juego* están destinadas a la impresión estética en la consola. Por otro lado, *menu_principal* es la que lanza como opciones los dos modos de juego (hosting y jugador) y la salida del programa.

```python
def centrar_texto_en_pantalla(texto):…

def imprimir_nombre_del_juego(): # Función que muestra el nombre del juego…

def menu_principal(): # Muestra el menú principal del juego…

def realizar_hosting(conjuntos, respuestas_correctas): # Función para el hosting…

def jugar_como_jugador(conjuntos, respuestas_correctas): # Función para el jugador…

if __name__ == "__main__":…
```



Al inicio de la función de menú:
se crean las dos listas vacías más importantes del programa: conjuntos y respuestas_correctas

```
conjuntos = [] # Se inicia
respuestas_correctas = []
```



Hosting

En la primera lista, conjuntos, se le asignarán como elementos las listas de palabras que ingrese el usuario, por ejemplo: la paloma vuela y la niña camina

```
conjunto = input(f"Ingrese las palabras a mezclar separadas por espacios: ").split()
```

En la primera lista, conjuntos, se le asignarán como elementos las listas de palabras que ingrese el usuario, por ejemplo: la paloma vuela y la niña camina

```
conjunto = input(f"Ingrese las palabras a mezclar separadas por espacios: ").split()
```

Con lo que, tendríamos algo así

LISTA CONJUNTOS

[(la,paloma,vuela) , (la,niña,camina)]

LISTA CONJUNTO (CONJUNTOS [0])

LISTA CONJUNTO (CONJUNTOS [1])

```python
for i, conjunto in enumerate(conjuntos):
```

Ahora, para cada lista conjunto de la lista conjuntos, realizaremos todas las permutaciones posibles entre sus elementos utilizando la función permutations de la librería itertools

```
>> [CONJUNTO 1]: ['la', 'paloma', 'vuela']

   Estas son todas las combinaciones posibles con las palabras del conjunto.
   Por favor, seleccione las combinaciones correctas separadas por comas:

   [1]: la paloma vuela
   [2]: la vuela paloma
   [3]: paloma la vuela
   [4]: paloma vuela la
   [5]: vuela la paloma
   [6]: vuela paloma la

   >> RESPUESTA (Ejemplo: 1,3,5):
```

```
>> [CONJUNTO 1]: ['la', 'paloma', 'vuela']

   Estas son todas las combinaciones posibles con las palabras del conjunto.
   Por favor, seleccione las combinaciones correctas separadas por comas:

   [1]: la paloma vuela
   [2]: la vuela paloma
   [3]: paloma la vuela
   [4]: paloma vuela la
   [5]: vuela la paloma
   [6]: vuela paloma la

   >> RESPUESTA (Ejemplo: 1,3,5):
```

En efecto, al tener tres elementos, se generarán 6 permutaciones posibles, pues:

3! = 6

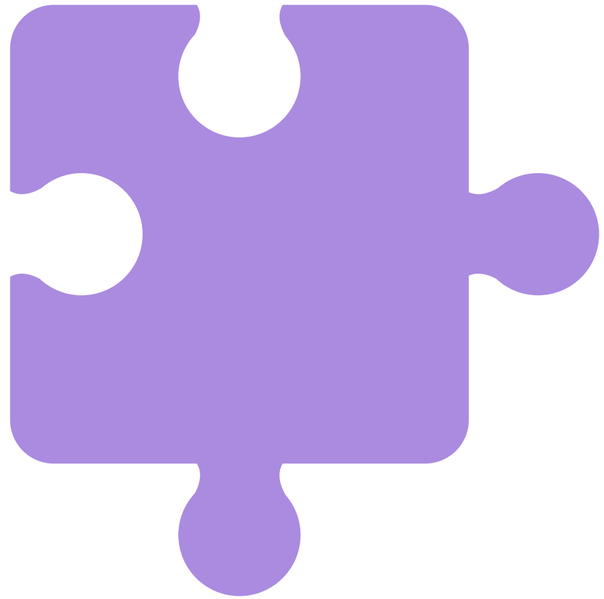
Posteriormente se preguntará si se quiere ingresar otro conjunto de palabras hasta que el hosting responda que no.

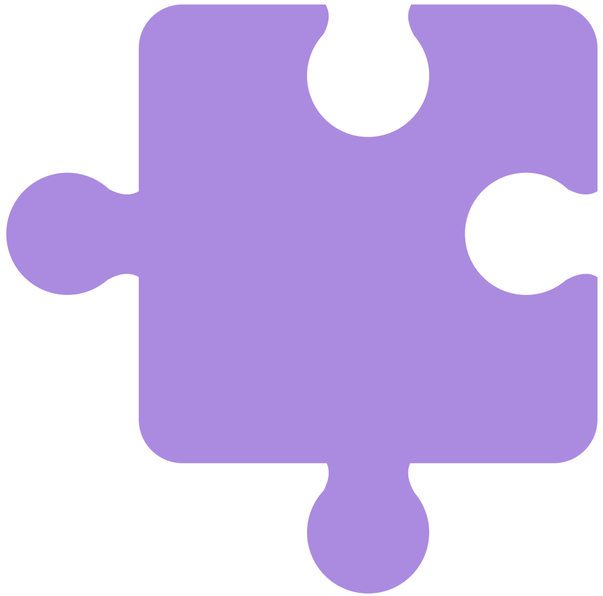
Posteriormente, para cada conjunto, todas sus posibles permutaciones se guardarán en una lista temporal llamada combinaciones, con lo cual nos quedará algo como:

LISTA COMBINACIONES

(la,paloma,vuela)	**LISTA COMBINACION COMBINACIONES [0]**
(la,vuela,paloma)	**LISTA COMBINACION COMBINACIONES [1]**
(paloma,la,vuela)	**LISTA COMBINACION COMBINACIONES [2]**
(paloma,vuela,la)	**LISTA COMBINACION COMBINACIONES [3]**
(vuela,la,paloma)	**LISTA COMBINACION COMBINACIONES [4]**
(vuela,paloma,la)	**LISTA COMBINACION COMBINACIONES [5]**

Cada lista combinación se guardará en una lista temporal oración para concatenarla por espacios en blanco y mostrársela con índice al hosting del juego, quien pasará ahora a establecer las respuestas correctas.





El hosting el juego ingresará las combinaciones correctas por número, las cuales se guardarán en una lista llamada respuestas, lista que, a su vez, se guardará como elemento de la lista inicial que creamos llamada respuestas_correctas.

Posteriormente, se le preguntará si desea ingresar otro conjunto.

El hosting el juego ingresará las combinaciones correctas por número, las cuales se guardarán en una lista llamada respuestas, lista que, a su vez, se guardará como elemento de la lista inicial que creamos llamada respuestas_correctas.

Posteriormente, se le preguntará si desea ingresar otro conjunto.


LISTA COMBINACIONES
(la,paloma,vuela)
(la,vuela,paloma)
(paloma,la,vuela)
(paloma,vuela,la)
(vuela,la,paloma)
(vuela,paloma,la)
LISTA COMBINACION COMBINACIONES [0]
LISTA COMBINACION COMBINACIONES [1]
LISTA COMBINACION COMBINACIONES [2]
LISTA COMBINACION COMBINACIONES [3]
LISTA COMBINACION COMBINACIONES [4]
LISTA COMBINACION COMBINACIONES [5]

LISTA COMBINACIONES

- *(la,paloma,vuela)* → **LISTA COMBINACION** COMBINACIONES [0] ✓
- *(la,vuela,paloma)* → **LISTA COMBINACION** COMBINACIONES [1]
- *(paloma,la,vuela)* → **LISTA COMBINACION** COMBINACIONES [2]
- *(paloma,vuela,la)* → **LISTA COMBINACION** COMBINACIONES [3]
- *(vuela,la,paloma)* → **LISTA COMBINACION** COMBINACIONES [4] ✓
- *(vuela,paloma,la)* → **LISTA COMBINACION** COMBINACIONES [5]

LISTA RESPUESTAS

- *(la,paloma,vuela)* → **LISTA COMBINACION** RESPUESTAS [0] ✓
- *(vuela,la,paloma)* → **LISTA COMBINACION** RESPUESTAS [1] ✓


LISTA RESPUESTAS_ CORRECTAS
LISTA RESPUESTAS (RESPUESTAS_ CORRECTAS[0])
(la,paloma,vuela)
LISTA RESPUESTAS [0]
(vuela,la,paloma)
LISTA RESPUESTAS [1]
Aquí iría la otra lista de respuestas correspondientes al segundo conjunto: RESPUESTAS CORRECTAS [1]

A continuación, para el primer conjunto, se guardará en una lista llamada respuesta_correcta la lista respuestas_correctas [0]

LISTA RESPUESTAS (RESPUESTAS_CORRECTAS[0])

- *(la,paloma,vuela)* → **LISTA RESPUESTAS [0]**
- *(vuela,la,paloma)* → **LISTA RESPUESTAS [1]**

LISTA RESPUESTA_CORRECTA

- *(la,paloma,vuela)* → **LISTA RESPUESTA_CORRECTA [0]**
- *(vuela,la,paloma)* → **LISTA RESPUESTAS_CORRECTA [1]**

La cual, a su vez se guardará en otra lista llamada opciones_correctas

LISTA OPCIONES_CORRECTAS

- *(la,paloma,vuela)* → **LISTA OPCIONES_CORRECTAS [0]**
- *(vuela,la,paloma)* → **LISTA OPCIONES_CORRECTAS [1]**

Jugador

Y ahora, para mostrarle al jugador las alternativas de las cuales deberá seleccionar la correcta, volverá a sacar todas las permutaciones posibles para el conjunto actual y guardarlas en una lista llamada permutaciones_posibles, es decir, el mismo procedimiento llevado a cabo al inicio para obtener la lista combinaciones

LISTA PERMUTACIONES POSIBLES

(la,paloma,vuela)	**LISTA PERMUTACIONES POSIBLES [0]**
(la,vuela,paloma)	**LISTA PERMUTACIONES_POSIBLES [1]**
(paloma,la,vuela)	**LISTA PERMUTACIONES_POSIBLES [2]**
(paloma,vuela,la)	**LISTA PERMUTACIONES_POSIBLES [3]**
(vuela,la,paloma)	**LISTA PERMUTACIONES_POSIBLES [4]**
(vuela,paloma,la)	**LISTA PERMUTACIONES_POSIBLES [5]**

De las cuales, deberemos restar las permutaciones que hemos guardado en la lista opciones_correctas, para obtener la lista opciones_incorrectas


LISTA PERMUTACIONES POSIBLES
(la,paloma,vuela)
(la,vuela,paloma)
(paloma,la,vuela)
(paloma,vuela,la)
(vuela,la,paloma)
(vuela,paloma,la)
LISTA PERMUTACIONES POSIBLES [0]
LISTA PERMUTACIONES_POSIBLES [1]
LISTA PERMUTACIONES_POSIBLES [2]
LISTA PERMUTACIONES_POSIBLES [3]
LISTA PERMUTACIONES_POSIBLES [4]
LISTA PERMUTACIONES_POSIBLES [5]
LISTA OPCIONES_ CORRECTAS
(la,paloma,vuela)
(vuela,la,paloma)
LISTA OPCIONES_CORRECTAS [0]
LISTA OPCIONES_CORRECTAS [1]

LISTA
PERMUTACIONES POSIBLES

-

LISTA
OPCIONES_ CORRECTAS

(A - B)

Diferencia o resta de conjuntos

LISTA
PERMUTACIONES
POSIBLES

LISTA
OPCIONES_
CORRECTAS

LISTA OPCIONES_ INCORRECTAS


LISTA OPCIONES_ INCORRECTAS
(la,vuela,paloma)
LISTA OPCIONES_INCORRECTAS [1]
(paloma,la,vuela)
LISTA OPCIONES_INCORRECTAS [2]
(paloma,vuela,la)
LISTA OPCIONES_INCORRECTAS [3]
(vuela,paloma,la)
LISTA OPCIONES_INCORRECTAS [4]

Finalmente, para mostrar las alternativas al usuario, el programa guardará en una lista llamada opciones los elementos de la lista opciones_correctas y mínimo 5 elementos random de la lista opciones_incorrectas. Gráficamente, podemos verlo de manera sintetizada del siguiente modo:

Lista opciones_correctas

(la,paloma,vuela)

(vuela,la,paloma)

Lista opciones_incorrectas

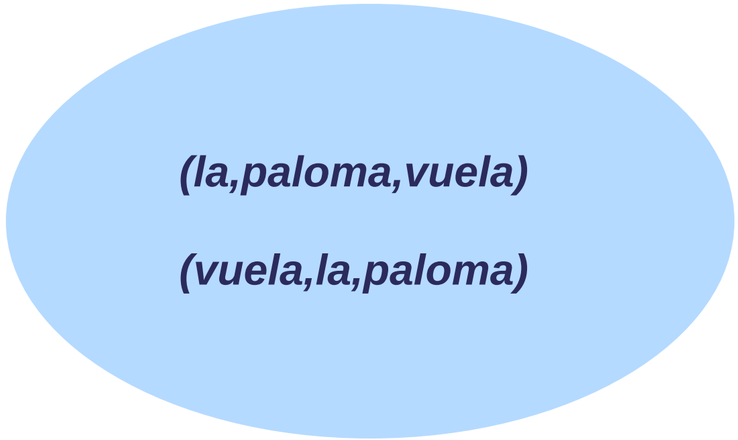
(la,vuela,paloma)

(paloma,la,vuela)

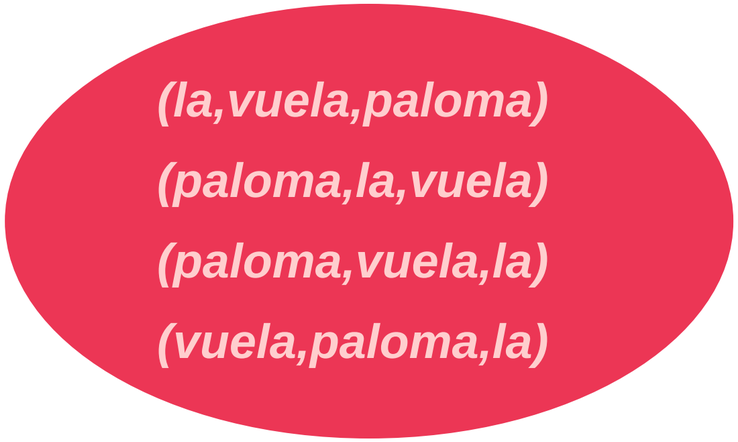
(paloma,vuela,la)

(vuela,paloma,la)

Lista opciones_correctas

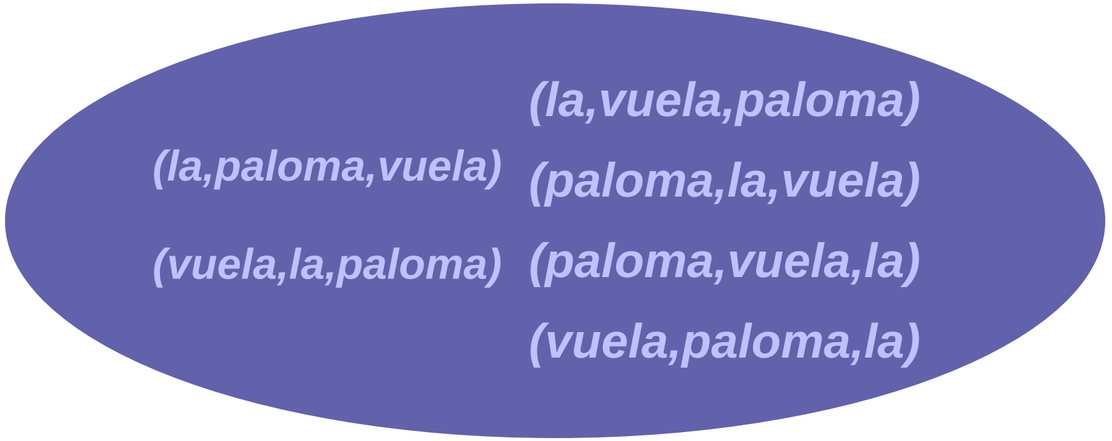


Lista opciones_incorrectas



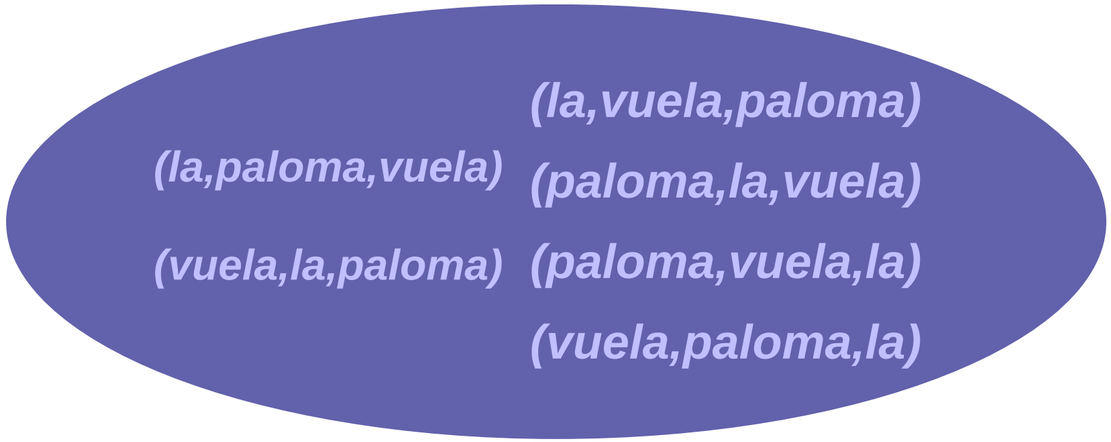
En este ejemplo, desde luego, se incluirán todos los elementos de la lista opciones_incorrectas dado que hay menos de cinco, pero en otros casos el programa de ajustará a la restricción y se estaría eligiendo un subconjunto (sublista) de la lista opciones_incorrectas.

Lista opciones



En este ejemplo, desde luego, se incluirán todos los elementos de la lista opciones_incorrectas dado que hay menos de cinco, pero en otros casos el programa de ajustará a la restricción y se estaría eligiendo un subconjunto (sublista) de la lista opciones_incorrectas.

Lista opciones



De cualquier modo, es sencilllo ver que se está aplicando el concepto de unión de conjuntos:

De cualquier modo, es sencilllo ver que se está aplicando el concepto de unión de conjuntos:

LISTA OPCIONES_ CORRECTAS

+

LISTA OPCIONES_ INCORRECTAS

(o subconjunto de la lista: una "sublista")

(A U B)

Finalmente, el programa verificará si las opciones elegidas por el usuario se encuentran en la lista opciones_correctas y, de ser así, se le sumará un punto. De lo contrario, no.

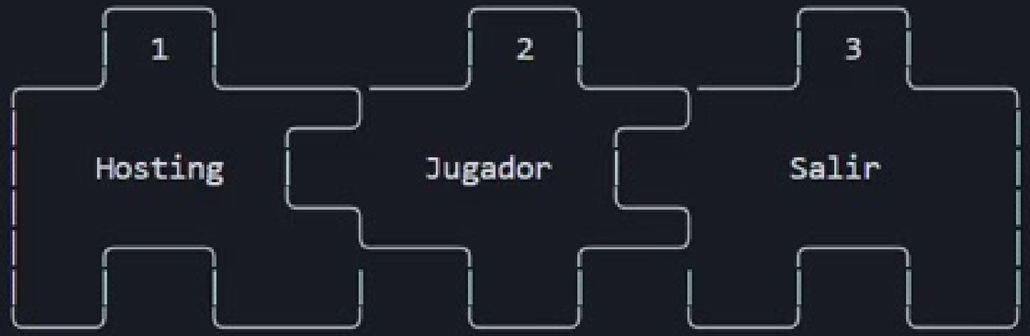
VEAMOS UN TESTING

PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH TERMINAL OUTPUT

Python

PUZZLEWORD

MENU PRINCIPAL



Ingrese su rol (1, 2 o 3) >>

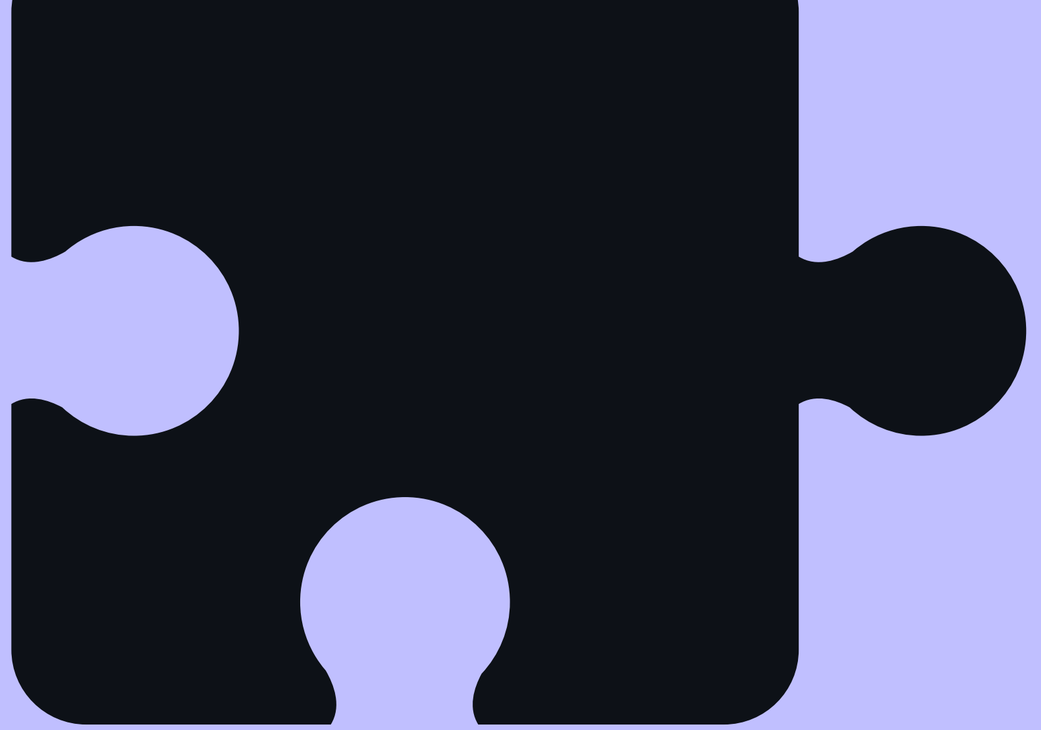
0 Share Code Link Blackbox Search Terminal Output Ln 8, Col 35 Spaces: 4 UTF-8 CRLF Python 3.10.9 64-bit Blackbox

¿CUÁL FUE EL PROBLEMA ENCONTRADO QUE MOTIVÓ EL DESARROLLO DE PUZZLEWORD?

El problema que aborda este proyecto radica en la dificultad que enfrentan las personas que están aprendiendo un nuevo idioma o que tienen desafíos en la comprensión de la gramática para construir oraciones coherentes a partir de palabras desordenadas.

Centrándonos, por ejemplo, en la gramática española, que es en base a la cual se realizaron los testeos de este programa, podemos pensar en su complejidad sintáctica y cómo esta impacta en la comprensión y producción de oraciones coherentes. El español es conocido por su riqueza gramatical, que incluye una amplia variedad de conjugaciones verbales, múltiples categorías gramaticales (verbo, sustantivo, adjetivo, artículo, pronombre, etc.) y una estructura de oración que puede cambiar significativamente según la intención del hablante

PUZZLE WORD

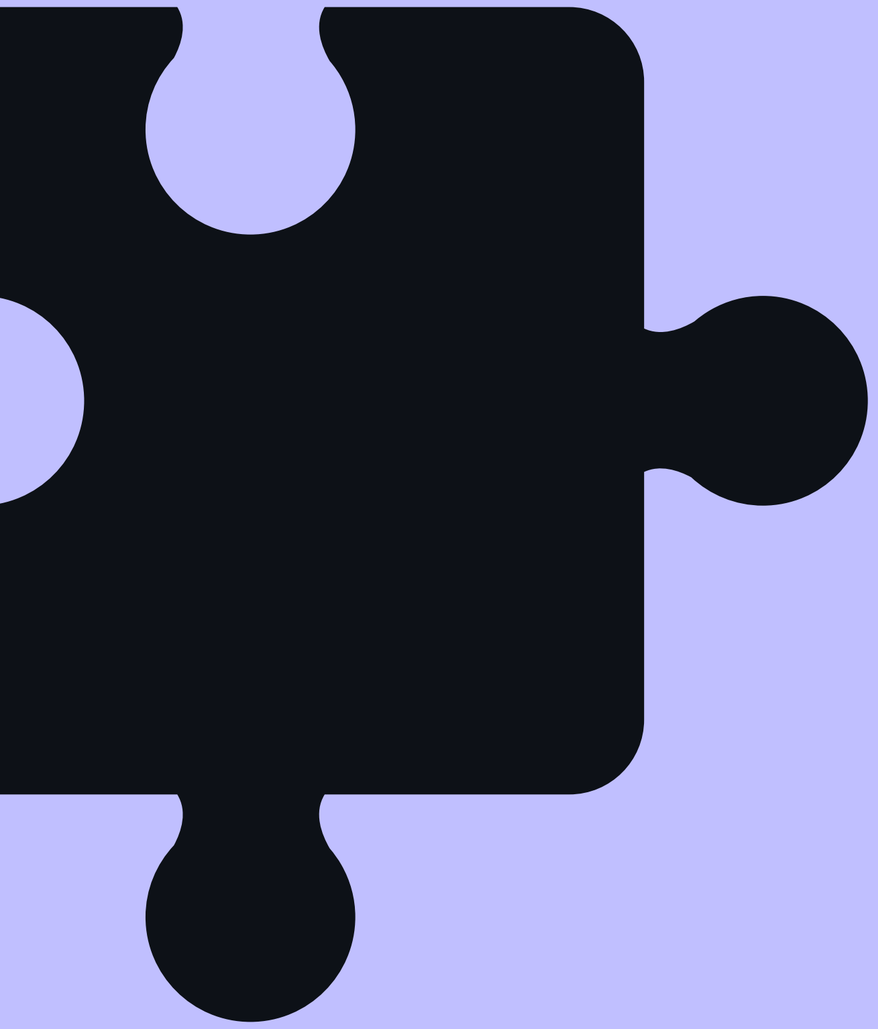


aborda este problema de la vida real al proporcionar a las personas, en particular a aquellos que aprenden español como idioma no nativo, una herramienta efectiva para mejorar su gramática y comprensión lectora. Al enfrentarse al desafío de organizar palabras en oraciones coherentes, los jugadores deben comprender las reglas gramaticales y sintácticas del español. Esto es esencial para estudiantes de idiomas extranjeros y personas que desean fortalecer sus habilidades de escritura y comunicación en español.

OBJETIVO GENERAL

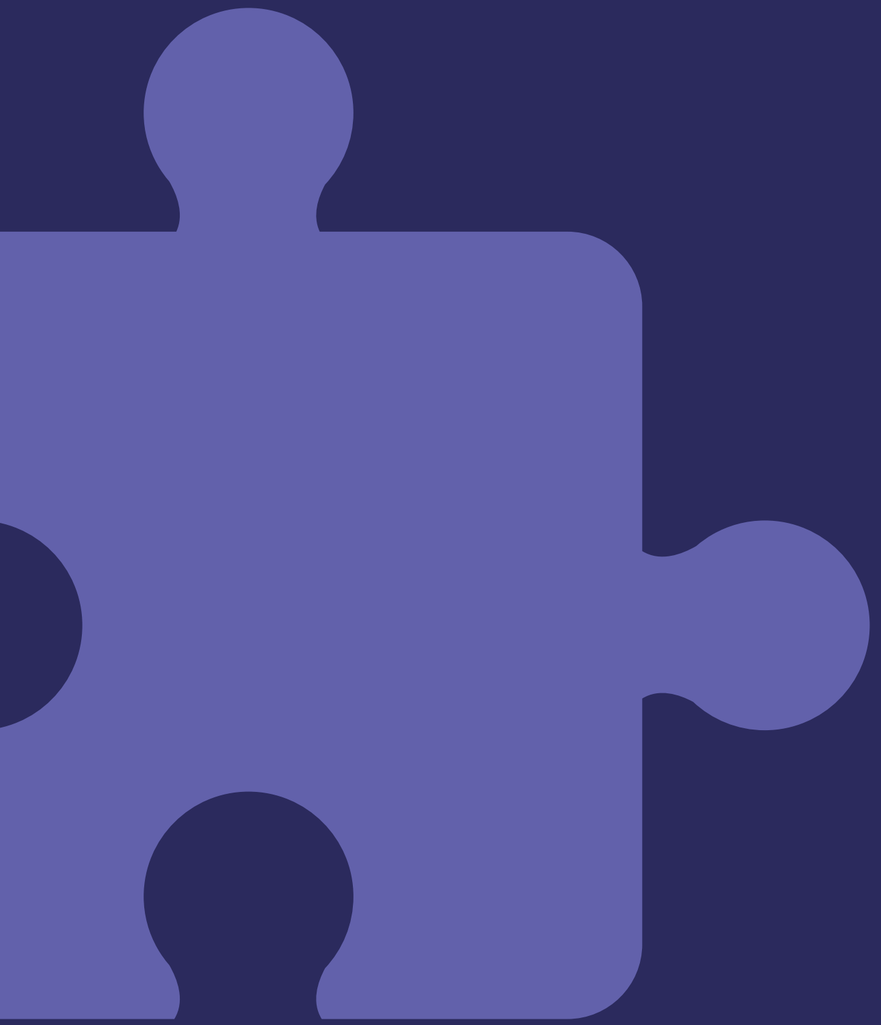
El objetivo general de este proyecto es desarrollar un juego educativo basado en permutaciones y teoría de conjuntos, utilizando la técnica de matemática discreta, que mejore la capacidad de los usuarios para construir oraciones coherentes a partir de palabras desordenadas. Este juego tiene como propósito principal proporcionar una solución efectiva y entretenida para abordar la problemática de la construcción de oraciones en contextos de aprendizaje de idiomas y gramática.

OBJETIVOS ESPECÍFICOS



1.

Implementar un generador de opciones que tome los conjuntos de palabras proporcionados por el usuario y genere las múltiples permutaciones posibles. Estas combinaciones servirán como desafíos para los jugadores y deben ser presentadas de manera aleatoria y variada.

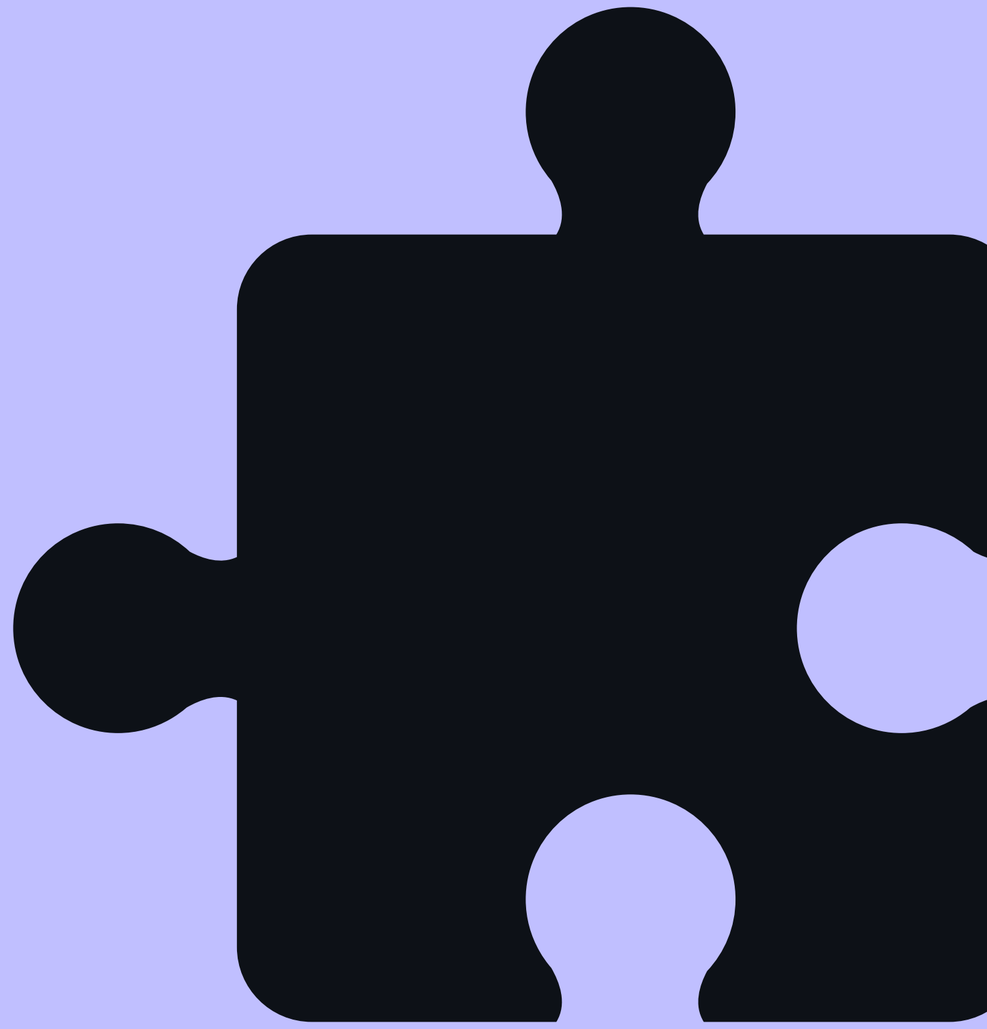


2.

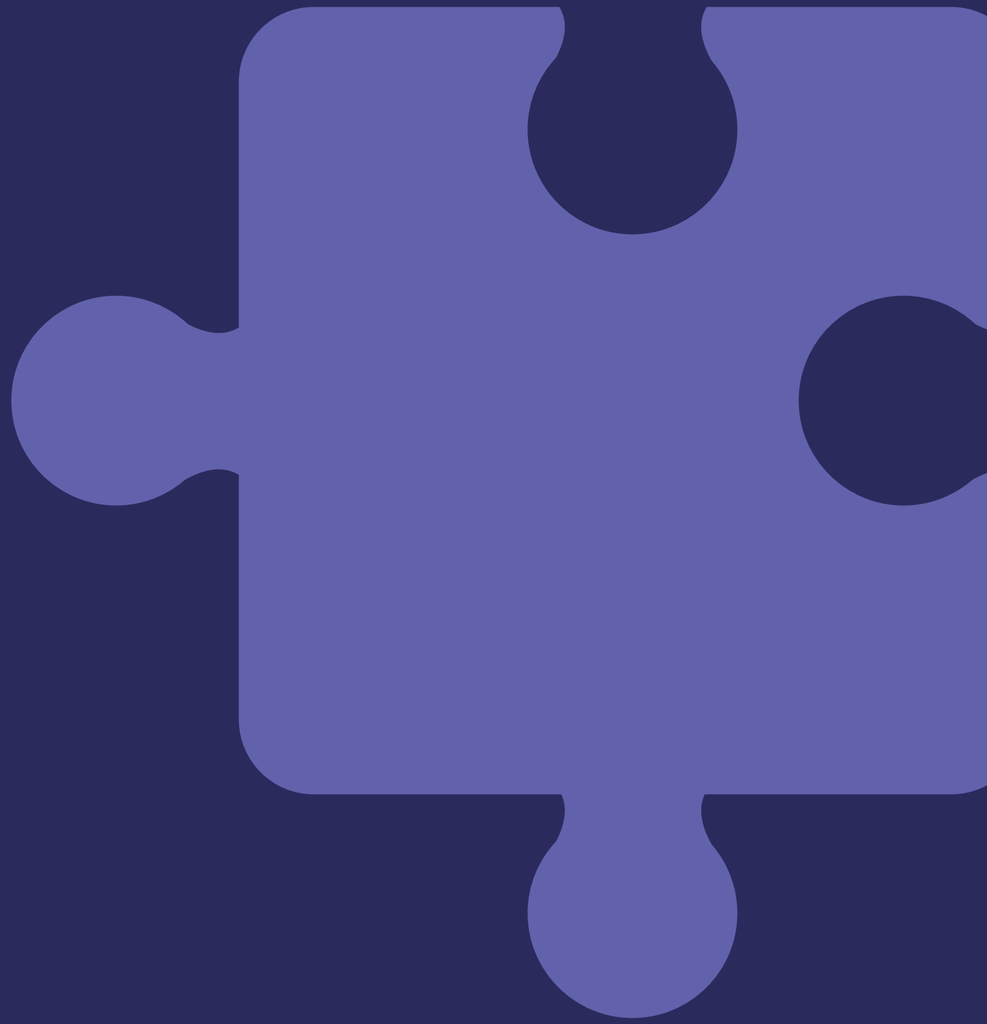
Desarrollar un sistema de evaluación que permita al juego verificar las respuestas proporcionadas por el usuario. Esto implica la comparación de las selecciones del jugador con las respuestas correctas establecidas previamente para determinar si son correctas o incorrectas.

3.

Facilitar la autoevaluación y retroalimentación al jugador. Al proporcionar retroalimentación inmediata sobre el desempeño del jugador y mostrar la puntuación final, se fomenta el aprendizaje autodirigido y la mejora continua de las habilidades lingüísticas en español.



4.



Permitir que el juego se adapte a diferentes niveles de habilidad y necesidades educativas. Esto implica la capacidad de configurar desafíos específicos, como identificar oraciones con estructuras gramaticales particulares o seleccionar oraciones donde ciertas palabras estén en posiciones específicas, lo cual será posible mediante el previo establecimiento de las respuestas correctas por parte del anfitrión y el informe de las reglas hacia los que jugarán.

META

La meta principal es mejorar la habilidad de los usuarios para construir oraciones coherentes a partir de palabras desordenadas en al menos un 60%.

Esto podría medirse a través de un seguimiento del progreso de los usuarios por parte de los anfitriones a medida que avanzan en el juego y adquieren experiencia en la construcción de oraciones correctamente estructuradas.