



# TEC

---

Tecnológico de Costa Rica

# Especificación de Requerimientos de Software Simplex Educativo

Profesora: Maria Estrada.

Curso: Proyecto de Ingeniería de Software.

10.01.2017

---

Jose Fernando Molina Chacón

Yordan Jiménez Hernández

# Cambios

Descripción	Versión	Fecha
Versión Inicial	0.1	6/01/17
Agregados casos de uso de iteración 2.	0.2	11/01/17

# Índice

<b>Índice</b>	<b>1</b>
<b>Introducción</b>	<b>2</b>
<b>Descripción General</b>	<b>4</b>
<b>Características del Sistema</b>	<b>6</b>
<b>Requerimientos de datos</b>	<b>10</b>
<b>Requerimientos de interfaces externas.</b>	<b>11</b>
<b>Atributos de calidad</b>	<b>12</b>

# 1. Introducción

## 1.1. Propósito

El objetivo de este documento es establecer, explicar y definir los requerimientos funcionales y no funcionales para el software Simplex Educativo. En él se listan las características del sistema junto con los requerimientos que describen los casos de uso. Los interesados en el siguiente escrito son los miembros del equipo de desarrollo, el cliente Jose Elias Helo, quien será el encargado de aprobarlo, así como futuras personas que deseen darle mantenimiento o extender la solución. Cualquier requerimiento o característica que no sea mencionada en este documento quedará fuera de la presentación final del proyecto para aprobación del cliente.

## 1.2. Convenciones en el documento

El orden en que se describen las características y los requerimientos del sistema no indica el grado de importancia que estos tienen en el proyecto. Todos los requerimientos de la aplicación tienen la misma prioridad para el equipo desarrollador. El documento está ordenado de acuerdo a la plantilla dispuesta por Karl Wiegers, tratando de que el lector avance de manera gradual a lo largo del mismo dándole los conceptos necesarios en cada sección para que avance la siguiente. Se definen los siguientes acrónimos para listar los distintos tipos de requerimientos:

- **REQ:** Requerimiento Funcional.
- **AO:** Ambiente Operativo.
- **RES:** Restricción.
- **AS:** Asunciones y Dependencias.
- **IU:** Interfaces de Usuario.

## 1.3. Audiencia y sugerencias de lectura

Este documento está escrito para ser leído y aprobado por el cliente de la aplicación a ser desarrollada, es decir, el profesor José Elías Helo, sin embargo también puede ser visto por personas con interés en ampliar los conocimientos sobre el proyecto por desarrollar sin necesidad de poseer conocimientos profundos en computación. Esta especificación de requerimientos de software también está dirigida para los programadores y desarrolladores de la solución, aquí encontrarán todos los requerimientos que el sistema debe cumplir, restricciones o limitaciones

que se deban tomar en cuenta a la hora del desarrollo de la solución, tanto a nivel funcional como no funcional: interfaz de usuario, rendimiento o calidad, para mencionar algunos.

Se recomienda leer este documento de manera lineal, de manera que el lector pueda comprender los varios aspectos que deben ser abordados por la solución. En la sección 2 se describe el producto de manera general, así que los lectores ya familiarizados con él pueden no leer esta sección.

Es de especial atención el apartado 3 (Características del sistema) que contiene los requerimientos principales con los que debe cumplir la aplicación. Es el principal punto de todo el documento pues menciona con detalle el comportamiento que va a tener la aplicación. Las secciones 4 y 5 describen requerimientos de formato y necesidades de las interfaces gráficas.

#### 1.4. Alcance del proyecto

El proyecto incluye el desarrollo de una aplicación para escritorio que ayude al profesor PhD. José Elías Helo a desarrollar su clase de Investigación de Operaciones de una manera más dinámica que propicie el entendimiento de la materia por parte del estudiante. Una vez desarrollada, esta aplicación será utilizada por el profesor solamente y será propiedad total y exclusiva de él, incluido el código fuente del programa.

El proyecto incluye documentación técnica en forma de comentarios en el código fuente, manual de usuario y documentación externa. También incluye una serie de artefactos de QA como pruebas unitarias y de integración, así como pruebas de sistema y pruebas de aceptación que serán realizadas por el cliente.

El proyecto no incluye el mantenimiento de la aplicación una vez entregado el producto. Cuando se haga la entrega final del proyecto, el cliente se hará responsable del producto de ese momento en adelante y el equipo de desarrollo ya no será responsable del mismo.

La construcción del proyecto está estimada en un plazo no mayor a 35 días. La aplicación podrá ser ejecutada en distintas plataformas ya sea Windows, Mac o Linux. Para ello se desarrollará en el lenguaje de programación Java, el cual es multiplataforma y portable.

#### 1.5. Referencias

1. Documento de Visión de Simplex Educativo.
2. Estándar de código de java. Disponible en:  
<https://google.github.io/styleguide/javaguide.html>

## 2. Descripción General

### 2.1. Perspectiva del producto

Durante el transcurso del curso de Investigación de Operaciones de la carrera de Ingeniería en Computación impartida en el Instituto Tecnológico de Costa Rica, se abarcan temas de maximización y minimización de costos, métodos matemáticos que pueden ser computarizados para su utilización en diversos campos. En la educación, un método utilizado en el curso nombrado anteriormente es el método Simplex. Ya existen en el mercado y de manera gratuita software como Lindo o páginas web como PhpSimplex que proveen un programa para solucionar dichos problemas.

Sin embargo, dichos programas poseen limitaciones a la hora de utilizarlos como herramienta educativa. Algunos de ellos solamente solucionan el problema, sin dar información que puede ser relevante para el profesor o estudiantes. Por otra parte, cada software representa los problemas de programación lineal por medio de una matriz con un formato diferente, por lo que se vuelve difícil interpretar los resultados o datos provistos por estas soluciones. Es por este motivos que para la enseñanza en el campus, estas implementaciones existentes no poseen las características que faciliten la enseñanza por parte del profesor encargado del curso.

Dichas carentes funcionalidades son importantes para facilitar la enseñanza son las que se buscan proveer mediante la elaboración del presente proyecto de software. El proyecto se centra en la implementación de un programa que solucione problemas del algoritmo simplex de una manera dinámica y amigable para el aprendizaje, donde el usuario puede indicar distintas formas de solucionar un ejemplo de maximización o minimización, con el fin que el usuario pueda resaltar características relevantes a los estudiantes acerca de comportamientos que tome el algoritmo simplex ante los diferentes escenarios posibles.

### 2.2. Clases de usuario y características

Clase de usuario	Descripción
Usuario Final	El usuario final es el profesor José Elías Helo del curso de Investigación de Operaciones o alguno de sus estudiantes que necesite resolver un problema de programación lineal bajo los diferentes métodos que se enseñan en dicha clase, y utilizar la solución ya sea para la enseñanza o el aprendizaje de los métodos matemáticos contenidos

	enseñados en la universidad.
--	------------------------------

## 2.3. Ambiente de operación

### 2.3.1. AO-1: Ambiente Operativo.

**Descripción:** Simplex Educativo podrá ser ejecutado en cualquier sistema operativo donde se ejecute la máquina virtual de Java con una versión superior al 8.0.

## 2.4. Restricciones de diseño e implementación

### 2.4.1. RES-1: Estándar de Código.

**Descripción:** El código desarrollado durante la elaboración de este sistema deberá seguir el estándar de Google para el desarrollo en Java.

### 2.4.2. RES-2: Librería de Interfaz Gráfica.

**Descripción:** Los objetos de la interfaz gráfica de la aplicación se elaborarán por medio de la librería Java Swing.

## 2.5. Asunciones y dependencias

### 2.5.1. AS-1: Notación Simplex.

**Descripción:** El usuario posee conocimientos sobre la notación utilizada por el profesor José Helo para la descripción de problemas de programación lineal.

### 2.5.2. AS-2: Conocimiento del algoritmo Simplex.

**Descripción:** El usuario conoce el comportamiento del algoritmo simplex, la manera en que se comporta de acuerdo a las entradas y los posibles resultados que pueda exhibir un problema.

### 2.5.3. AS-3: Ejecución temporal y eficiencia.

**Descripción:** El algoritmo de solución simplex no dependerá de alguna métrica de eficiencia de ejecución o de tiempo.

### 2.5.4. AS-4: Ambiente de ejecución del programa.

**Descripción:** El equipo que ejecutará la aplicación de escritorio, deberá tener la capacidad de ejecutar la Máquina Virtual de Java (JVM) en su versión 8.0 o superior.

### 3. Características del Sistema

#### 3.1. Verificar la factibilidad de un problema de programación lineal ingresado.

##### 3.1.1. Descripción

El sistema debe poseer la capacidad para determinar si un problema de programación lineal posee una solución factible óptima.

##### 3.1.2. Prioridad

Alta

##### 3.1.3. Secuencia Estímulo/Respuesta

1. El sistema muestra el apartado donde se ingresa el problema de programación lineal.
2. El usuario ingresa el problema de programación lineal por medio de una cadena de texto.
3. El usuario escoge el formato de salida del programa.
4. El usuario escoge el modo de solución directa.
5. El usuario selecciona la opción de solucionar un problema.
6. El sistema muestra por medio de un mensaje si el problema es infactible y los pasos realizados.

##### 3.1.4. Requerimientos Funcionales

###### REQ-1: Infactibilidad de un problema

El sistema debe de ser capaz de interpretar el problema de programación lineal ingresado por el usuario e indicar si las características presentes en su contenido por medio de restricciones, variables u otros, hacen que el problema no contenga una solución factible óptima. Dicho resultado es mostrado por medio de un mensaje de información y la lista de pasos realizados.

#### 3.2. Obtener de manera inmediata una solución óptima de un problema de programación lineal.

##### 3.2.1. Descripción

El sistema será capaz de solucionar un problema de programación lineal por medio del método Simplex brindando inmediatamente la solución al usuario.

##### 3.2.2. Prioridad



Alta

### **3.2.3. Secuencia Estímulo/Respuesta**

1. El sistema muestra el apartado donde se ingresa el problema de programación lineal.
2. El usuario ingresa el problema de programación lineal por medio de una cadena de texto.
3. El usuario escoge el formato numérico en el cual desea la respuesta.
4. El usuario selecciona la opción de solución inmediata.
5. El usuario selecciona que desea resolver el problema por medio del método Simplex.
6. El sistema interpreta el problema ingresado y obtiene la primera solución óptima que genere el método Simplex.
7. El sistema muestra el resultado óptimo final del problema.

### **3.2.4. Requerimientos Funcionales**

#### **REQ-2** Interpretar un problema ingresado:

El sistema debe interpretar dentro del problema de programación lineal ingresado, los valores enteros, variables y restricciones que simboliza la cadena de texto ingresado por el usuario, en caso de que no contenga el formato esperado, el sistema le indicará al usuario por medio de un mensaje la naturaleza del error.

#### **REQ-3** Solución por medio del Simplex:

El sistema debe utilizar el método Simplex para obtener la primera solución óptima de un problema de programación lineal.

#### **REQ-4** Resultados fraccionarios o decimales:

Los valores numéricos brindados por la ejecución del sistema deben ser mostrados en formato decimal o fraccionario. Dicho formato el usuario podrá seleccionarlo antes de ejecutar la solución del problema.

## **3.3. Listar los pasos intermedios para encontrar la solución a un problema de programación lineal.**

### **3.3.1. Descripción**

El sistema será capaz de solucionar un problema de programación lineal por medio del método Simplex donde se pueda generar las tablas o matrices intermedias que se produjeron para obtener la primera solución óptima del problema ingresado.

### **3.3.2. Prioridad**

Alta.

### **3.3.3. Secuencia Estímulo/Respuesta**

1. El sistema muestra el apartado donde se ingresa el problema de programación lineal.
2. El usuario ingresa el problema de programación lineal por medio de una cadena de texto.
3. El usuario escoge el formato numérico en el cual desea la respuesta.
4. El usuario selecciona la opción de solución por pasos.
5. El usuario selecciona que desea resolver el problema por medio del método Simplex.
6. Sistema muestra uno a uno cada iteración necesaria para obtener la solución óptima. El usuario avanza manualmente mediante un botón entre las iteraciones.
7. El sistema indica y muestra que se ha obtenido la solución final óptima.

### **3.3.4. Requerimientos Funcionales**

**REQ-5** Iteraciones intermedias de la solución de un problema de programación lineal

El sistema será capaz de interpretar las iteraciones intermedias que se generaron a la hora de obtener una solución óptima de un problema de programación lineal. En el transcurso de dicho proceso se indicará las características de la iteración, ya sea las variables de holgura, artificiales o las variables a las cuales se intenta obtener la solución óptima, mostrando los valores que contienen, así como información sobre el estado del problema en cada paso .

## **3.4. Verificar si un problema de programación lineal ingresado se encuentra acotado.**

### **3.4.1. Descripción**

El sistema debe poseer la capacidad para determinar si un problema de programación lineal se encuentra acotado dentro de un rango válido.

### **3.4.2. Prioridad**

Alta

### **3.4.3. Secuencia Estímulo/Respuesta**

1. El sistema muestra el apartado donde se ingresa el problema de programación lineal.
2. El usuario ingresa el problema de programación lineal por medio de una cadena de texto.
3. El usuario escoge el formato numérico en el cual desea la respuesta.
4. El usuario selecciona la opción de solución inmediata.
5. El usuario selecciona que desea resolver un problema de programación lineal.
6. El sistema muestra por medio de un mensaje si el problema no se encuentra acotado.

#### **3.4.4. Requerimientos Funcionales**

##### **REQ-6 Problema No Acotado:**

El sistema debe de ser capaz de interpretar el problema de programación lineal ingresado por el usuario e indicar si las características presentes en su contenido por medio de restricciones, variables u otros, hacen que el problema no contenga una solución acotada. Dicho resultado es mostrado por medio de un mensaje de advertencia en caso que el problema no se encuentre acotado.

### **3.5. Ingresar una matriz para realizar operaciones**

#### **3.5.1. Descripción**

Debe ser posible ingresar una matriz de  $N \times M$  números y realizar pivoteos en dicha matriz. También deben mostrarse los radios entre el lado derecho y la columna que está seleccionada en cierto momento. El formato de dicha matriz puede ser el utilizado por el programa para mostrar los pasos intermedios o solamente una matriz numérica con las entradas separadas por espacios y cambios de línea.

#### **3.5.2. Prioridad**

Alta

#### **3.5.3. Secuencia Estímulo/Respuesta**

1. El sistema muestra el apartado donde se ingresa el problema de programación lineal.
2. El usuario ingresa la cadena de texto que representa la matriz en el apartado correspondiente.
3. El usuario selecciona que desea realizar operaciones a una matriz numérica.
4. El sistema muestra la representación de la matriz.
5. El usuario escoge la casilla donde desea hacer el pivoteo.

6. El usuario realiza el pivoteo.

#### **3.5.4. Requerimientos Funcionales**

##### **REQ-7** Interpretar matriz ingresada:

El sistema debe de ser capaz de interpretar la matriz numérica ingresada por el usuario e indicar si las características presentes en su contenido hacen que la matriz tenga alguna información o dato inválido. En caso de que exista alguna inconsistencia o el formato sea inválido, el sistema responderá con un mensaje de información que clarifique el error.

##### **REQ-8** Realizar pivoteos en las matrices intermedias:

Ya sea por el ingreso de un problema de programación lineal o por el ingreso de una matriz, debe ser posible para el usuario elegir la casilla donde desee realizar el siguiente pivoteo.

- 3.6. Obtener de manera inmediata la solución entera de un problema de programación lineal mediante el algoritmo de Gomory.

#### **3.6.1. Descripción**

Debe ser posible encontrar la solución entera óptima de un problema de programación lineal por medio del algoritmo de cortes de gomory de manera inmediata. Dicho algoritmo agrega restricciones una vez finalizado el simplex común para ir eliminando las soluciones que no son enteras, hasta el momento que se llegue a un óptimo entero, junto con sus variables básicas, las cuales deben ser enteras también.

#### **3.6.2. Prioridad**

Alta

#### **3.6.3. Secuencia Estímulo/Respuesta**

1. El sistema muestra el apartado donde se ingresa el problema de programación lineal.
2. El usuario ingresa el problema de programación lineal por medio de una cadena de texto.
3. El usuario escoge el formato numérico en el cual desea la respuesta.
4. El usuario selecciona que desea resolver el problema por medio del método Cortes de Gomory.
5. El usuario selecciona la opción de solución directa.

6. El sistema interpreta el problema ingresado y obtiene la primera solución óptima que genere el método Cortes de Gomory.
7. El sistema muestra el resultado óptimo final del problema.

#### **3.6.4. Requerimientos Funcionales**

**REQ-9** Obtener solución entera directa por medio del algoritmo de Gomory:

El sistema debe de ser capaz de obtener la solución óptima entera de manera directa, sin listar los pasos intermedios, de un problema de programación lineal ingresado por medio del algoritmo de Gomory.

### **3.7. Listar los pasos intermedios para encontrar la solución de un problema de programación lineal mediante el algoritmo de Gomory**

#### **3.7.1. Descripción**

Debe ser posible listar los pasos intermedios para encontrar la solución entera óptima de un problema de programación lineal por medio del algoritmo de cortes de gomory. Dicho algoritmo agrega restricciones una vez finalizado el simplex común para ir eliminando las soluciones que no son enteras, hasta el momento que se llegue a un óptimo entero, junto con sus variables básicas, las cuales deben ser enteras también.

#### **3.7.2. Prioridad**

Alta

#### **3.7.3. Secuencia Estímulo/Respuesta**

1. El sistema muestra el apartado donde se ingresa el problema de programación lineal.
2. El usuario ingresa el problema de programación lineal por medio de una cadena de texto.
3. El usuario escoge el formato numérico en el cual desea la respuesta.
4. El usuario selecciona que desea resolver el problema por medio del método Cortes de Gomory.
5. El usuario selecciona la opción de solución por pasos.
6. El sistema interpreta el problema ingresado y muestra la primera tabla del problema.
7. El usuario indica que desea conocer el siguiente paso.
8. El sistema muestra el siguiente paso.

- a. Si existe otro paso, volver a 7.
9. El sistema muestra la solución óptima entera encontrada.

#### **3.7.4. Requerimientos Funcionales**

**REQ-10** Obtener los pasos intermedios para resolver un problema de programación lineal por medio del algoritmo de Gomory

El sistema debe de ser capaz de obtener los pasos intermedios para encontrar la solución óptima entera de un problema de programación lineal ingresado por medio del algoritmo de Gomory. El sistema irá agregando cortes al final de la última tabla encontrado en caso de que la solución óptima no sea entera. No se optimizará desde el origen.

### **3.8. Obtener de manera inmediata una solución entera de un problema de programación lineal mediante el algoritmo de Branch and Bound.**

#### **3.8.1. Descripción**

Debe ser posible encontrar la solución entera óptima de un problema de programación lineal por medio del algoritmo de Branch and Bound de manera inmediata. Dicho algoritmo resuelve N cantidad de subproblemas simplex agregando restricciones según sea necesario en un paso dado, llegando a acotar la solución en algún momento al valor óptimo válido.

#### **3.8.2. Prioridad**

Alta

#### **3.8.3. Secuencia Estímulo/Respuesta**

1. El sistema muestra el apartado donde se ingresa el problema de programación lineal.
2. El usuario ingresa el problema de programación lineal por medio de una cadena de texto.
3. El usuario escoge el formato numérico en el cual desea la respuesta.
4. El usuario selecciona que desea resolver el problema por medio del método Branch and Bound.
5. El usuario selecciona la opción de listar los pasos intermedios.
6. El sistema interpreta el problema ingresado y obtiene la primera solución óptima que genere el método Branch and Bound.
7. El sistema muestra el resultado óptimo final del problema.

#### **3.8.4. Requerimientos Funcionales**

**REQ-11** Obtener solución entera directa por medio del algoritmo de Branch and Bound:

El sistema debe de ser capaz de obtener la solución óptima entera de manera directa, sin listar los pasos intermedios, de un problema de programación lineal ingresado por medio del algoritmo de Branch and Bound.

### 3.9. Listar los pasos intermedios para encontrar la solución de un problema de programación lineal mediante el algoritmo de Branch and Bound.

#### 3.9.1. Descripción

Debe ser posible listar los pasos intermedios para encontrar la solución entera óptima de un problema de programación lineal por medio del algoritmo de Branch and Bound. Dicho algoritmo resuelve N cantidad de subproblemas simplex agregando restricciones según sea necesario en un paso dado, llegando a acotar la solución en algún momento al valor óptimo válido.

#### 3.9.2. Prioridad

Alta

#### 3.9.3. Secuencia Estímulo/Respuesta

1. El sistema muestra el apartado donde se ingresa el problema de programación lineal.
2. El usuario ingresa el problema de programación lineal por medio de una cadena de texto.
3. El usuario escoge el formato numérico en el cual desea la respuesta.
4. El usuario selecciona que desea resolver el problema por medio del método Branch and Bound.
5. El usuario selecciona la opción de solución por pasos.
6. El sistema interpreta el problema ingresado y muestra la primera tabla del problema.
7. El usuario indica que desea conocer el siguiente paso.
8. El sistema muestra el siguiente paso.
  - a. Si existe otro paso, volver a 7.
9. El sistema muestra la solución óptima entera encontrada.

#### 3.9.4. Requerimientos Funcionales

**REQ-12** Obtener los pasos intermedios para resolver un problema de programación lineal por medio del algoritmo de Branch and Bound.

El sistema debe de ser capaz de obtener los pasos intermedios para encontrar la solución óptima entera de un problema de programación lineal ingresado por medio del algoritmo de Branch and Bound. El sistema irá desplegando a manera de árbol de texto las soluciones a los subproblemas encontrados solamente.

### 3.10. Agregar restricciones a un problema de programación lineal.

#### 3.10.1. Descripción

Debe ser posible, una vez iniciado el programa, agregar restricciones a la matriz que representa el programa de programación lineal. Dichas restricciones pueden ser de tipo mayor igual, menor igual o igual. En el caso de menor o igual, se agregará una nueva variable de holgura. En el caso de igual, se agregará una nueva variable artificial. En el caso de mayor o igual, se agregará una variable negativa de holgura y una artificial.

#### 3.10.2. Prioridad

Alta

#### 3.10.3. Secuencia Estímulo/Respuesta

1. El sistema muestra el apartado donde se ingresa el problema de programación lineal.
2. El usuario ingresa el problema de programación lineal por medio de una cadena de texto.
3. El usuario escoge el formato numérico en el cual desea la respuesta.
4. El usuario selecciona que desea resolver el problema por medio del método Simplex.
5. El usuario selecciona la opción de solución por pasos.
6. El sistema interpreta el problema ingresado y muestra la primera tabla del problema.
7. El usuario indica que desea agregar una restricción.
8. El usuario indica el tipo de restricción.
9. El sistema agrega la restricción adecuada.

#### 3.10.4. Requerimientos Funcionales



**REQ-13** Agregar una restricción a un problema de programación lineal.

Una vez representado el problema en forma matricial, debe ser posible para el usuario durante el primer paso de la solución agregar restricciones mayor igual, menor igual o igual a la matriz mostrada en pantalla. Los casos igual y mayor o igual solamente serán válidos para problemas de dos fases.

## 4. Requerimientos de datos

### 4.1. Formatos de Entrada

#### 4.1.1. Formato de entrada de problema Simplex

**Descripción:** El formato de entrada de problemas de programación lineal será el utilizado por el profesor José Helo durante su curso de programación lineal. La descripción se realizará con el siguiente ejemplo:

$$(0) \max z = 10 x_1 + 15 x_2 + 5 x_3$$

$$(1) 2 x_1 + 1 x_2 \leq 6000$$

$$(2) 3 x_1 + 3 x_2 + 1 x_3 \leq 9000$$

$$(3) 1 x_1 + 2 x_2 + 2 x_3 \leq 4000$$

Los números de restricción marcados con color rojo son opcionales. La primera línea contendrá la función objetivo. Si el problema es de maximización o minimización dependerá de la primera palabra en la primera línea, la cual puede ser “max” o “min”. Seguidamente se espera el nombre de la variable objetivo, un signo de igual (=) y el valor de la función. El formato de las restricciones y función objetivo será coeficiente primero y nombre de la variable después. No se tomarán en cuenta los espacios en blanco entre coeficiente y variable o entre operador y variable.

La segunda y subsiguientes líneas serán las restricciones del problema. Las restricciones tienen una de las siguientes formas:

$$c_1 x_1 + c_2 x_2 + \dots + c_n x_n \leq b(1)$$

$$c_1 x_1 + c_2 x_2 + \dots + c_n x_n \geq b(1)$$

$$c_1 x_1 + c_2 x_2 + \dots + c_n x_n = b(1)$$

donde  $c_1, c_2, \dots, c_n$  son números válidos enteros, fraccionarios o punto flotante. Es importante aclarar que el programa no realizará simplificación de la expresión. Por ejemplo,  $2 x_1 + 3 x_1$  no será

simplificado a 5 x1, sino que será tomado como si existieran dos variables x1 diferentes.

#### 4.1.2. Formato de entrada de matriz numérica

**Descripción:** El formato de entrada de matriz numérica puede tener el siguiente formato: N números en M líneas, por ejemplo:

```
0 1 2 3
4 5 6 7
8 9 0 1
```

Los números aceptados serán números enteros, punto flotante o fracciones. La cantidad de espacios entre los números no tiene importancia siempre que haya al menos uno para separar las diferentes entradas.

## 5. Requerimientos de interfaces externas.

### 5.1. Interfaces de usuario

#### 5.1.1. IU-1: Características generales

**Descripción:** El sistema Simplex Educativo poseerá la estructura de una aplicación de escritorio, con botones y entradas de texto brindados por la librería de texto Java swing. El tamaño de las ventanas serán automáticos según su contenido o a preferencia del usuario.

#### 5.1.2. IU-2: Colores para resaltar operación.

**Descripción:** El sistema debe mostrar por medio de colores, durante los pasos intermedios, la fila y columna donde se realizará el próximo pivoteo, así como marcar la casilla que ha sido elegida como pivote.

#### 5.1.3. IU-3: Siguiete operación a realizar.

**Descripción:** El sistema debe mostrar las siguientes operaciones fila que realizará dependiendo de la casilla pivote que esté seleccionada en cierto momento.

#### 5.1.4. IU-3: Mostrar radios.

**Descripción:** El sistema debe mostrar los radios del lado derecho con las casillas de la columna seleccionada en ese momento.

### 5.2. Interfaces de Software

No se ha identificado ninguna interfaz de software.

### 5.3. Interfaces de Hardware

No se ha identificado ninguna interfaz de hardware.

## 6. Atributos de calidad

Estado	Definición
Usabilidad	<ol style="list-style-type: none"><li>1. El sistema Simplex Educativo permitirá al usuario solucionar un problema de programación lineal al usuario por medio de pasos manuales o de manera inmediata.</li><li>2. Los nuevos usuarios podrán encontrar en el manual de usuario la forma de ejecutar todas las funcionalidades provistas por el sistema.</li><li>3. Los usuarios deben comprender el significado de los mensajes de error o información lanzados cuando no se pueda ejecutar el el algoritmo o suceda alguna situación extraordinaria.</li><li>4. La interfaz de usuario estará diseñada de una manera sencilla y utilizando colores que ayuden a resaltar la funcionalidad.</li></ol>
Rendimiento	El proyecto que se encuentra en elaboración no contendrá ninguna métrica de rendimiento o eficiencia debido a que no es un punto central o prioritario establecido por el cliente.
Portabilidad	<ol style="list-style-type: none"><li>1. El sistema se podrá ejecutar en distintos sistemas operativos ya sea Windows, Mac o Linux, donde su ejecución y comportamiento no sea dependiente de la plataforma donde se esté ejecutando.</li><li>2. Para el desarrollo de la solución se utilizará el IDE NetBeans, sin embargo, se espera que la utilización de otra herramienta de desarrollo sea posible. No se utilizarán plugins propios del ambiente de desarrollo y se espera que el proyecto no sea dependiente al programa utilizado para su desarrollo.</li></ol>
Mantenibilidad	<ol style="list-style-type: none"><li>1. Todo el código escrito por el equipo de desarrollo estará comentado utilizando comentarios estilo Javadocs,</li></ol>

	<p>además de comentarios tradicionales donde se considere pertinente.</p> <ol style="list-style-type: none"> <li>2. Para claridad del código, se utilizará el estándar de código de Google para Java.</li> <li>3. Se desarrollarán documentos técnicos donde se describa detalladamente el funcionamiento del sistema.</li> <li>4. Se generará documentación en formato HTML mediante la herramienta Doxygen.</li> </ol>
Confiabilidad	<ol style="list-style-type: none"> <li>1. El sistema debe brindar soluciones veraces y válidas para los problemas ingresados por el usuario.</li> <li>2. Los mensajes lanzados por el sistema deben solamente suceder en los casos correctos.</li> </ol>

PhD. Jose Elías Helo  
***Cliente***

Yordan Jiménez Hernández  
***Equipo de Desarrollo***

Fernando Molina Chacón  
***Equipo de Desarrollo***