9UNIVERSIDAD ESTATAL A DISTANCIA

VICERRECTORIA ACADEMICA

ESCUELA DE CIENCIAS EXACTAS Y NATURALES

CARRERA INGENIERÍA INFORMATICA

Proyecto

# Algoritmo del Banquero

MODALIDAD ESCOGIDA: Proyecto

PROYECTO #1 PARA EL CURSO

DE Sistemas Operativos

PABLO ANDRÉ VALENCIANO BLANCO

1-1572-0043

CENTRO UNIVERSITARIO DE HERERIA

PAC: 2024-2

CIUDAD: HEREDIA

# Contenido



# Índice de Figuras

# Introducción

El siguiente documento dará por presentado el proyecto del curso de sistemas operativos para lo que es el presente año del 2024 y segundo periodo. El cual poseerá no únicamente el documento escrito, sino también un entregable del tipo programa, el cual será diseñado en un IDE y se entregará el ejecutable y el documenta servirá de guía para el uso del mismo.

El proyecto tiene como objetivo: Investigar los temas descritos en el capítulo 13 del libro de texto de curso, así como la administración de procesos en los sistemas operativos mediante un software desarrollado que simule dicho tema. El tema en cuestión es el algoritmo del banquero también conocido como algoritmo de Dijkstra, como tal es un proceso dentro de los sistemas operativos que resuelven los interbloqueos entre los procesos.

Como se vio en la tarea 2 del curso anterior, los interbloqueos se dan cuando hay múltiples tareas que se conforman sea de un hilo o múltiples hilos, y cado unos de estos hilos pertenecen a un proceso específico para cumplir lo que la tarea requiere en la cual su solicitud debe ser cumplida en el mínimo tiempo posible, considerando que los programas requieren realizar estos procesos el computador les debe dar por prestado los recursos y el sistema operativo se encargara de administrarlo, ahí ocasiones que los procesos ocupan los mismos recursos en diferentes cantidades. Un ejemplo puede ser que una unidad lógica algorítmica se ocupe para una suma y también para una multiplicación, pero siendo la multiplicación más complicada requiere el uso del recurso con mayor complejidad que la suma

Para la presentación del proyecto se dividirá en 5 partes:

- Parte 1: Descripción del Algoritmo
- Parte 2: Solución Propuesta
- Parte 3: Interfaz de Usuario y las partes que lo componen
- Parte 4: Manual de Usuario
- Parte 5: Limitaciones

# Parte 1. Descripción del Algoritmo

Un algoritmo por definición de ingeniería es un concepto que llega a dar solución a un problema, el cual se inicia con una hipótesis, se formulan soluciones en ámbito de ingeniería, se hacen las pruebas y se llegan a las conclusiones del caso.

El algoritmo del banquero no es una excepción del caso, y esta nos ayuda a conocer y resolver los interbloqueos que suceden al momento de usar el computador, por medio de matrices entre los procesos que actúan en cada programa, y los recursos que el computador debe brindarle, por lo que no solo es responsabilidad del creador del programa, hacer que sea funcional en todos los dispositivos, el que implementa los sistemas operativos debe procurar que este programa no falle usando los requerimientos mínimos que el diseñador estableció para que el trabajo sea mínimamente satisfactorio.

Como ejemplo, juegos de gama-alta como Red Dead Redemption, Horizon Zero Dawn o emuladores de consola, requieren como mínimo ciertas características gráficas para ser funcionales, a pesar de que el código se trabaje de forma óptima, no va a ser posible que dispositivos muy viejos o con falto de recursos permitan su ejecución ya que requieren una cantidad mayor de procesamiento. Gracias a páginas web, podemos preguntarnos si algún juego es posible ejecutar en nuestro computador o no, esto lo pueden visualizar en la siguiente página web:
https://www.systemrequirementslab.com/cyri

Según el usuario jechaiz de la UNS de Argentina (2013), describe el algoritmo del banquero como: en sistemas operativos es una forma de evitar el interbloqueo, propuesta por primera vez por Edsger Dijkstra. Es un acercamiento teórico para evitar los interbloqueos en la planificación de recursos.

Requiere conocer con anticipación los recursos que serán utilizados por todos los procesos. Esto últimogeneralmente no puede ser satisfecho en la práctica.

Este algoritmo usualmente es explicado usando la analogía con el funcionamiento de un banco. Los clientes representan a los procesos, que tienen un crédito límite, y el

dinero representa a los recursos. El banquero es el sistema operativo. El banco confía en que no tendrá que permitir a todos sus clientes la utilización de todo su crédito a la vez. El banco también asume que si un cliente maximiza su crédito será capaz de terminar sus negocios y retornar el dinero de vuelta a la entidad, permitiendo servir a otros clientes.

El algoritmo mantiene al sistema en un estado seguro. Un sistema se encuentra en un estado seguro si existe un orden en que pueden concederse las peticiones de recursos a todos los procesos, previniendo el interbloqueo. El algoritmo del banquero funciona encontrando estados de este tipo.

También permite determinar si la asignación de recursos inicial, no va a poder cumplir con los estados seguros, sin bloquear al programa.

Figura 1. Representación del Algoritmo del Banquero. Extraído de: https://wiich0.blogspot.com/2010/12/algoritmo-del-banquero.html

# Parte 2. Solución Propuesta

La solución propuesta es simular la forma como este banquero reparte los recursos para cada uno de los procesos, para evitar hacer una aplicación muy compleja y que no va al punto de demostrar el uso del mismo, se decide que se maneje siempre como matriz y vectores de 3 dimensiones, por lo que se concentrara en realizar pruebas numéricas  y no tanto una prueba de estrés o de hasta que punto esta deja de ser funcional, ya que de la tarea anterior esta entre más procesos posea simultáneos, más complejo se vuelve.

Se decide usar el IDE de NetBeans y lenguaje de programación Java, ya que las posibilidades de inserción y validación de datos, es mucho más rápida que hacerlas en C++

Una vez definida la limitación de solo usar 3 dimensiones se avanzará con las validaciones de datos, por lo que en primera instancia se sabe que solo vamos a trabajar con datos numéricos enteros, por lo que no se va a realizar cálculos, sin previamente haberse validado que los datos ingresados sean válidos, el segundo punto de validación es donde una vez definido la cantidad de tipo de recursos, ningún proceso puede demandar más de la cantidad de cada tipo de recursos. Por último, no se pueden asignar más de los tipos de recursos de los demandados.

Ejemplo de las validaciones: En caso de ingresar en la línea de recursos, que cada uno tiene 6 hilos iguales, no se podrá en la demanda pedir mayor a esa cantidad, dado que el algoritmo del banquero requiere que el proceso de algún modo se pueda cumplir los requerimientos en algún punto donde se pueda a llegar a tener la máxima cantidad de recurso.

La segunda validación es más que obvia y trata que no se pueden asignar más recursos de lo que el proceso demanda, imagínese un proceso que no requiere del tipo de recurso 3 y que la persona que ingrese algún dato de tipo de recurso 3 en dicho proceso, ningún sistema operativo va asignar recursos que no va a usar a un proceso.

Por último, por simplicidad se va a usar una única ventana y por lo tanto no se podrá usar iniciar el algoritmo del banquero hasta que se ingresen los datos de forma correcta. Por lo que contara de dos botones, uno de validación de datos y otro en el cual se va a ir iniciando el proceso del algoritmo del banquero, paso a paso liberando los recursos.

# Parte 3. Interfaz de usuario y las partes que lo componen.

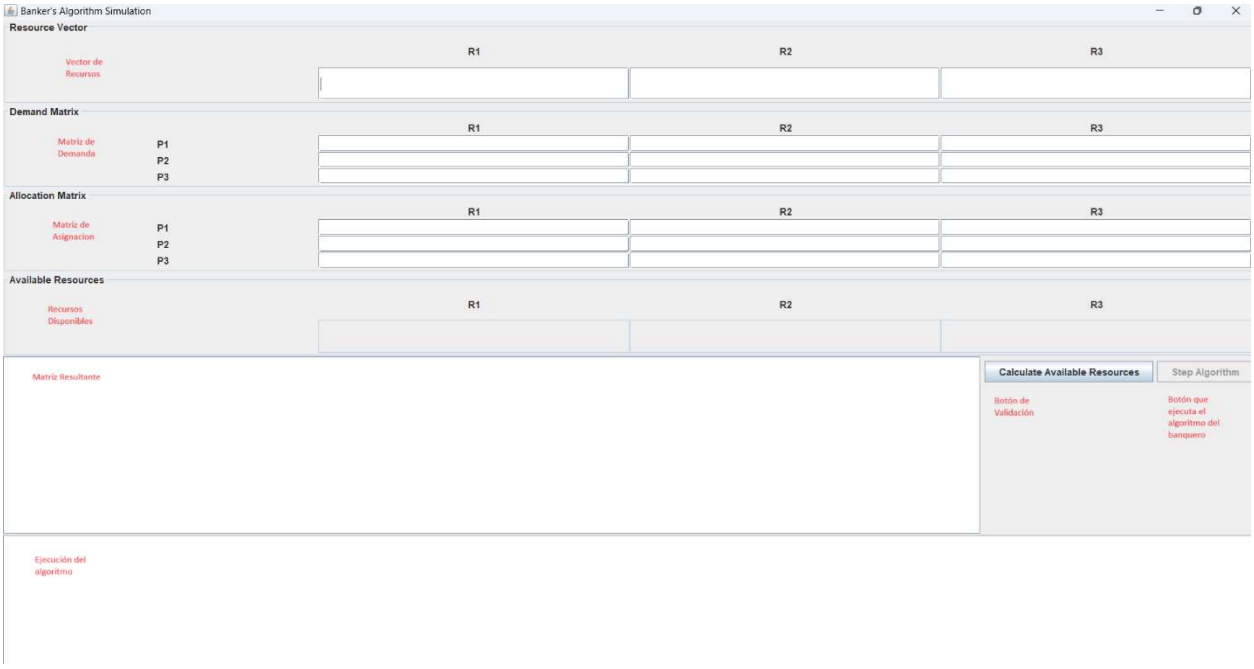



Figura 2. Interfaz Inicial

Las partes que componen a la interfaz son:

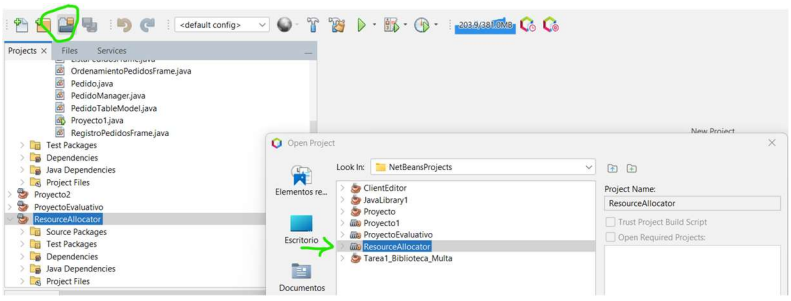
1. Vector de Recursos: Compuesta por los 3 valores que deben ser ingresados por el usuario. Los cuales deben ser valores enteros positivos.
2. Matriz de demanda: Son los valores que requiere cada proceso de cada tipo de recursos.
3. Matriz de asignación: Son los valores que inicialmente se asignaron a cada proceso, según esta asignación puede suceder el deadblock o no.
4. Recursos disponibles: El cual también es un vector, que indica los recursos que quedaron disponibles después de la asignación. Su cálculo se realiza sumando todos los recursos utilizados en cada proceso y restándole ese valor al vector de los recursos para cada uno de los tipos de recursos. Se calcula al presionar el botón de cálculo de recursos disponibles. Se va a ir actualizando según el algoritmo va procediendo

5. Matriz Resultante: Se va a mostrar la matriz de apoyo, que se usara en el algoritmo del banquero, se calcula como la resta de la matriz de demanda contra la matriz de asignación. Se calcula luego de validar los datos.
6. Ejecución del algoritmo: Aquí va a ser donde se va a ver como el algoritmo, determina si los procesos llegan a un modo seguro o el deadblock es inevitable
7. Botón de cálculo de recursos disponibles: Este botón tiene 3 tareas:
    a. Validar los datos.
    b. Calcular los recursos disponibles.
    c. Calcular la matriz resultante.
    d. Habilitar el botón del algoritmo.
8. Botón del algoritmo, esta tiene 2 funcionalidades:
    a. Indicar en la sección de ejecución de algoritmo, que proceso se asignan los recursos.
    b. Liberar los recursos ya usados.

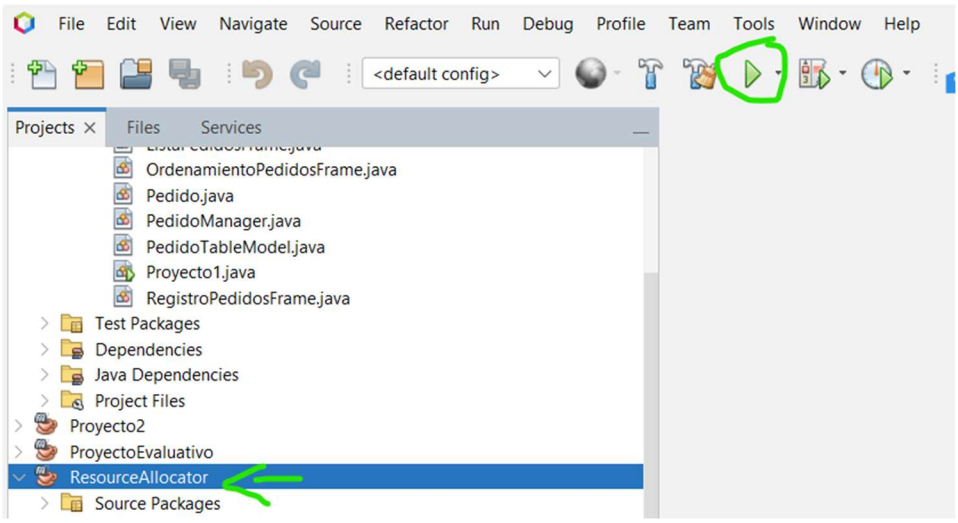
# Parte 4. Manual de Usuario

## Instalación

Abrir Netbeans y abrir el proyecto nombrado como “Resource Allocator”



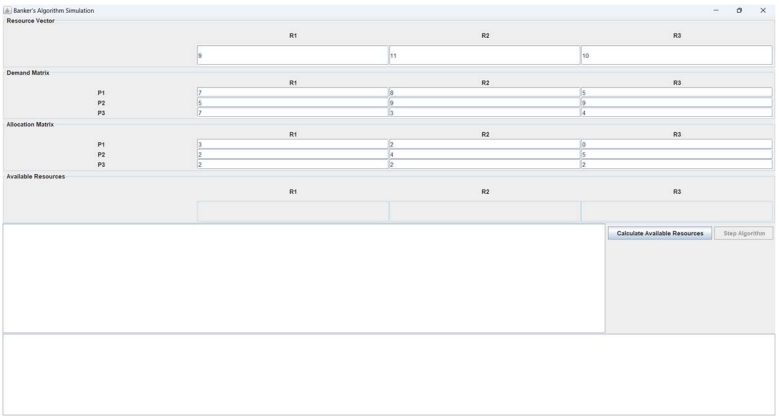

Seleccionas el proyecto en la barra de trabajo y se le da al botón “play”.



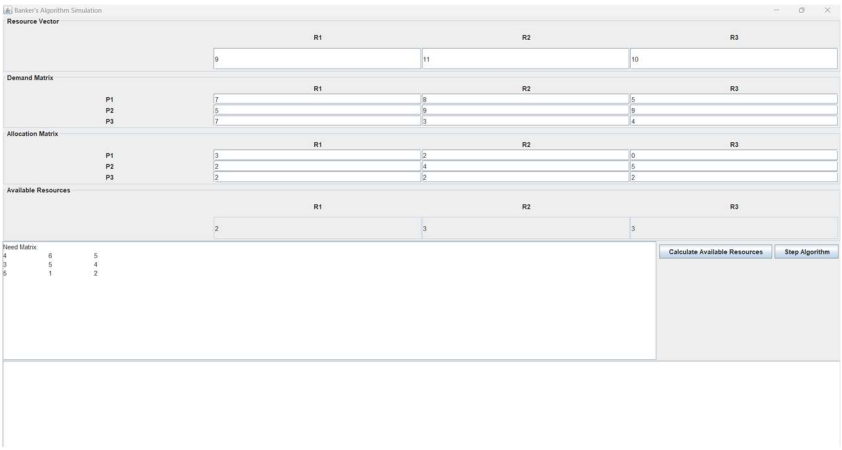

Este se va abrir en modo ventana, abrirlo en modo completo para ver todos los componentes que la componen.

## Ejecución

Ingresamos los datos en los vectores y matrices.

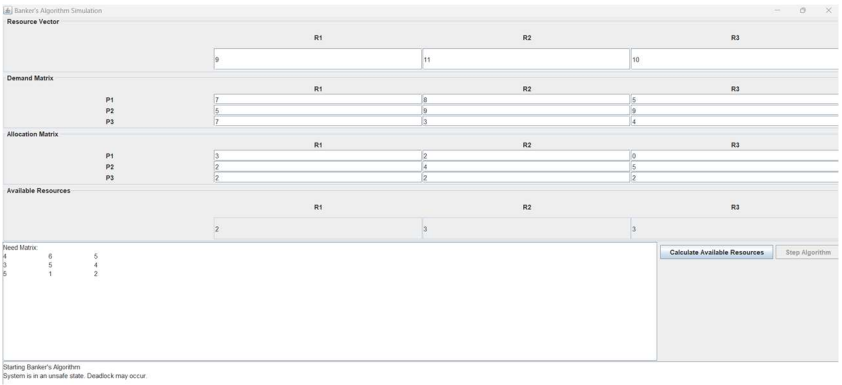


Se presiona el botón de cálculo de recursos disponibles.

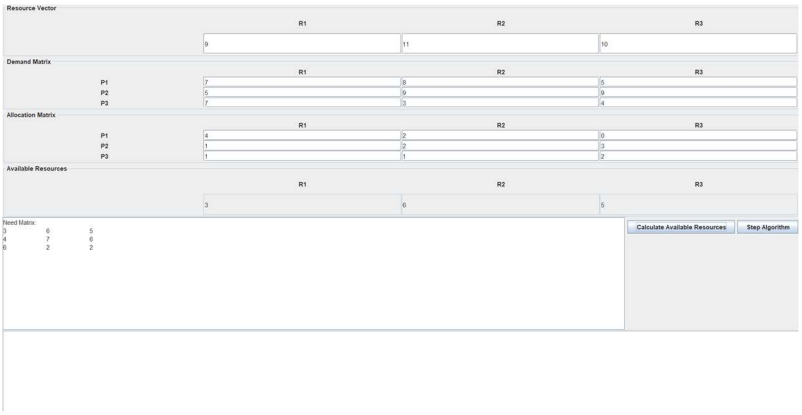



Con esto, obtenemos los recursos disponibles y la matriz resultante.

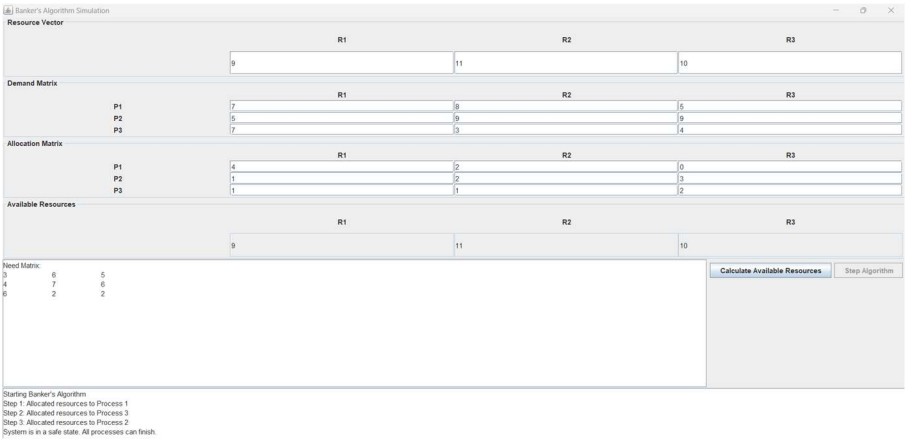
Se da clic en el botón de los pasos de algoritmo y para estos datos se alcanzó un deadblock, por lo que se detiene la ejecución y la asignación deberá cambiar para poder proceder con el algoritmo.




Se cambia la matriz de asignación.



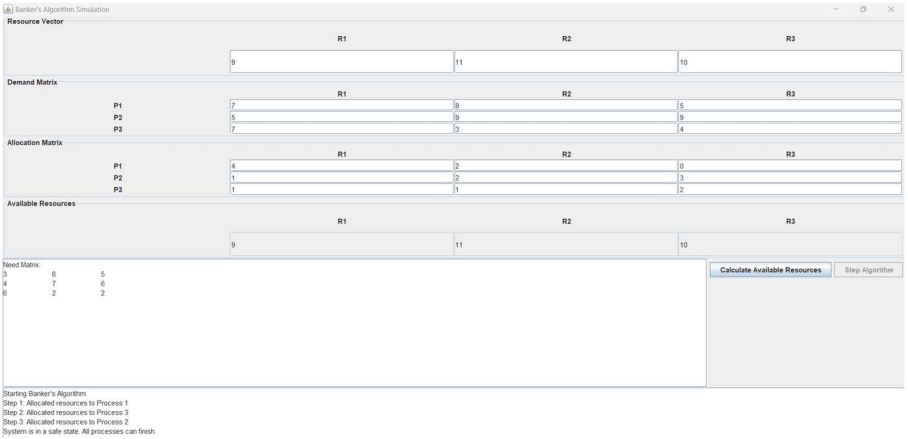
Volvemos a ejecutar los pasos de algoritmo, hasta el fin del algoritmo del algoritmo. Durante el proceso, los recursos disponibles se van a ir liberando y aumentando hasta completar los procesos.



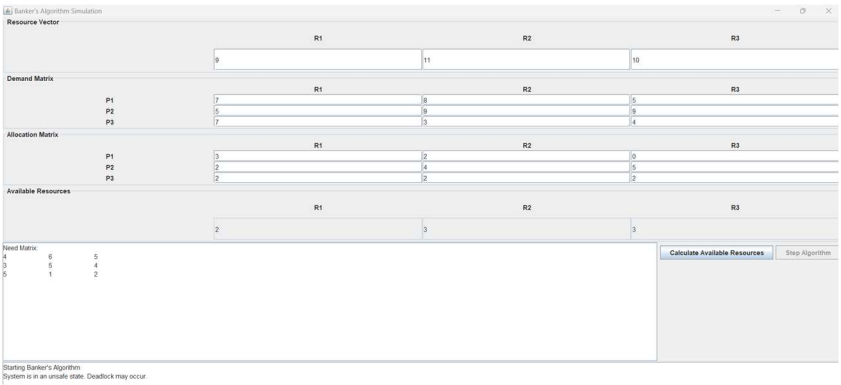

## Salidas del programa

Salidas normales de ejecución:

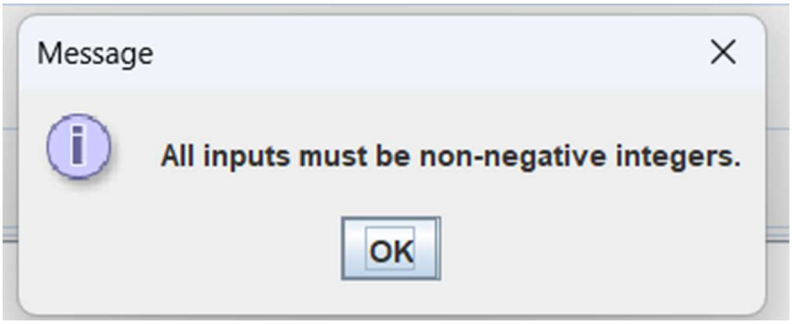
Sistema seguro, es cuando los datos se ingresaron de forma correcto con la validación de datos. Este sucede luego de validar los datos y pasarlos por el algoritmo y llegar a un estado seguro.



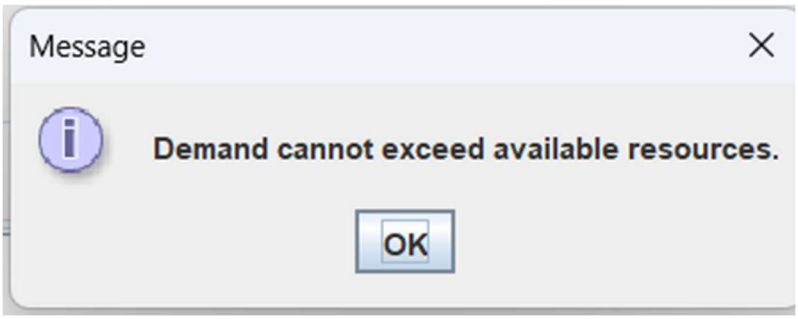

Deadblock, es cuando los datos se ingresaron de forma correcto con la validación de datos. Este sucede luego de validar los datos ingresados y que se llegue a la terminación por deadblock.



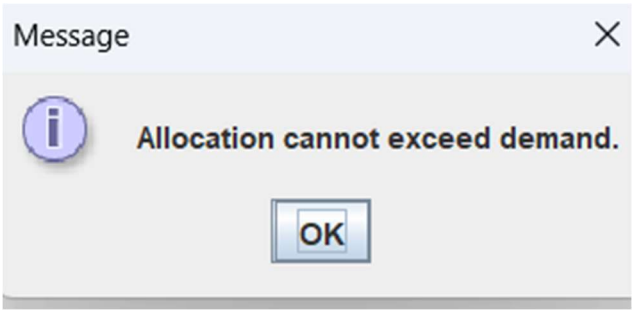

Tipo de datos no permitido, sucede que al intentar datos en alguna casilla no sean datos de tipo numérico entero positivo.




Datos de Necesidad mayor a la cantidad de posibles recursos de un tipo, sucede cuando algún dato de la matriz necesidad supera la cantidad del tipo de recurso del vector de recursos.




Datos de Asignación son mayores a los datos de demanda, esto no es correcto ya que nunca debe darse más recursos de lo que el proceso demanda.



## Parte 5. Limitaciones

La única limitación que tiene el programa es que la simulación esta únicamente para programa con 3 recursos y 3 procesos, quedando limitada a un 3x3. Aun así, es funcional y sirve para mostrar de manera simple a aquellas personas que van aprendiendo del tema.

## Conclusión

Ya, por último, se darán las conclusiones. Pero antes de ello, me gustaría agradecer a todos los profesores, tutores y asistentes que hicieron este curso posible, los cuales integran parte importante de la comunican y ayudan que se formen los profesionales que el país necesita y también brindan las oportunidades a no quedarse encerrados en una compañía, sino también la posibilidad de tener el suficiente criterio técnico para satisfacer propias y ajenas necesidades, por ello y muchas cosas más mil gracias.

Con respecto al documento presentado, al realizar dicho documento se logra entender de lo que es un interbloqueo y que problemas puede conllevar una mala asignación por parte de los sistemas operativos, pero como en un buen algoritmo, esto puede ser protegido y salvaguardando muchos problemas de latencia o errores dentro de los resultados.

Durante la realización de la simulación del algoritmo del banquero, se debió del comprender del porqué del nombre, donde para poder prestado siempre es necesario saber de los recursos que tenemos y no dar por sentado que en todo momento estarán a disposición, pero con una buena administración es posible dar solución a los procesos.

Por último, aunque sea breve al desarrollar una mini aplicación que simule el algoritmo del banquero, está preparando al estudiante de lo que está por venir en su trabajo, donde además de desarrollar, debe aprender del problema, proveer posibles soluciones, consultar otras alternativas, generar la documentación y para finalizar, hacer las pruebas para darle fin a lo que inicio como una idea y plasmarla en algo físico o digital, es lo que nos hace diferente a los ingenieros del resto de carreras.

# Referencias

Baylon, K. (2012). Algoritmo del Banquero. Extraído de: https://es.slideshare.net/slideshow/algoritmo-del-baquero/12084000#1

Jechaiz (2013). Algoritmo del Banquero. Extraído de: https://cs.uns.edu.ar/~jechaiz/sosd/clases/extras/Interbloqueos-AlgBanquero.pdf

*Tanembaum, A. (2009). Sistemas Operativos Modernos. Tercera edición. Editorial Pearson Educación. México.*