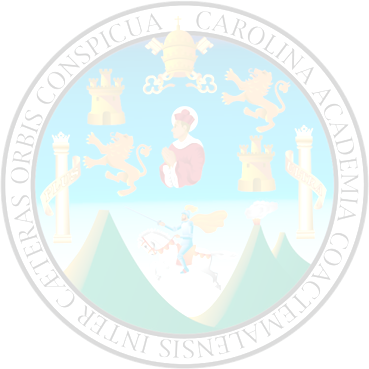
UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

ESTRUCTURA DE DATOS

ING. JESÚS GUZMÁN

AUX. WALTER MACH



**MANUAL TÉCNICO**

201801295

JOSÉ RODRIGO GARCIA GODÍNEZ

GUATEMALA, 20 DE FEBRERO DE 2022

# OBJETIVOS

## MANUAL

* Presentar todos los componentes, métodos, rutas, funciones y objetos utilizados para la realización del programa, para brindar una mejor solución al momento de realizar una modificación, optimizar o hacer mantenimiento de la aplicación

## APLICACIÓN

* Proporcionar a la empresa “Drawing Paper” una solución viable al atender a sus clientes a la hora de proporcionar el servicio de impresión de imágenes a color y blanco/negro. Esto realizando una ayuda por medio de la librería gráfica de Graphviz y ver el estado de las estructuras implementas en la todos los almacenamientos de datos y ver que imágenes y clientes estén asignados en las ventanillas en uso y en las impresoras que se estén manejando.

# REQUERIMIENTOS MÍNIMOS DEL ENTORNO DE DESARROLLO

* JAVA JDK: versión 17.0.1
* IDE :
  + Netbeans
  + Eclipse
  + Visual Estudio Code
  + IntelliJ IDEA
  + Preferencia
* SISTEMA OPERATIVO: Windows 10 Pro
* FRAMEWORKS O LIBRERIAS:
  + Graphviz: 2.50.0
  + Json-simple

# MÓDULOS

## OBJETOS

Clases implementadas en el sistema para la ayuda de encapsular los datos y poner tener un mejor manejo en el sistema, y poder tener una carga de datos rápida en el sistema con la implementación de cargas masivas con la ayuda de archivos JSON.

### RESUMEN DE CLASES

| Nombre | Tipo | Encapsulamiento | Descripción/Funcionamiento |
| --- | --- | --- | --- |
| Busqueda | Clase | Publico | Resumen de datos para la implementación de ordenamiento rápido en todas estructuras utilizadas en el sistema. |
| Cliente | Clase | Publico | Datos de todos los clientes ingresados en el sistema, dando un resumen de los datos que se necesiten operar en el sistema. |
| Dato | Clase | Publico | Clase capturadora de datos en el sistema por medio de BufferReader. |
| Estructura | Clase | Publico | Resumen de estados de todas las estructuras utilizadas. Guardar la información del momento precisó en que se use una estructura o se registre un cambio en el sistema. |
| Imagen | Clase | Publico | Imagen registrada para la impresora. |
| Impresora | Clase | Publico | Dispositivo que lleva el control de todas las imágenes en el sistema, tanto de color, como en blanco y negro. |
| Ventanilla | Clase | Publico | Segmentos habilitados para la atención del cliente entrante en el sistema. |
| ESTADO | Enumerada | Publico | Estados de las impresoras, clientes, imágenes y ventanillas en todos los sistemas cuando se inicie la simulación con los datos. |
| TIPO | Enumerada | Publico | Clasificación de las imágenes según el tipo de imagen que se requiera imprimir y asignar con facilidad a la impresora designada con la misma clase. |

### FUNCIONALIDAD (CLASES)

#### DATO

Clase creada para el manejo del ingreso de datos, capturando todos los datos escritos en aplicación. Con ello se dividió el uso entre dos clases:

* dato\_int(): usado para capturar los datos crear un tipo de dato int, devolviendo este dato para su manejo.
* dato\_string(): usado para capturar los datos y crear un tipo dato string, devolviendo este dato para su manejo.

## ESTRUCTURAS

Estructuras creadas para la implementación de listas dinámicas en el sistema descrito con anterioridad. Cada estructura usada es genérica para evitar el uso de una lista para cada tipo de dato utilizado.

### RESUMEN DE CLASES

| NOMBRE | TIPO | Descripción/Funcionamiento |
| --- | --- | --- |
| NodoCola | Nodo | Nodo principal en la mayoría de las estructuras creadas, adaptando la funcionalidad de la cola para reducir la implementación de deferentes nodos con una similitud de funcionalidad. |
| NodoDoble | Nodo | Nodo usado en la lista doblemente enlazada y circular. |
| Cola | Estructura | Estructura cola implementada en el sistema usando la definición de primero en entrar es el primero en salir. Esta estructura en una clase padre para algunas estructuras creadas. |
| Doblemente Enlazada | Estructura | Estructura doblemente enlazada implementada en el sistema. |
| Pila | Estructura | Estructura pila implementada en el sistema usando la definición de primero en entrar es el último en salir. |
| Simple | Estructura | Estructura que hereda las funciones y atributos de la estructura Cola, ya que su funcionalidad y estructura es similar. |

### FUNCIONALIDAD

Cada estructura cuenta con las funcionalidades siguientes:

#### COLA()

* PUSH: ingresa un dato a la parte inferior de la cola.
* VACIA(T): estado de la estructura conforme a los datos ingresados.
* POP(): elimina el primer dato ingresado.
* FRONT(): devuelve la cabeza de la estructura.
* TAMANIO(): devuelve el número de datos que están en la estructura.

#### DOBLEENLAZADA()

* AGREGAR(): ingresa un valor a la estructura.
* ELIMINAR(T): elimina el dato ingresado por medio de una comparación.
* VACIA(T): estado de la estructura conforme a los datos ingresados.
* TAMANIO(): devuelve el número de datos que están en la estructura.

PILA():

* PUSH: ingresa un dato a la parte inferior de la cola.
* VACIA(T): estado de la estructura conforme a los datos ingresados.
* POP(): elimina el primer dato ingresado.
* FRONT(): devuelve la cabeza de la estructura.
* TAMANIO(): devuelve el número de datos que están en la estructura.

SIMPLE():

* PUSH: ingresa un dato a la parte inferior de la cola.
* VACIA(T): estado de la estructura conforme a los datos ingresados.
* POP(): elimina el primer dato ingresado.
* FRONT(): devuelve la cabeza de la estructura.
* TAMANIO(): devuelve el número de datos que están en la estructura.
* ELIMINAR(T): elimina el dato ingresado por medio de una comparación.

## PRINCIPAL

Método y clase principal para ejecutar la simulación.

## CONTROLADOR

Controlador de la clase principal, donde está todo el manejo de los datos.

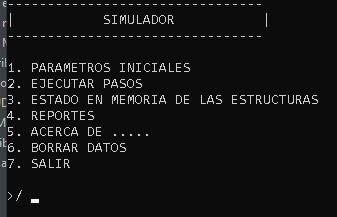
### RESUMEN DE MÉTODOS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NOMBRE | ENCAPSULADO | PARAMETRO | TIPO | DESCRIPCION/FUNCIONALIDAD |
| MENUINICIAL | PUBLICO |  | VOID | Método principal para la ejecución de la simulación en todo momento. |
| ASIGNARCLIENTE | PRIVADO | CLIENTE | BOOLEAN | Asignar un cliente a una ventanilla creada con anterioridad para poder ser atentado. |
| ASIGNARID | PRIVADO | TIPO | VOID | Enciende las impresoras para empezar a imprimir todas las imágenes mandadas por las ventanillas. |
| CARGARVENTANILLA | PRIVADO | INT | VOID | Carga las ventanillas al sistema. |
| EXTENSIONARCHIVO | PRIVADO | STRING | STRING | Devuelve la extensión del archivo que se requiera cargar en el sistema. |
| GRAFICARVENTANILLA | PRIVADO | SIMPLE | STRING | Devuelve el contenido del archivo con extensión dot de las ventanillas ingresadas. |
| GRAFICARBW | PRIVADO | COLA | STRING | Devuelve el contenido del archivo con extensión dot de las imágenes ingresadas en la impresora blanco/negro. |
| GRAFICARCOLOR | PRIVADO | COLA | STRING | Devuelve el contenido del archivo con extensión dot de las imágenes ingresadas en la impresora color. |
| GRAFICARCLIENTEFINAL | PRIVADO | DOBLEENLAZADA | STRING | Devuelve el contenido del archivo con extensión dot de los clientes ya atendidos. |
| CREARDOT | PRIVADO | STRING, STRING | VOID | Crea el archivo con la extensión dot para ser procesado por medio de Graphviz. |
| GRAPHIZ | PRIVADO | STRING, STRING | VOID | Crea la imagen por medio de Graphviz y el archivo dot. |
| CARGARESTRUCTURA | PRIVADO | INT | VOID | Guarda el estado de las estructuras en cada paso que se ejecute la simulación |
| ORDENARCLIENTEFINAL | PRIVADO | INT | SIMPLE | Devuelve la lista final de cliente ya ordenada según el estado que se requiera. |
| GRAFICARBUSQUEDA | PRIVADO | SIMPLE | STRING | Devuelve el contenido del archivo con extensión dot de las imágenes ingresadas en la búsqueda de datos. |

# VISTAS

## PRINCIPAL

Menú principal de la simulación



1. PARAMETROS INICIALES

Permite cargar todos los datos que requiera el sistema.

1. EJECUTAR PASOS

Inicia la simulación con los datos previamente cargados.

1. ESTADO DEN MEMORIA DE LAS ESTRUCTURAS

Crea reportes de las estructuras creadas por medio de los pasos que se hicieron en la simulación.

1. REPORTES

Crea reportes con los datos generados en la simulación.

1. ACERCA DE …..

Muestra los datos del estudiante que realizó el programa.

1. BORRAR DATOS

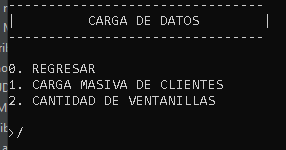
Reinicia todas las estructuras con los datos ingresados.

1. SALIR

Termina la simulación del sistema.

## PARAMETROS INICIALES

Menú de la carga masiva de datos de todos los clientes y las ventanillas ingresadas.



1. REGRESAR

Regresa al menú principal del sistema.

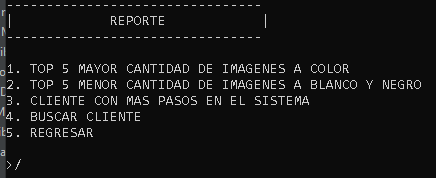
1. CARGA MASIVA DE CLIENTES

Carga todos los clientes por medio de un archivo JSON.

1. CANTIDAD DE VENTANILLAS

Carga todas las ventanillas por medio de un numero de ventanillas máximas que se generan.

## REPORTES



Crea un reporte gráfico de los datos ingresados y solicitados.

# LIBRERIAS SECUNDARIAS

## GRAPHVIZ

G raphviz (Graph Visualization) es un conjunto de herramientas de software para el diseño de diagramas definido en el lenguaje descriptivo DOT. Graphviz consiste en un lenguaje de descripción de gráficos llamado DOT, un conjunto de herramientas y librerías que pueden generar o procesar archivos DOT:

1. dot

Una herramienta en línea de comandos para producir imágenes en capas de grafo dirigido en una variedad de formatos de salida (PostScript, pdf, svg, png, etc.).

1. neato

Para graficar usando el "modelo de primavera".

1. sfdp

Un motor de diseño para grafos muy grandes.

1. fdp

Otro motor de diseño para grafos.

1. twopi

Para esquemas de grafos radiales.

1. circo

Para esquemas de gráficos circulares.

1. dotty

Una interfaz gráfica de usuario para visualizar y editar gráficos.

1. lefty

Un widget programable que muestra gráficos DOT y permite al usuario realizar acciones sobre ellos con el ratón.

## JSON-SIMPLE

Existen múltiples librerías para poder manipular contenido en JSON con Java y ya hemos visto cómo podemos crear objeto JSON con GSON. En este caso vamos vamos a crear objeto JSON con JSON-Simple mediante código Java. Así podremos comparar ambas librerías y ver cuál se adapta más a nuestras necesidades.

JSON-Simple es una librería ligera que nos permite serializar y deserializar contenido en JSON. A día de hoy es mantenida por los Clifton Labs.

Lo primero de todo será importar la librería JSON-Simple. Para ello lo que hemos hecho es añadir una dependencia Maven en nuestro pom.xml.

Contenido

[OBJETIVOS 2](#_Toc96193381)

[MANUAL 2](#_Toc96193382)

[APLICACIÓN 2](#_Toc96193383)

[REQUERIMIENTOS MÍNIMOS DEL ENTORNO DE DESARROLLO 2](#_Toc96193384)

[MÓDULOS 3](#_Toc96193385)

[OBJETOS 3](#_Toc96193386)

[RESUMEN DE CLASES 3](#_Toc96193387)

[FUNCIONALIDAD (CLASES) 4](#_Toc96193388)

[ESTRUCTURAS 4](#_Toc96193389)

[RESUMEN DE CLASES 4](#_Toc96193390)

[FUNCIONALIDAD 5](#_Toc96193391)

[PRINCIPAL 6](#_Toc96193392)

[CONTROLADOR 6](#_Toc96193393)

[RESUMEN DE MÉTODOS 7](#_Toc96193394)

[VISTAS 9](#_Toc96193395)

[PRINCIPAL 9](#_Toc96193396)

[PARAMETROS INICIALES 9](#_Toc96193397)

[REPORTES 10](#_Toc96193398)

[LIBRERIAS SECUNDARIAS 10](#_Toc96193399)

[GRAPHVIZ 10](#_Toc96193400)

[JSON-SIMPLE 11](#_Toc96193401)