Luis Rodrigo Morales Florián

202208521

LABORATORIO ESTRUCTURAS DE DATOS Sección B

Fase 1

Manual Técnico

# Descripción del proyecto

La empresa "Pixel Print Studio" se dedica a la impresión de figuras de pixel art en distintos tamaños y tipos de papel. Con el crecimiento constante de clientes, ha surgido la necesidad de optimizar los procesos de recepción y producción. En este sentido, se solicita a un estudiante de ingeniería en sistemas aplicar conocimientos en estructuras de datos para mejorar la eficiencia operativa.

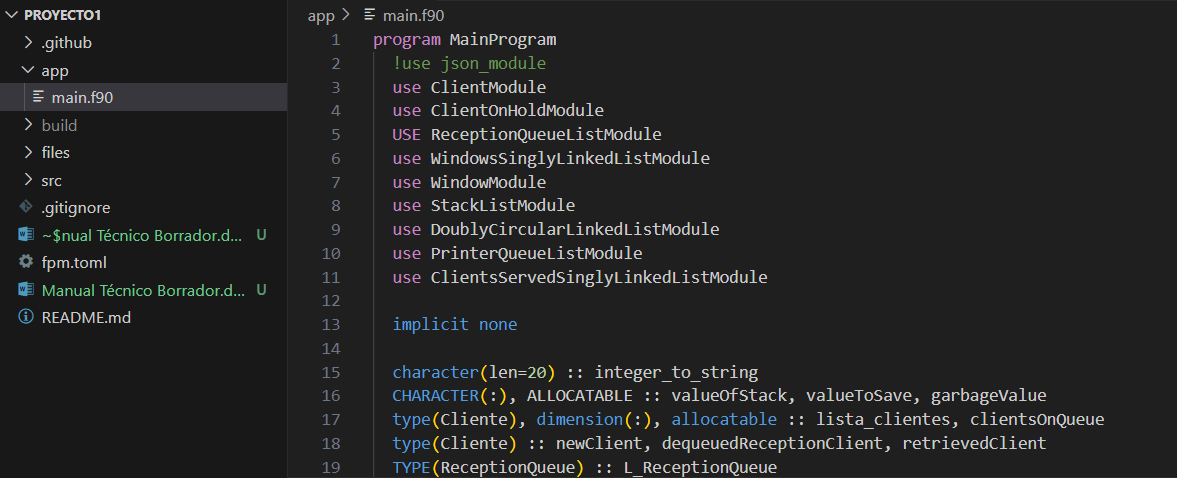
Deberás desarrollar una aplicación que utilice estructuras de datos y algoritmos para simular los diversos procesos en la empresa "Pixel Print Studio". La aplicación deberá representar visualmente las estructuras mediante bibliotecas compatibles, como Graphviz.

# Generalidades

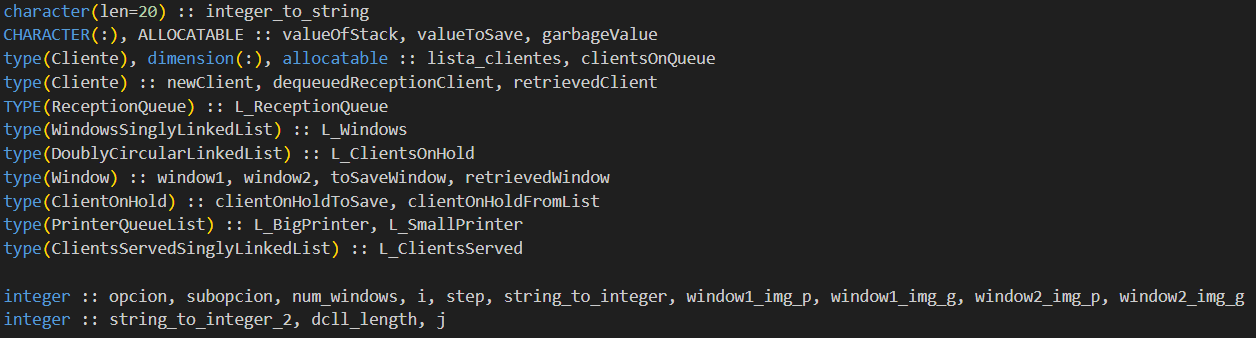
* El manejador de proyectos y paquetes es FPM (Fortran Package Manager)
* Para la graficación cuenta con la generación de un archivo de texto, el cual genera un lenguaje entendible para Graphviz, de modo que luego se tendrá que copiar ese código y ponerlo en un intérprete para poder observar el resultado
* Se realiza carga masiva de clientes a través de un json, y la librería que procesa los JSON se llama json\_module
* Se utiliza un sistema de:
  + Main File: El archivo principal donde arranca la aplicación, aquí mismo se programan y ejecutan los procesos de cada módulo de la tienda.
  + Models: Son modelos de datos, los cuales se requieren para una mejor estructura. Son modules de fortran, que gracias a sus atributos, éstos modelos pueden ser implementados en otros lados y hacer uso de sus atributos e incluso hasta métodos
  + Lists: Son todas las listas TDA que utilizamos en el proyecto. En mi caso, opté por crear algunas listas “personalizadas”, de modo que el nodo de cada lista aceptara diferentes tipos de datos, ya sean nativos o de los Models.

Se procede entonces con la descripción de cada parte del sistema:

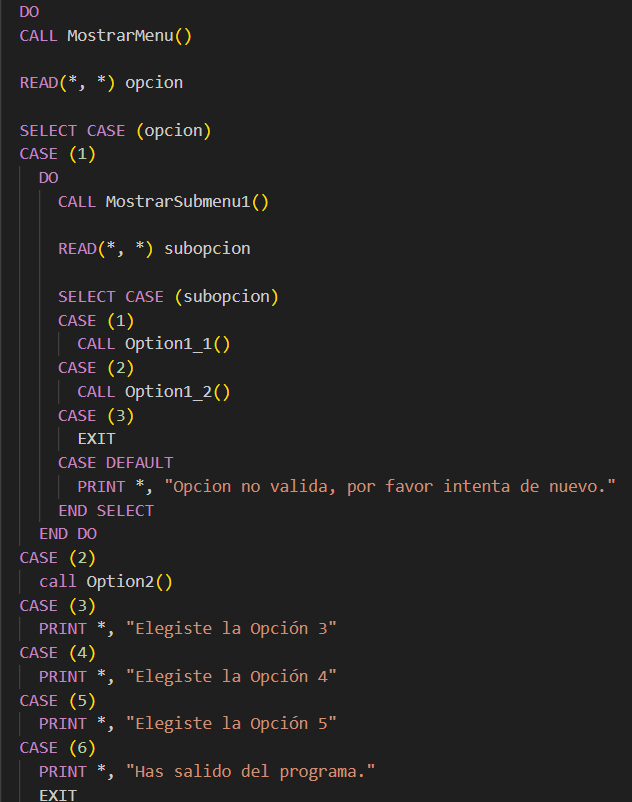
## Main File



Básicamente se inicia con la declaración de variables con cada uno de sus respectivos tipos. Todas las variables en pantalla:

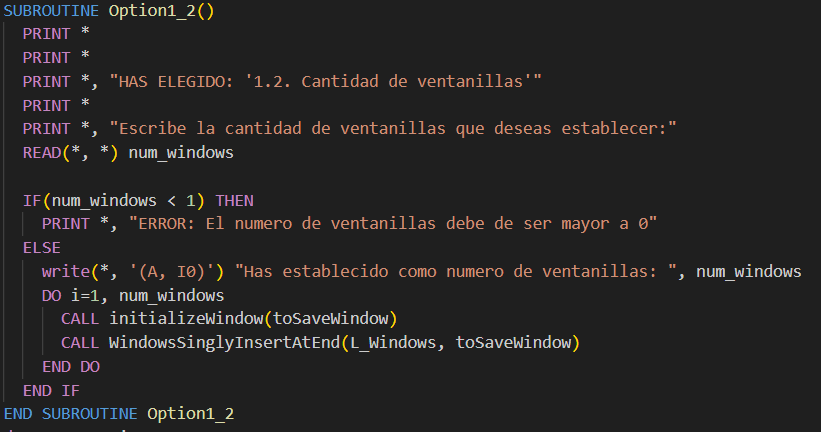


Luego se procede con la ejecución del menú, donde cada opción está separada por su propia subrutina:



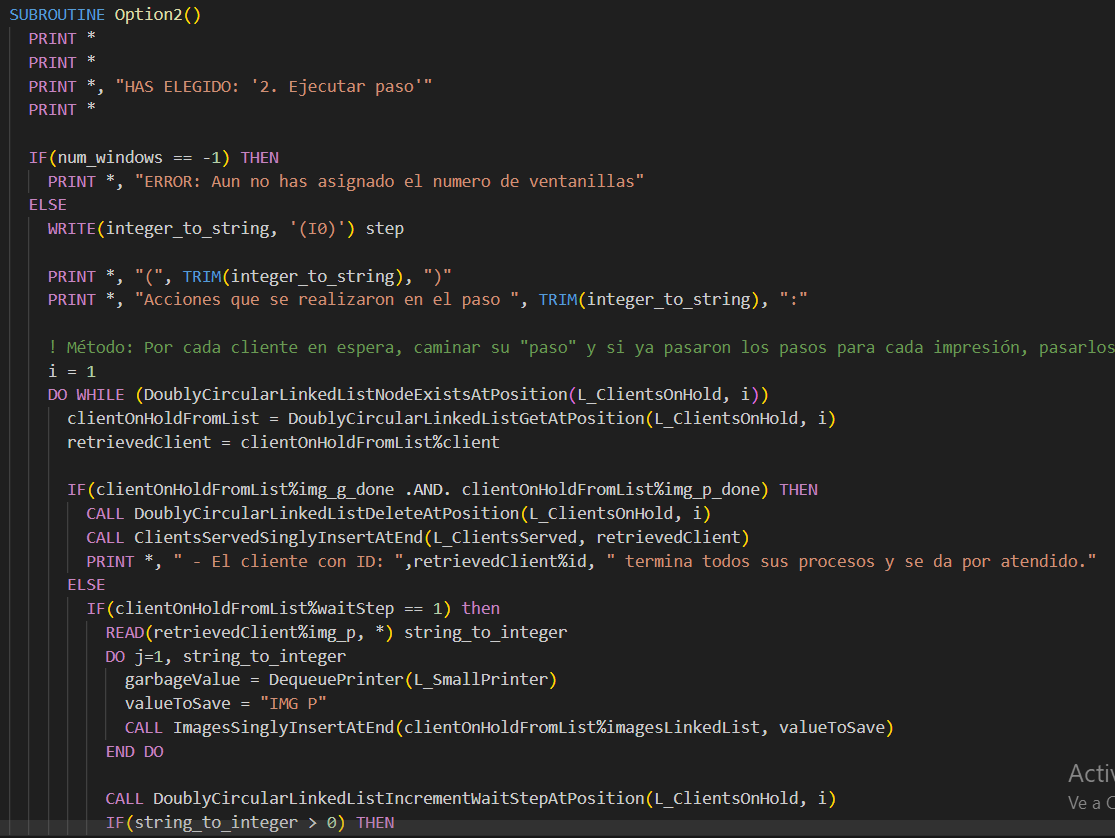
Hablaremos de dos destacables con el objetivo de ejemplificar un poco, las demás opciones usted tome en cuenta que son similares, así que estos ejemplos deberían de ser suficientes.

### 1.2. Cantidad de ventanillas



Aquí se asignan las ventanillas que el cliente requiera, luego se crea una lista simplemente enlazada con esa cantidad de ventanillas. El número se queda guardado en el sistema y cada vez que se quiera tratar con las ventanillas, se procede con una iteración DO de i=1 hasta num\_windows, que es la cantidad establecida por el usuario.

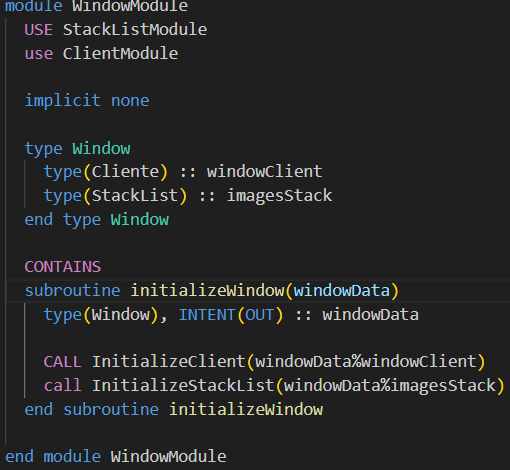
### 2. Ejecutar paso



Aquí podemos observar que, por cada módulo del programa, se tiene un bloque donde se hacen los chequeos respectivos y finalmente, se procede con los pasos que se requieran.

Básicamente es así como está distribuido el main file.

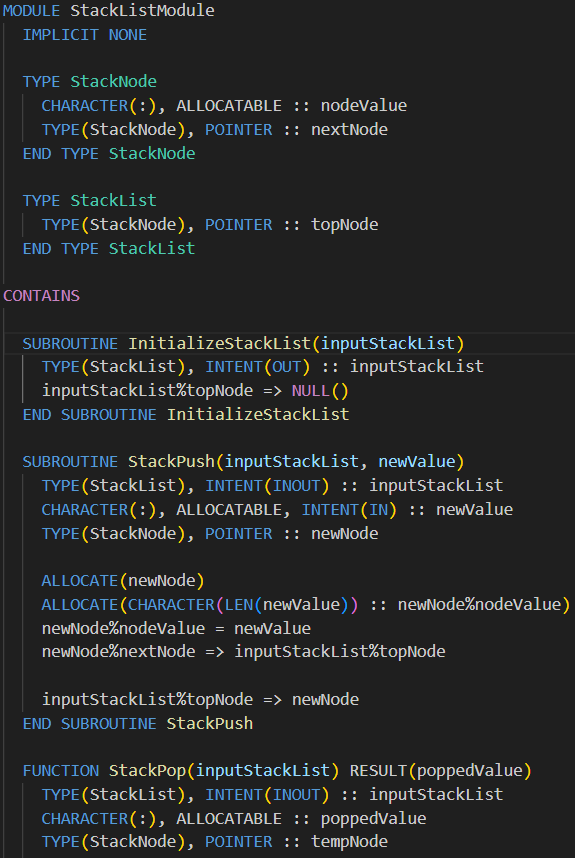
## Models



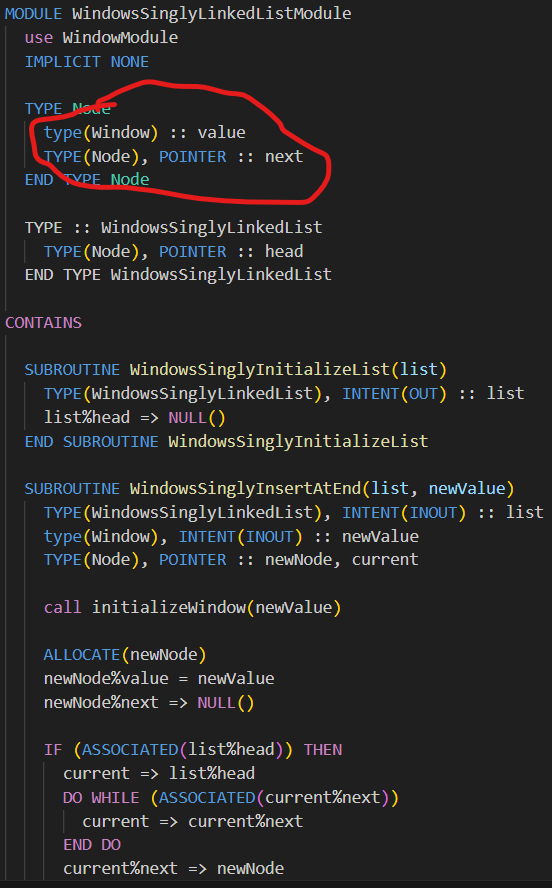
Como se había dicho, cuando se necesita generar una estructura específica en el programa, se recurre a módulos, y se aprovecha sus capacidades de generar atributos. De esta manera, se puede generar tipos de datos. Recordamos que para acceder a un atributo de una variable del tipo Modelo es con modelo%atributo, lo cual hace más eficiente el manejo de datos en fortran.

Al no existir algo como constructores, ya que fortran no es un lenguaje que tenga muy “pulida” la programación orientada a objetos, se recurre a una subrutina, donde se envía una variable del tipo del Modelo y se le agrega lo que se requiera.

## Lists



Para las listas TDA, se requiere también del uso de módulos. Aquí, gracias a la capacidad de tener atributos-métodos, y el manejo de apuntadores, es posible lograr este tipo de listas. Como decía, recurrí a crear listas personalizadas para cada lista requerida en el proyecto, por tanto, el nodo de la lista cambia a, por ejemplo, a tener uno o varios atributos de datos nativos o Modelos. Así mismo, se crean métodos personalizados, donde también se puede acceder a una lista enviada bajo tal lista y su tipo y se puede gestionar desde ahí.



Aquí se puede una lista simplemente enlazada modificada para manejar todas las ventanillas. Nótese como cambia el nodo de la lista, usando tipos de Modelos fabricados por el desarrollador y también los atributos propios de la lista.