Instituto Tecnológico de Costa Rica

Ingeniería en Computación

Curso: Investigación de Operaciones, Grupo 60

# Proyecto programación dinámica

Profesor: Ing. Jean Carlos Miranda

Estudiantes:

Adjany Gard Alpízar 2021117164

Omar Madrigal Rodríguez 2020059280

Gabriel Fiatt Vargas 2022180167

Fredrik Jared Aburto Jiménez 2021438331

Verano, 2024

# Descripción del problema

El proyecta busca crear una aplicación web utilizando la librería React. Esta aplicación deberá implementar todo lo necesario para la ejecución de los siguientes algoritmos.

- Arboles binarios
- Rutas más cortas
- Problema de la mochila
- Series deportivas

Es importante que el usuario cuente con todos los componentes webs necesarios para ejecutar cada algoritmo según la necesidad. Además, se debe permitir cargar y crear archivos con los parámetros necesarios para la ejecución de cada algoritmo

# Manual de usuario

## Series deportivas

Para ejecutar el algoritmo de series deportivas lo primero que el usuario debe de hacer es buscar en el menú principal de la aplicación la opción y darle click al botón que dice iniciar.


*Ilustración 1: Menú principal*

Al darle click al botón de iniciar la aplicación web nos enviará al componente creado para ejecutar el algoritmo de series deportivas. En esta página encontraremos un botón de volver en la esquina superior izquierda.


*Ilustración 2: Interfaz principal de las series deportivas*

En esta interfaz podemos observar las siguientes entradas:

**Probabilidad local:** Probabilidad asociado al equipo A cuando este juega de local.

**Probabilidad visita:** Probabilidad asociada al equipo B cuando este juega de local.

**Orden de los partidos:** Conjunto de checkbox para cada partido, este dirá si el partido se jugará en casa o no para el equipo A.

**Calcular:** Botón que ejecuta el algoritmo y crea la tabla de probabilidades.

**Guardar configuración:** Botón necesario para guardar los parámetros ingresados por el usuario en un archivo tipo JSON.

**Seleccionar archivos:** Botón utilizado para cargar los parámetros en alguno de los archivos JSON previamente creados.

Una vez que se hayan cargados los respectivos parámetros, al darle al botón de calcular se le mostrará una tabla de probabilidades al usuario.

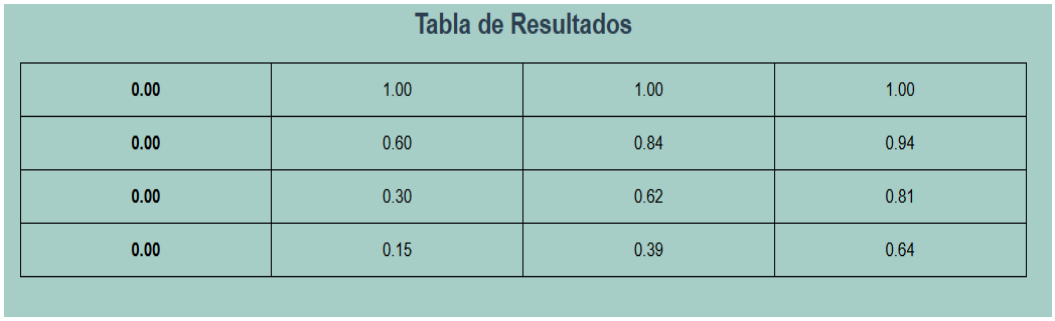


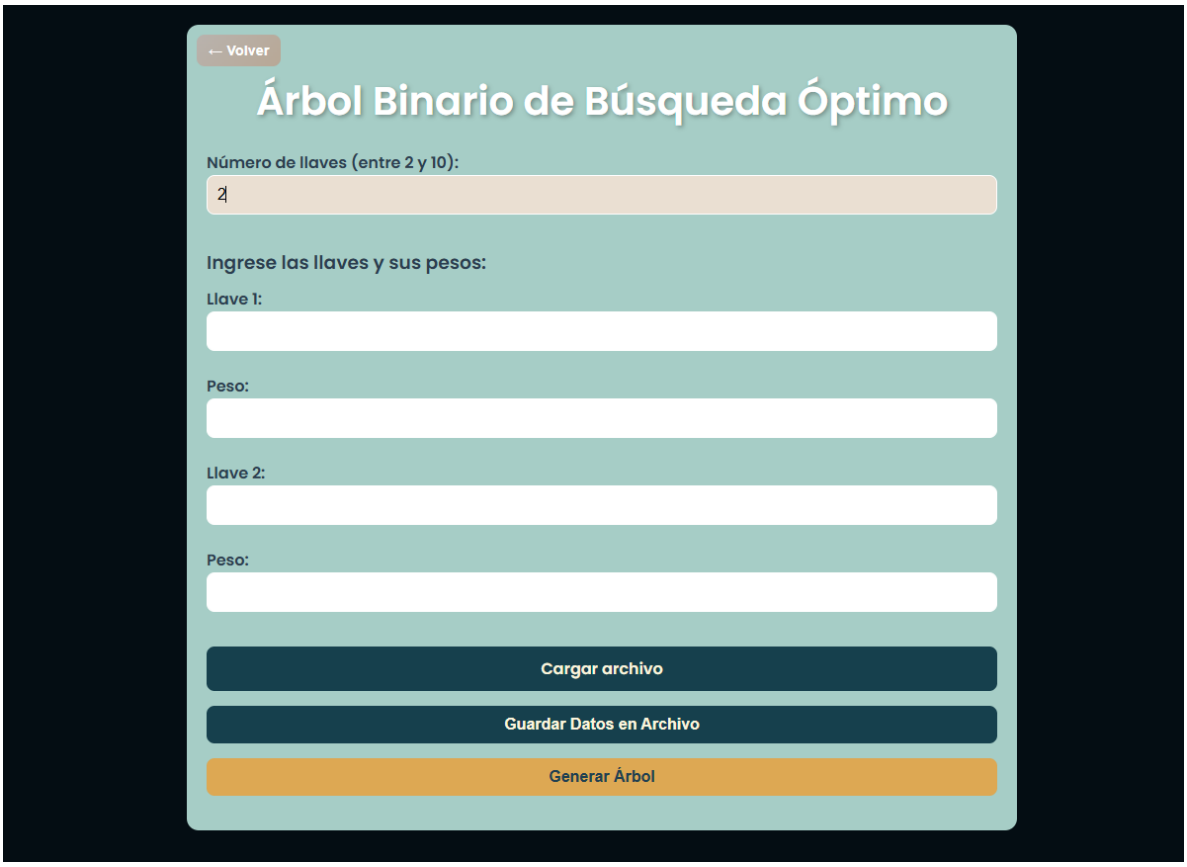

*Ilustración 3: Tabla de probabilidades*

Para calcular el valor de cada casilla se utilizó la siguiente formula:

$$\boldsymbol{tabla}[\boldsymbol{i}][\boldsymbol{j}] = \boldsymbol{p} * \boldsymbol{tabla}[\boldsymbol{i}-\mathbf{1}][\boldsymbol{j}] + \boldsymbol{tabla}[\boldsymbol{i}][\boldsymbol{j}-\mathbf{1}]$$

## Árboles binarios

Otro algoritmo que podemos acceder es el de árboles binarios. En esta página, encontraremos un botón de volver en la esquina superior izquierda.




En esta interfaz podemos observar las siguientes entradas:

**Numero de llaves:** El usuario ingresa la cantidad de llaves que va a tener el árbol binario, en el que puede ser un numero entre 2 y 10.
**Entrada para cada llave con su respectivo peso:** Aparecerán con respecto a la cantidad de numero de llaves ingresada por el usuario entradas con el que ingresa el nombre de la llave y su peso.

**Cargar Archivo:** Botón donde el usuario puede subir un archivo json que contenga la información necesaria para generar el árbol.

**Guardar Datos en Archivo:** Botón donde se descarga un archivo json con los datos ingresados en las entradas.
**Generar Árbol:** Botón donde se genera el árbol binario cuando todas las entradas se hayan llenado.

Y una vez que se hayan cargados los respectivos parámetros, al darle al botón de Generar Árbol, se le mostrará dos tablas al usuario, la tabla A y la tabla R.

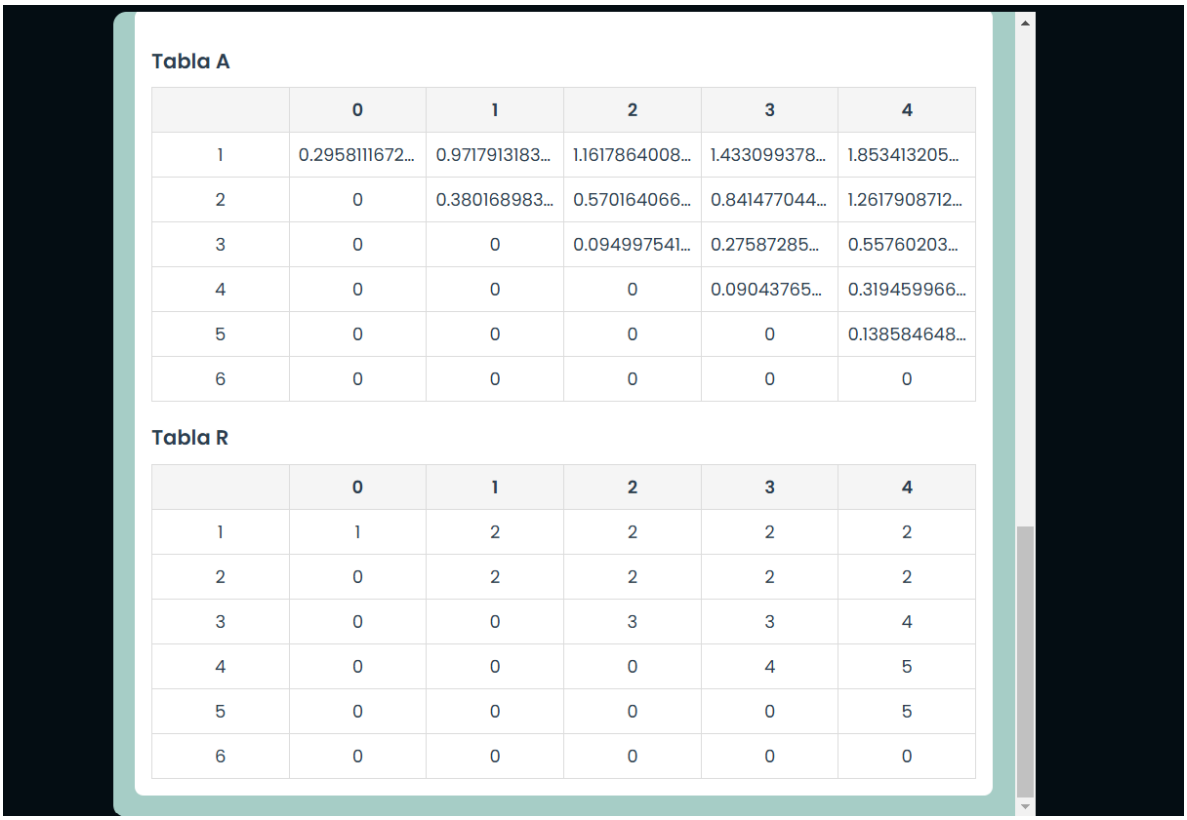


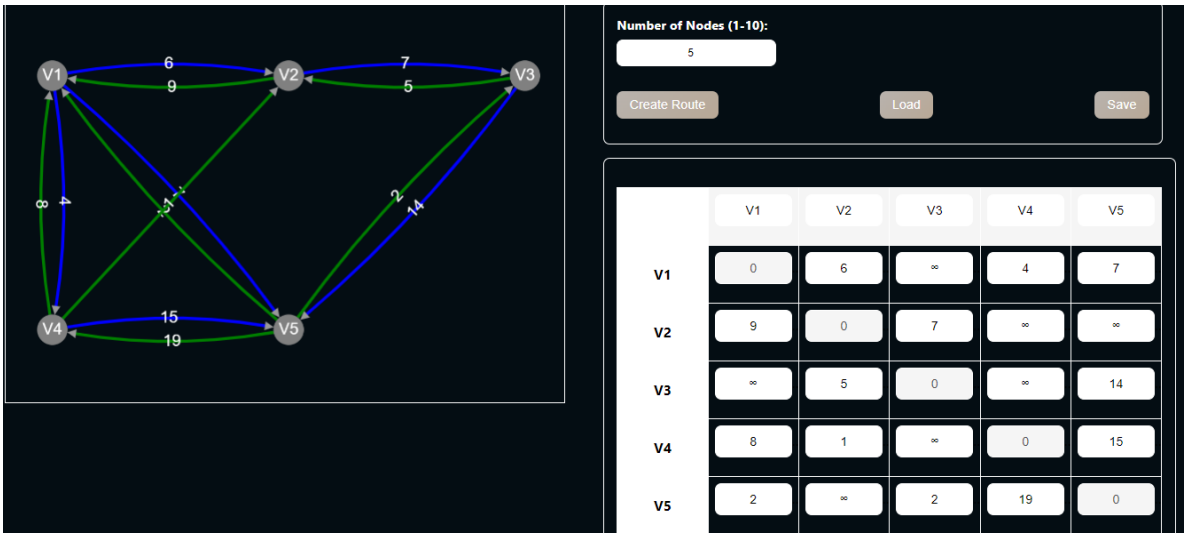
Tabla A

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 1 | 0.2958111672... | 0.9717913183... | 1.1617864008... | 1.433099378... | 1.853413205... |
| 2 | 0 | 0.380168983... | 0.570164066... | 0.841477044... | 1.2617908712... |
| 3 | 0 | 0 | 0.094997541... | 0.27587285... | 0.55760203... |
| 4 | 0 | 0 | 0 | 0.09043765... | 0.319459966... |
| 5 | 0 | 0 | 0 | 0 | 0.138584648... |
| 6 | 0 | 0 | 0 | 0 | 0 |

Tabla R

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 1 | 1 | 2 | 2 | 2 | 2 |
| 2 | 0 | 2 | 2 | 2 | 2 |
| 3 | 0 | 0 | 3 | 3 | 4 |
| 4 | 0 | 0 | 0 | 4 | 5 |
| 5 | 0 | 0 | 0 | 0 | 5 |
| 6 | 0 | 0 | 0 | 0 | 0 |

## Rutas más cortas




En esta parte estamos realizando una prueba que muestra dos componentes principales:

Un grafo visual con nodos (V1, V2, V3, V4, V5) conectados por aristas con pesos

Matrices de distancia (D) y de camino (P) que se actualizan durante el cálculo

Número de Nodos: Campo para especificar la cantidad de nodos (1-10)

Botones de navegación:

Back to Matrix: Regresa a la vista de matriz

Previous: Va al paso anterior

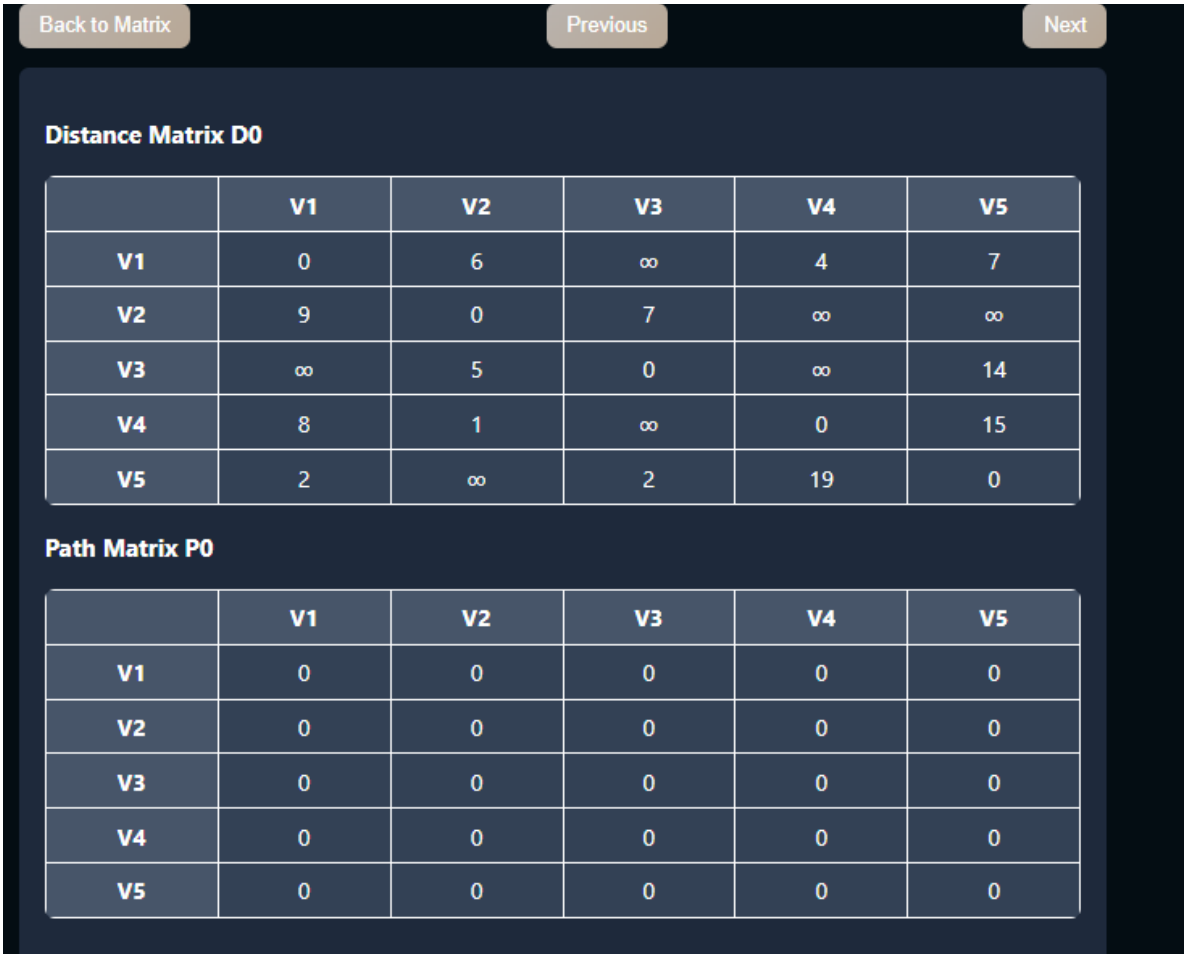
Next: Avanza al siguiente paso

Botones de acción:

Create Route: Genera una nueva ruta

Load: Carga una configuración existente

Save: Guarda la configuración actual




| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 6 | ∞ | 4 | 7 |
| V2 | 9 | 0 | 7 | ∞ | ∞ |
| V3 | ∞ | 5 | 0 | ∞ | 14 |
| V4 | 8 | 1 | ∞ | 0 | 15 |
| V5 | 2 | ∞ | 2 | 19 | 0 |

| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 0 | 0 | 0 | 0 |
| V2 | 0 | 0 | 0 | 0 | 0 |
| V3 | 0 | 0 | 0 | 0 | 0 |
| V4 | 0 | 0 | 0 | 0 | 0 |
| V5 | 0 | 0 | 0 | 0 | 0 |

Matriz de Distancia (D)

Muestra las distancias directas entre nodos

Los valores ∞ indican que no hay conexión directa

La diagonal principal siempre es 0

Matriz de Camino (P)

Indica los nodos intermedios en las rutas óptimas

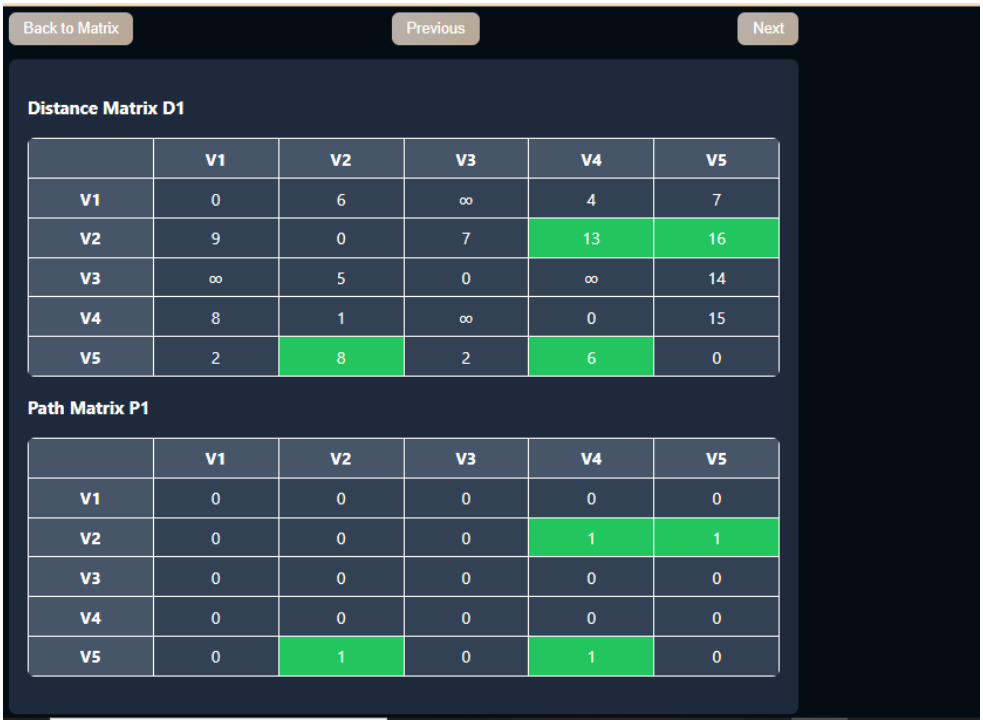
Se actualiza conforme se encuentra un camino mejor

Ahora mostraremos como quedan las tablas de D1, D2,D3,D4, D5

Interpretación de Resultados

Las celdas resaltadas en verde en las matrices indican los caminos actualizados

La distancia mostrada es la longitud total de la ruta más corta




| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 6 | ∞ | 4 | 7 |
| V2 | 9 | 0 | 7 | 13 | 16 |
| V3 | ∞ | 5 | 0 | ∞ | 14 |
| V4 | 8 | 1 | ∞ | 0 | 15 |
| V5 | 2 | 8 | 2 | 6 | 0 |

| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 0 | 0 | 0 | 0 |
| V2 | 0 | 0 | 0 | 1 | 1 |
| V3 | 0 | 0 | 0 | 0 | 0 |
| V4 | 0 | 0 | 0 | 0 | 0 |
| V5 | 0 | 1 | 0 | 1 | 0 |

**Distance Matrix D2**

| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 6 | 13 | 4 | 7 |
| V2 | 9 | 0 | 7 | 13 | 16 |
| V3 | 14 | 5 | 0 | 18 | 14 |
| V4 | 8 | 1 | 8 | 0 | 15 |
| V5 | 2 | 8 | 2 | 6 | 0 |

**Path Matrix P2**

| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 0 | 2 | 0 | 0 |
| V2 | 0 | 0 | 0 | 1 | 1 |
| V3 | 2 | 0 | 0 | 2 | 0 |
| V4 | 0 | 0 | 2 | 0 | 0 |
| V5 | 0 | 1 | 0 | 1 | 0 |

**Distance Matrix D3**

| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 6 | 13 | 4 | 7 |
| V2 | 9 | 0 | 7 | 13 | 16 |
| V3 | 14 | 5 | 0 | 18 | 14 |
| V4 | 8 | 1 | 8 | 0 | 15 |
| V5 | 2 | 7 | 2 | 6 | 0 |

**Path Matrix P3**

| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 0 | 2 | 0 | 0 |
| V2 | 0 | 0 | 0 | 1 | 1 |
| V3 | 2 | 0 | 0 | 2 | 0 |
| V4 | 0 | 0 | 2 | 0 | 0 |
| V5 | 0 | 3 | 0 | 1 | 0 |

**Distance Matrix D4**

| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 5 | 12 | 4 | 7 |
| V2 | 9 | 0 | 7 | 13 | 16 |
| V3 | 14 | 5 | 0 | 18 | 14 |
| V4 | 8 | 1 | 8 | 0 | 15 |
| V5 | 2 | 7 | 2 | 6 | 0 |

**Path Matrix P4**

| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 4 | 4 | 0 | 0 |
| V2 | 0 | 0 | 0 | 1 | 1 |
| V3 | 2 | 0 | 0 | 2 | 0 |
| V4 | 0 | 0 | 2 | 0 | 0 |
| V5 | 0 | 3 | 0 | 1 | 0 |

**Distance Matrix D5**

| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 5 | 9 | 4 | 7 |
| V2 | 9 | 0 | 7 | 13 | 16 |
| V3 | 14 | 5 | 0 | 18 | 14 |
| V4 | 8 | 1 | 8 | 0 | 15 |
| V5 | 2 | 7 | 2 | 6 | 0 |

**Path Matrix P5**

| | V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|---|
| V1 | 0 | 4 | 5 | 0 | 0 |
| V2 | 0 | 0 | 0 | 1 | 1 |
| V3 | 2 | 0 | 0 | 2 | 0 |
| V4 | 0 | 0 | 2 | 0 | 0 |
| V5 | 0 | 3 | 0 | 1 | 0 |

Cómo Encontrar una Ruta Óptima

Haz clic en "Find Optimal Route"

En el diálogo emergente:

Selecciona el nodo de origen en el primer menú desplegable

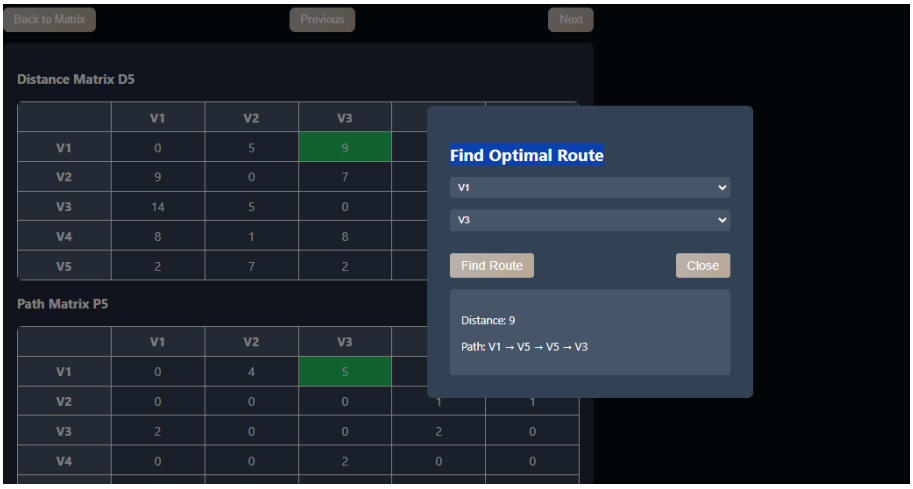
Selecciona el nodo de destino en el segundo menú desplegable

Presiona "Find Route"

La aplicación mostrará:

La distancia total de la ruta óptima

El camino completo (por ejemplo: V1 → V5 → V5 → V3)





Ejemplos de Uso

Como se muestra en las imágenes:

Ruta de V1 a V3: Distancia = 9, Path: V1 → V5 → V5 → V3

Ruta de V1 a V5: Distancia = 7, Path: V1 → V5

## Problema de la mochila

El último algoritmo implementado para este proyecto fue el problema de la mochila, también llamado “Knapsack Problem”. Al ingresar, el menú se ve de la siguiente manera:


El usuario tiene la opción de cargar un archivo JSON ya creado, o bien, guardar uno desde la aplicación. Si no desea hacer esto, el usuario podrá usar la aplicación directamente, ingresando los siguientes datos:

**Capacidad**: Es la capacidad que se tiene en el problema. Por ejemplo, en un caso donde se requieran llevar objetos en un saco que solo soporte cierta cantidad de kilos, esta cantidad deberá ser ingresada en este espacio.

**Cantidad de objetos**: En este espacio, el usuario deberá poner la cantidad de objetos que se estarán tomando en cuenta para ese problema. Una vez ingresado este valor, se generará una tabla con esa cantidad de filas, para que el usuario ingrese los detalles de cada objeto. La tabla que se genera se ve así:


En esta tabla, el usuario deberá ingresar cuatro datos por cada objeto, los cuales son los siguientes:

- Nombre: El nombre que le quiera dar al objeto.
- Valor: El valor que tiene el objeto. El valor total es lo que se busca optimizar.
- Costo: El costo que tiene cada objeto. Por ejemplo, pueden ser kilos, dólares, kilómetros, etc.
- Cantidad: La cantidad que haya de ese objeto. Hay muchos casos en donde se debe poner un 1 en este campo.

Finalmente, al generar la respuesta, se muestra una tabla con los valores generados, y de diferentes colores para que se entienda mejor. Si el valor está en rojo significa que no se está tomando en cuenta ese objeto, mientras que si está en verde significa que sí se está tomando en cuenta. Además de esto, se muestra un texto en donde está la solución óptima al problema. Así es como se ve todo esto:

**Valor Máximo: 29000**

Solución Óptima: 1 Anillo, 1 Póster de Elvis, 1 Radio

| Capacidad | Candelero | Radio | Póster de Elvis | Anillo |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 15000 |
| 2 | 0 | 0 | 0 | 15000 |
| 3 | 0 | 9000 | 9000 | 15000 |
| 4 | 0 | 9000 | 9000 | 24000 |
| 5 | 10000 | 10000 | 10000 | 24000 |
| 6 | 10000 | 10000 | 10000 | 25000 |
| 7 | 10000 | 10000 | 14000 | 25000 |
| 8 | 10000 | 19000 | 19000 | 29000 |

# Análisis de resultados

| Algoritmo | Alcanzado | No alcanzado | Justificación |
|---|---|---|---|
| Series deportivas | Sí | | N/A |
| Mochila | Sí | | N/A |
| Árboles binarios | Sí | | N/A |
| Series deportivas | Sí | | N/A |