

**IC4700**

*Lenguajes de Programación*

**II Proyecto**

*(Gestión de Inventarios)*

**Integrantes:**

Mynell Jemuel Myers Hall

Andrew Denilson López Herrera

Ingeniería en Computación

Prof. Allan Rodríguez Dávila

Semestre 1, 2024

24/04/2024

# Manual de Usuario y Pruebas de funcionalidad

Este proyecto ha sido construido en Windows usando Cabal como herramienta (la debe tener descargada), por lo que el incumplimiento de este entorno de ejecución puede causar errores en el programa.

1. Compilación

Ante todo, descargaremos el proyecto desde el repositorio de GitHub (Puede encontrar el link del repositorio en la sección de final).

Luego de descargar el proyecto nos posicionaremos en la carpeta programa y abriremos la terminal en la posición de la carpeta.

En la terminal escribiremos el comando de ejecución de Cabal **‘cabal run’**. Con esto el programa se ejecutará para su uso, es probable que dure algunos minutos la interpretación del proyecto antes de ejecutarse.



1. Ejecución

Gracias al método de ejecución de cabal el programa va a iniciar después de su compilación, y mostrara inmediatamente el menú del programa.

1. Uso del programa

Al inicio, el programa le pedirá los datos de la empresa. Se pide el nombre de la empresa, el sitio web de la empresa y el contacto de la empresa. Se rellena cada espacio con la información solicitada.

Texto

Descripción generada automáticamente

Luego se pide la información de las bodegas. Por cada bodega se solicita la cantidad de espacio en metros cúbicos de la bodega y la ubicación de la bodega. La cantidad de espacio de la bodega solo pueden ser números, estos pueden contener decimales.

Después de haberse agregado cinco bodegas, el sistema pregunta si desea agregar una bodega más. Para responde SI se escribe una ‘S’, esta debe ser mayúscula, por el contrario, si responde NO se escribe cualquier otra letra.

Texto

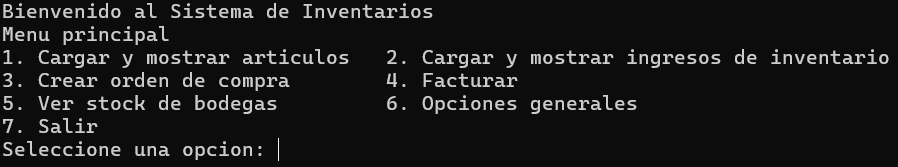
Descripción generada automáticamente

Luego, se deben ingresar los usuarios de la aplicación. Para los usuarios se pide la cédula (debe ser número), el nombre (debe ser texto), el primer apellido (debe ser texto) y el puesto (debe ser texto). No se permiten cédulas repetidas. Después de haber agregado cinco usuarios, el sistema le preguntará si desea agregar otro usuario, si su respuesta es un SI debe escribir ‘S’ en mayúscula solamente, si su respuesta es un NO debe escribir cualquier otra letra.

Interfaz de usuario gráfica, Aplicación

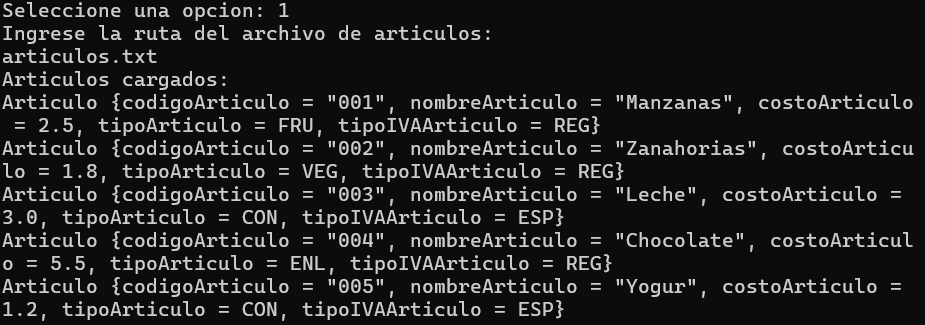
Descripción generada automáticamente

Menú Principal: este menú muestras las opciones operativas con la opción de poder ir a las opciones generales, en total son 7 opciones únicas.



El usuario deberá ingresar cualquiera de los números del menú para ingresar a las opciones. Cualquier otro número será rechazado por el sistema.

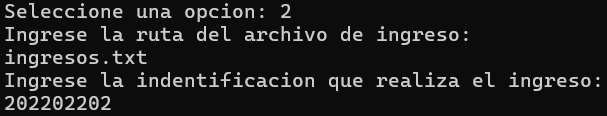
Cargar y mostrar artículos: con este espacio se puede leer un archivo csv lleno de artículos, si se cumple el formato de los artículos se ingresan y se muestran cuáles fueron los artículos logrados.



Registro de artículos en archivo JSON



Cargar y mostrar ingresos: con este espacio se puede leer un archivo csv lleno de ingresos, si se cumple el formato de los ingresos se ingresan y se muestran cuáles fueron los ingresos logrados.

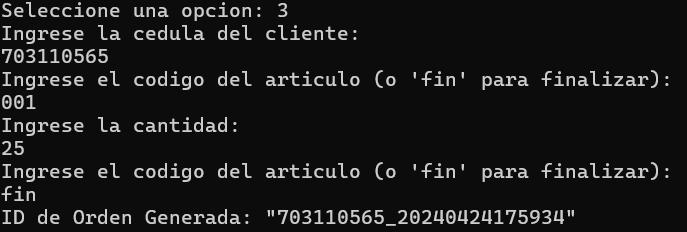


Registros de ingresos en archivo JSON



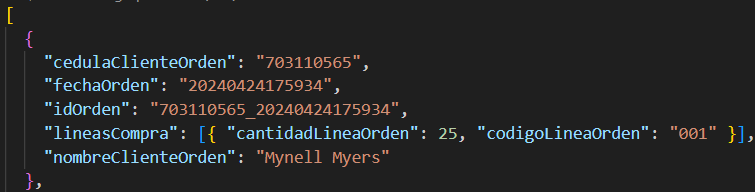
Si no existen alguno de los dos datos, entonces el programa finalizara.

Crear orden de compra: este apartado crea una orden de compra, con sus respectivos datos para luego ser facturado y comprobados los datos de esta orden antes de retirar los datos.

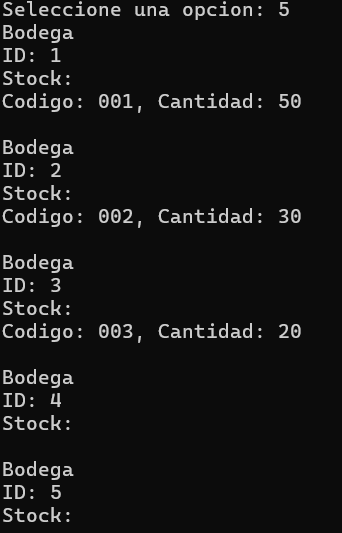


Para finalizar el ingreso de artículos, debe de ingresar ‘fin’.

El siguiente es el archivo de ordenes registradas.



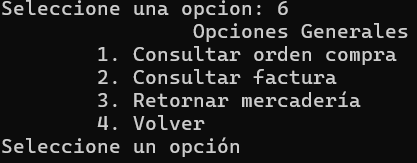
Ver Stock de bodegas: Sirve para ver los datos de las bodegas, incluidos el stock ingresado a ellos.



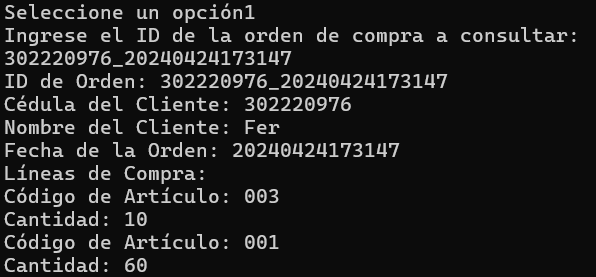
Archivo del registro de bodegas.



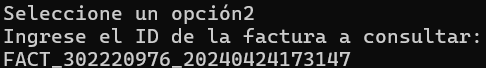
Opciones Generales: Es para ingresar a las demás opciones, que son de consulta y demás.



Consultar orden compra: opción general, puede consultar cualquier orden realizada y guardada en el archivo mediante el id Generado.



Consultar factura: opción general, puede consultar cualquier factura realizada y guardada en el archivo mediante el id Generado.



# Pruebas de funcionalidad

El registro de la empresa funciona con los siguientes valores:

Texto

Descripción generada automáticamente  


El registro de bodegas funciona con los siguientes valores:

Texto

Descripción generada automáticamente

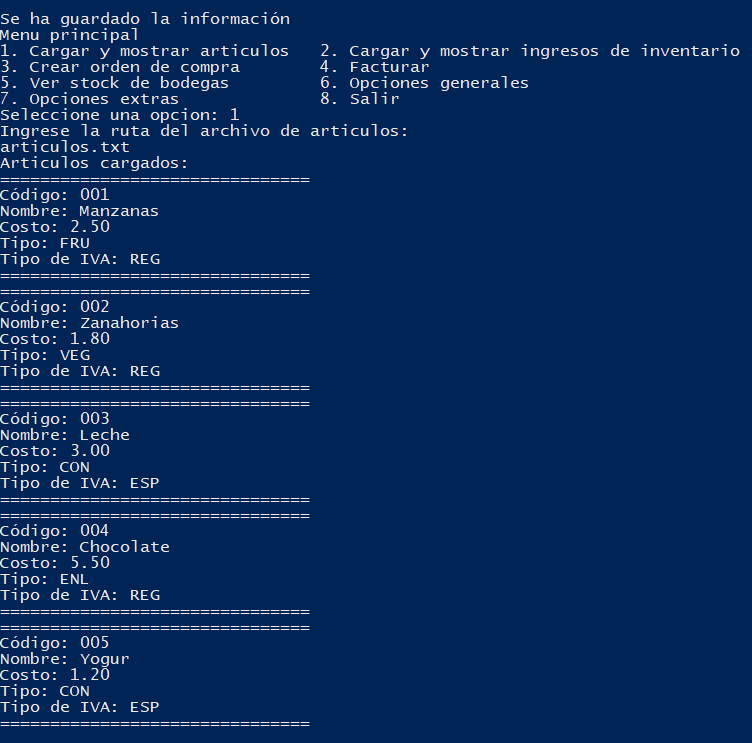
El registro de usuarios funciona con los siguientes datos:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente



La carga y visualización de artículos nuevos funciona con los siguientes datos:





La carga y visualización de ingresos funciona con los siguientes datos:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente



La creación de ordenes de compra funciona con los siguientes datos:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente



Ver stock de bodega funciona:

Texto

Descripción generada automáticamente

Las opciones generales funcionan:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

# Descripción del problema

Un sistema de gestión de inventarios es una herramienta que permite a las empresas llevar un registro preciso y actualizado de sus existencias de productos. Este tipo de sistema proporciona una plataforma para administrar y controlar el flujo de productos desde su recepción hasta su almacenamiento, distribución y venta.

Dentro de las principales funcionalidades se encuentra el seguimiento de inventario, la recepción de mercancías, gestión de órdenes de compra, gestión de ventas, control de stock, organización de almacenes, informes y análisis.

El proyecto busca someter al estudiante a entornos y realidades lo más cercano a la realidad, con el objetivo de generar un aprendizaje efectivo.

El trabajo consiste en implementar una aplicación de escritorio para administrar la gestión de inventarios utilizando el lenguaje Haskell y la Programación Funcional.

El programa deberá desplegar un menú al usuario con dos submenús o grupo de funciones (el menú debe mostrarse recurrentemente después de cada opción realizada -a excepción del salir- y tener la opción de subir al menú principal desde los menús secundarios) y salir.

# Diseño del programa

Persistencia de datos: Para la persistencia de datos se decidió utilizar los archivos JSON, ya que estos son muy flexibles al momento de guardar y extraer datos.

Se decidió utilizar el administrador de proyectos cabal, ya que este permitía la importación de otras librerías útiles como aeson, cabal hace mucho más fácil organizar el proyecto.

Las estructuras utilizadas para hacer el manejo de la información es la Data que ofrece Haskell.

Cargar empresa: Se busca en el archivo JSON si la empresa existe, si el archivo está vacío se procede a crear una empresa nueva y guardar la empresa creada en el archivo JSON, si el archivo no está vacío se extrae la información del archivo JSON.

Cargar bodegas: Se cargar el arreglo JSON que guarda las bodegas. Si este arreglo está vacío, se procede a crear cinco bodegas o más, esto se hace de manera recursiva. El ID de cada bodega se va a asignando de manera automática durante la creación. Si el arreglo de bodegas no está vacío, se cargan las bodegas que existen en el archivo.

Cargar usuarios: Se carga el arreglo JSON que guarda los usuarios. Si el arreglo está vacío, se decide crear los usuarios, cinco o más, luego se guardan en el archivo JSON. Si el arreglo no está vacío, se decide cargar los usuarios que se encuentran en el arreglo.

Maybe: Otro pilar importante en nuestro proyecto fue el Maybe, que sirvió para esas búsquedas de datos donde puede no haber resultados o si, esta implementación agilizo el programa y nos dio puerta a otras funciones extras de distintas librerías.

Otro punto importante es que se decidió en conjunto resguardar los tipos de datos personalizados en una carpeta especifica par ser consultada fácilmente.

Y por ultimo se empleo una interfaz de consola agradable para el usuario, buscando la claridad de las instrucciones y empeñando en ser lo más intuitiva posible.

# Librerías usadas

Todas las librerías usadas fueron importadas con el administrador de proyecto cabal, este es el que permite una importación más fácil sin la necesidad de descargar archivos. Las librerías utilizadas son:

1. aeson
2. text
3. bytestring
4. filepath
5. directory
6. time
7. filepath
8. directory
9. IO
10. Generics
11. data

# Análisis de resultados

|  |  |  |  |
| --- | --- | --- | --- |
| Objetivos | Sí | No | Observaciones |
| Documentación interna | X |  |  |
| Documentación externa | X |  |  |
| Información comercial | X |  |  |
| Información de bodegas | X |  |  |
| Información de usuarios | X |  |  |
| Cargar y Mostrar artículos | X |  |  |
| Cargar y Mostrar ingresos de inventario | X |  |  |
| Crear orden de compra | X |  |  |
| Facturar | X |  | La facturación se logró, pero presenta conflictos al verificar los elementos de una bodega. |
| Ver stock de bodegas | X |  |  |
| Consultar orden de compra | X |  |  |
| Consultar factura | X |  |  |
| Retornar mercadería |  | X | No se logró regresar los artículos por problemas encontrar la bodega ideal |
| Persistencia de datos | X |  |  |

# Repositorio:

<https://github.com/AndrewLopezHerrera/Proyecto2LenguajesDeProgramacion.git>