Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 2do. Semestre 2024

PROYECTO 2 IPC2

202307534 – Daniel Alejandro Portillo Garcia

Resumen

El manejo de datos por medio de estructuras de TDA demuestra ser la forma mas eficiente de aprender la lógica de interpretación de datos dentro del mundo de la programación, usando estructuras de diseño web BackEnd y FrontEnd, formado estructuras dinámicas para que una persona pueda familiarizarse con todos estos procesos. Ma allá de solo el manejo de XML, se observa que, En lenguajes de programación, pueden usarse varios tipos al mismo tiempo, con distintas funciones, entre estas existe HTML y Python, que pueden trabajar de manera simultanea para mostrar y procesar datos dependiendo de instrucciones que se nos solicite manejar a nosotros como programadores. El manejo de estructura de listas nos permite el ordenar e insertar datos de manera más compleja, pero de una forma de interpretación mas eficiente, puesto que estas estructuras se usan para que las futuras generaciones de programadores comprendan mejor de las estructura de manejo de datos.

Palabras clave
Listas, datos, TDAs, Dinámico.

Abstract

Data management through TDA structures proves to be the most efficient way to learn the logic of data interpretation within the programming world, using BackEnd and FrontEnd web design structures, forming dynamic structures so that a person can become familiar with all these processes. Beyond just the handling of XML, it is observed that, in programming languages, several types can be used at the same time, with different functions, among these there is HTML and Python, which can work simultaneously to display and process data depending on instructions that we are asked to handle as programmers. The management of list structures allows us to order and insert data in a more complex way, but in a more efficient way of interpretation, since these structures are used so that future generations of programmers better understand data management structures.

Keywords
Lists, data, TDAs, Dynamic.

Introducción

El manejo de eventos simultáneos es una forma eficiente para el manejo, uso e interpretación de datos, por medio de esto podemos leer e interpretar archivos y darles ciertas instrucciones en base a ciertos parámetros que nosotros especifiquemos a ciertos programas. Bien conocemos que el manejo ordenado de datos es una base de suma importancia para el desarrollo de un programador, por ello se debe de saber la forma de como enviar y manejar datos por medio de estructuras simples como listas, para mejorar la comprensión lógica de la dinámica de manejo de datos. Por ello en este ensayo se muestra el proceso de aplicación de un proyecto del curso de Introducción a la programación y computación en el ámbito de listas enlazadas y manejo de datos, para que los estudiantes de ingeniería aprendan a como es que funcionamiento eficiente de las estructuras de TDAs y manejo de datos dinámicos.

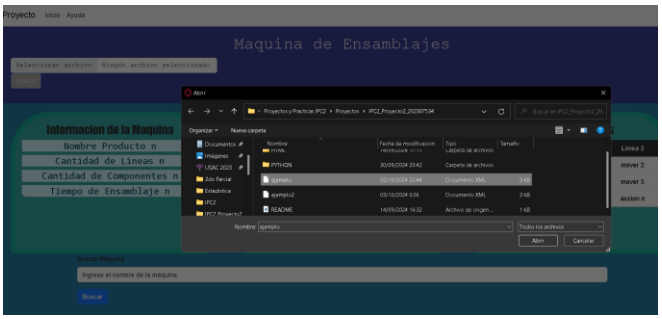
Desarrollo del tema

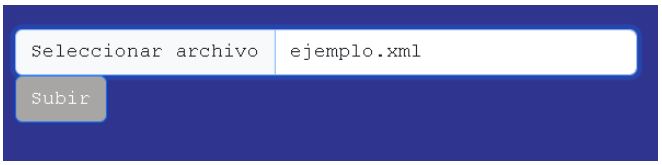
Para demostrar el manejo de las diversas estructuras de datos por medio de listas enlazadas y TDAs, y la lectura de datos por medio de XML para generar gráficos por medio del lenguaje de programación de Python se necesita conocer el problema que se nos solicita resolver. Utilizaremos herramientas externas para la lectura y creación gráfica, xml.dom.minidom como herramienta de lectura y creación y lenguaje dot para interpretar Grafiz.

El problema principal solicita en la lectura de datos para simular una maquina virtual, la cuel tendrá varios atributos, como un nombre, lines de ensamblaje, componentes a ensamblar y un tiempo definido de ensamblaje, el brazo puede moverse de línea en line hasta llegar al componente de ensamblaje, tenen que ensamblarse de manera simultanea y obtener un tiempo optimo de producción, para después hacer reportes del total de maquinas generadas y generar un reporte de los movimientos de líneas de producción, para ello se opto por un diseño de html basado en back end y frontend para manejar los datos de forma eficaz.

Flujo del programa:

A continuación, se ejemplificará el proceso del programa desarrollado, explicando desde la entrada de datos hasta su salida. L interfaz inicial le indicara al usuario el manejo de datos. En donde encontrara la opción para subir archivos de forma acumulativa y procesarlos, es importante destacar que solo se admitirán archivos .XML, el usuario podrá visualizar el tipo de archivo antes de subirlo para después almacenarlo en el sistema.

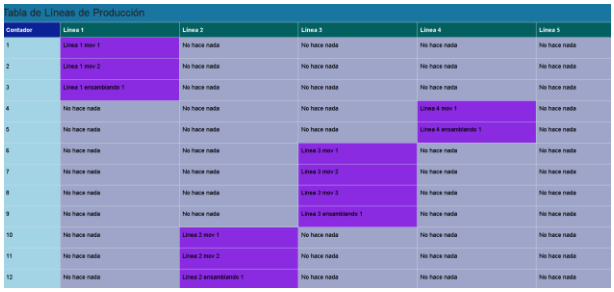




Al momento de almacenar los datos del xml se precederá a la opción de búsqueda, que permitirá

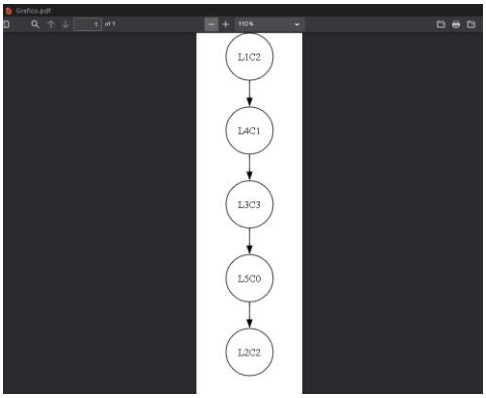
buscar una maquina en concreto y mostrar sus datos, líneas de producción y componentes. Al ser encontrada la maquina redireccionara a la pagina con los datos y mostrara una opción para poder buscar entre varios productos dentro de la máquina.

Al mostrar los datos del producto y maquina seleccionada se precederá hacer un reporte de una tabla con los movimientos solicitados del brazo virtual, colocando cada movimiento y componente en su línea de producción correspondiente, para llevar un control del tiempo del proceso.



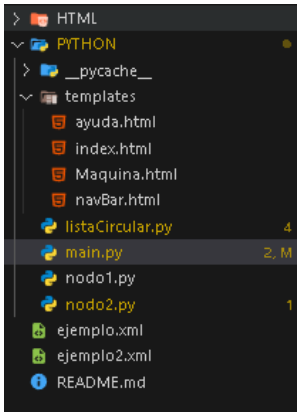
Seguido se mostrara el total del tiempo realizada y opciones para generar el archivo con todas las maquina cargadas y un reporte grafico del producto y las líneas y componentes de movimientos utilizados


```
<?xml version="1.0" ?>
<SalidaSimulacion>
  <Maquina>
    <NombreMaquina>TelevisoresPro</NombreMaquina>
    <CantidadLineasProduccion>3</CantidadLineasProduccion>
    <CantidadComponentes>13</CantidadComponentes>
    <TiempoEnsamblaje>2</TiempoEnsamblaje>
    <ListadoProductos>
      <Producto>
        <NombreProducto>tv samsung</NombreProducto>
        <Elaboracion>
          L2C5
          L1C6
          L2C12
          L3C4
          L3C10
        </Elaboracion>
      </Producto>
      <Producto>
        <NombreProducto>tv hisense</NombreProducto>
        <Elaboracion>
          L1C4
          L3C8
          L2C11
          L1C2
          L3C0
        </Elaboracion>
      </Producto>
      <Producto>
        <NombreProducto>televisión LG</NombreProducto>
        <Elaboracion>
          L2C5
          L1C7
          L3C1
          L2C9
          L3C12
        </Elaboracion>
      </Producto>
    </ListadoProductos>
  </Maquina>
```



Con esto el uso de manejo de datos se demuestra de forma eficiente y cumpliendo con la simulación de una máquina de producción de forma virtual.

Archivos de clases utilizadas dentro del programa



Clases y funciones importantes de cada archivo

```
from nodo1 import nodo_1
from nodo2 import nodo_2, nodo_3, nodo_4
from os import startfile, system
import os

class ListaMaquinas:
    def __init__(self):
        self.cabeza = None

    def insertarMaquina(self, nombre, ctdLinea, ctdComponente, tiempo):...

    def buscarMaquina(self, nombre):...

    def imprimir(self):...

    '''def agregarElaboracionProducto(self, nombre_maquina, producto_nombre

class ListaProductos:
    def __init__(self):
        self.cabeza = None

    def insertarProductos(self, nombre, elaboracion):...

    def buscarProducto(self, nombre):...

    def imprimir(self):...

    def crearGraphviz(self, producto):...

class ListaElaboracion:
    def __init__(self):...

    def insertarElaboracion(self, linea, componente, tiempo):...


    def movs(self):...

    def imprimir(self):...

    def iterar(self):...

    def buscarElaboracion(self, lineaN, componenteN):...

class ListaMovimientos:
    def __init__(self):...

    def insertarMovimiento(self, movimiento):...

    def imprimir(self):...

    def iterar(self):...
```

Conclusiones

- El uso de manejo de datos por medio de listas y TDAs permite una mejor comprensión del proceso de interpretación de datos dentro de estructuras XML basada en programación POO.
- El manejo de datos Backend y Frontend demuestra ser una forma eficiente de manejo de datos amigable para los usuarios, evidenciando el proceso de tras de nuestros programas, mostrando como es la lógica de interpretación por varias capas de manejo de datos.
- Los procesos de inserción de datos mostraron ser mas complejos, puesto que inserción de listas dentro de otras listas permiten encapsular datos para manejos mas rápidos, pero es una forma mas compleja de realizarlos.

Referencias bibliográficas

Graphviz. (s.f.). *Graphviz*. PyPI. Recuperado de https://pypi.org/project/graphviz/#files

Anexos

Diagrama de Clases

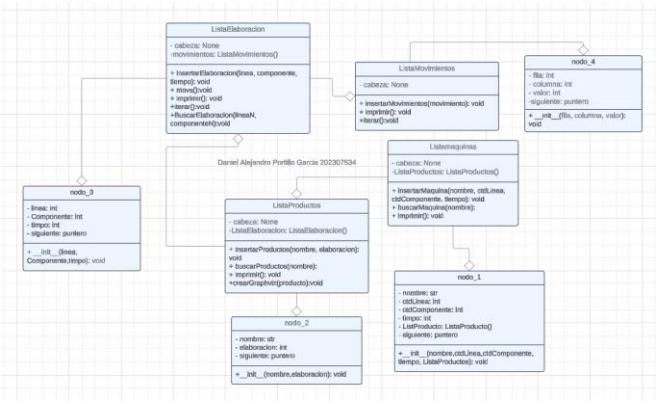


Diagrama de Actividades