Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Introducción a la Programación y Computación 2

Inga. Claudia Liceth Rojas Morales
Ing. Marlon Antonio Pérez Türk
Ing. Byron Rodolfo Zepeta Arevalo
Ing. José Manuel Ruiz Juárez
Ing. Edwin Estuardo Zapeta Gómez

Tutores de curso:
Oscar René Jordán Orellana, Juan Pablo Osuna de León
Astrid Edith Hernández González, Henry Adolfo Gálvez
César Dionicio Sazo Mayen, Oscar Alejandro Rodríguez Calderón
Edwar Everaldo Zacarias, Javier Estuardo Lima Abrego
Mónica Raquel Calderón Muñoz, Marco Antonio López Grajeda

PROYECTO 2

OBJETIVO GENERAL

Se busca que el estudiante sea capaz de dar una solución al problema que se le plantea, mediante la lógica y conocimientos que se les ha impartido durante la clase y aplicarlos en el laboratorio.

OBJETIVOS ESPECÍFICOS

- Implementar una solución utilizando el lenguaje de programación Python.
- Utilizar estructuras de programación secuenciales, cíclicas y condicionales.
- Que el estudiante aprenda a generar reportes con la herramienta Graphviz.
- Que el estudiante sea capaz de manipular archivos XML.
- Que el estudiante utilice los conceptos de TDA y aplicarlos a memoria dinámica.
- Uso de programación orientada a objetos.

DESCRIPCIÓN DEL PROBLEMA

La empresa Digital Intelligence, S. A. ha desarrollado una máquina capaz de ensamblar las partes de cualquier producto.

La máquina creada por Digital Intelligence, S.A. puede construir cualquier producto ensamblando automáticamente los componentes (partes) que lo conforman. Para esto, la máquina desarrollada consta de **"n"** líneas de ensamblaje y un brazo robótico para cada una de éstas, además, cada línea de ensamblaje posee un mecanismo que le permite acceder a **"m"** componentes distintos. El brazo robótico demora 1 segundo en colocarse sobre el recipiente que contiene el 1er componente, 2 segundos para colocarse en el recipiente que contiene el 2do componente y así sucesivamente hasta requerir **"m"** segundos para colocarse en el **"m-ésimo"** componente. Adicionalmente, para ensamblar el componente en el producto que se construye, el brazo robótico utilizará **"x_m"** segundos para el **"m-ésimo"** componente. La figura 1 muestra una línea de ensamblaje.

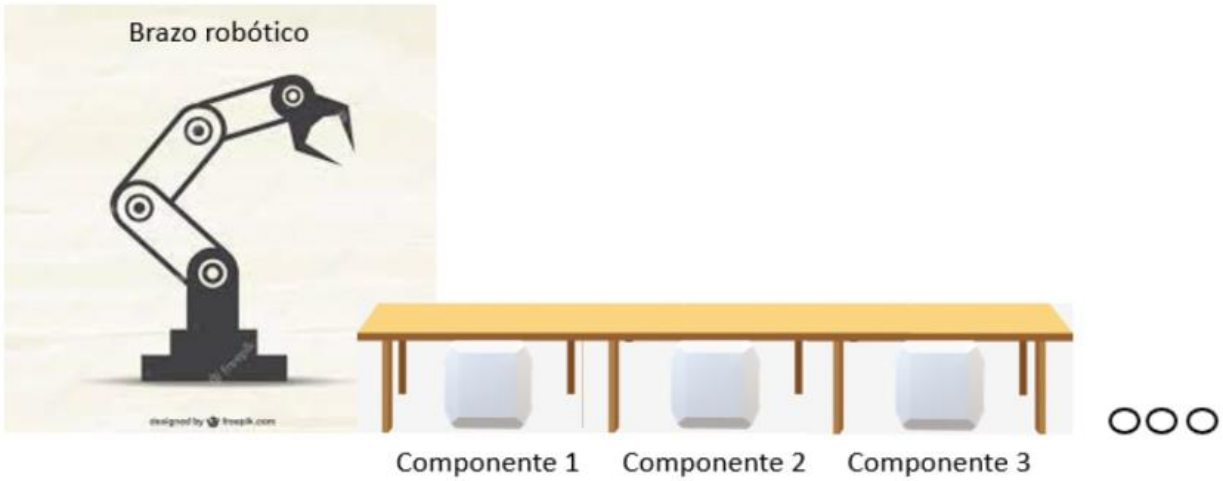


Figura No. 1 – Una línea de ensamblaje

La máquina funciona de la siguiente forma: Se define un producto a ensamblar y se le da un conjunto instrucciones indicando la línea de producción y el componente que deben ser ensamblados para construir dicho producto. En cada segundo, un brazo robótico solamente puede moverse hacia adelante, moverse hacia atrás, ensamblar, o no hacer nada. Los brazos robóticos pueden accionarse simultáneamente, es decir, pueden moverse varios brazos a la vez, sin embargo, el proceso de ensamblaje si debe realizarse uno a la vez, ya que la construcción del producto requiere que se ensamble en el orden correcto.

Por ejemplo, si se tiene una máquina con 2 líneas de producción, con 5 componentes cada línea y cada componente requiere 1 segundo para ser ensamblado (cualquier componente de cualquier línea), entonces, se puede definir el producto "SmartWatch" que requiere la siguiente secuencia de trabajo:

L1C2 L2C1 L2C2 L1C4

Donde: L1C2 corresponde a Línea de ensamblaje 1 Componente 2

L2C1 corresponde a Línea de ensamblaje 2 Componente 1

L2C2 corresponde a Línea de ensamblaje 2 Componente 2

L1C4 corresponde a Línea de ensamblaje 1 Componente 4

Usted ha sido contratado para desarrollar un software que simule el funcionamiento de esta máquina con “n” líneas de ensamblaje y cada línea de ensamblaje con “m” posibles componentes a seleccionar de forma que pueda predecir el tiempo “óptimo” para elaborar cualquier producto que pueda ser ensamblado en la máquina. Entonces, si se solicitara a su software la simulación de la construcción de un producto “SmartWatch”, su software debe dar la siguiente salida:

Tiempo	Línea de ensamblaje 1	Línea de ensamblaje 2
1er. Segundo	Mover brazo – componente 1	Mover brazo – Componente 1
2do. Segundo	Mover brazo – componente 2	No hacer nada
3er. Segundo	Ensamblar componente 2	No hacer nada
4to. Segundo	Mover brazo – Componente 3	Ensamblar – Componente 1
5to. Segundo	Mover brazo – Componente 4	Mover brazo – Componente 2
6to. Segundo	No hacer nada	Ensamblar – Componente 2
7mo. Segundo	Ensamblar componente 4	No hacer nada
El producto SmartWatch se puede elaborar óptimamente en 7 segundos.		

Figura No. 2 - Tabla de Resultados

ENTRADA

El programa podrá recibir 2 tipos de archivos XML, el primero, para configurar la máquina y el segundo que contendrá los productos que deben ser simulados.

El archivo XML para configurar la máquina tendrá la siguiente estructura:

maquina.xml

```
<Maquina>
    <CantidadLineasProduccion>
        <!--[0<NumeroEntero<1000]-->
    </CantidadLineasProduccion>
    <ListadoLineasProduccion>
        <LineaProduccion>
            <Numero>
                <!--[NumeroEntero>0]-->
            </Numero>
            <CantidadComponentes>
                <!--[0<NumeroEntero<1000]--> *Componentes de la línea
            </CantidadComponentes>
            <TiempoEnsamblaje>
                <!--[NumeroEntero>0]--> *Representa el tiempo en segundos
                                         que toma ensamblar en la línea
            </TiempoEnsamblaje>
        </LineaProduccion>
        …
    </ListadoLineasProduccion>
    <ListadoProductos>
        <Producto>
            <nombre>
                <!--[Texto]--> *Nombre del Producto
            </nombre>
            <elaboracion> *Corresponde a instrucciones para ensamblar el producto
                <!--[Texto con formato L1pC1p L2pC2p L3pC3p … LnpCnp]-->
            </elaboracion>
        </Producto>
        …
    </ListadoProductos>
</Maquina>
```

El archivo XML que contiene el listado para simular la elaboración de productos tendrá la siguiente estructura:

simulación_N.xml

```
<Simulacion>
    <Nombre>
        <!--[Texto]-->
    </Nombre>
    <ListadoProductos>
        <Producto>
            <!--[Texto]--> *Nombre de producto configurado en la máquina
        </Producto>
        <Producto>
            <!--[Texto]2-->
        </Producto>
        …
    </ListadoProductos>
</Simulacion>
```

Tomar en cuenta que, debido a que cada archivo de simulación permite simular 1 o muchos productos, esto significa que pueden cargarse 1 o más archivos de simulación diferentes con diferentes productos. Además, que debe ser posible simular un producto en específico a través de la interfaz de usuario (esto se detalla en la sección de Interfaz de Usuario).

SALIDA

Para cada archivo de simulación o simulación individual de elaboración de productos se debe generar una salida como la siguiente:

```
<SalidaSimulacion>
    <Nombre>
        <!--[Texto]--> *Nombre de la simulación que generó esta salida.
    </Nombre>
    <ListadoProductos>
        <Producto>
            <Nombre>
                <!--[Texto]-->
            </Nombre>
            <TiempoTotal><!--[NumeroEntero>0]--></TiempoTotal>
            <ElaboracionOptima>
                <Tiempo NoSegundo="[NumeroEntero]">
                    <LineaEnsamblaje NoLinea="[NumeroEntero]">
                        <!--[Texto]--> *Representa la acción en la línea (movimiento,
                                         no hacer nada ó ensamblar)
                    </LineaEnsamblaje>
                    …
                </Tiempo>
                …
            </ElaboracionOptima>
        </Producto>
        …
    </ListadoProductos>
</SalidaSimulacion>
```

REPORTES

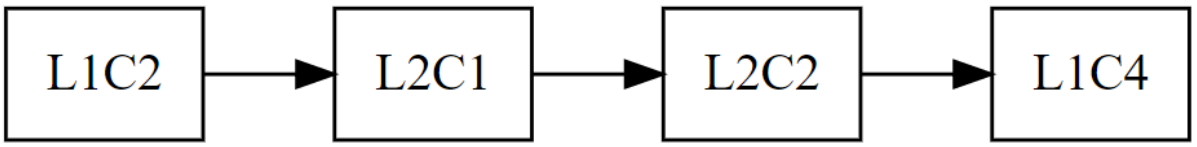
Reporte HTML - ReporteSimulación.html

Para cada producto ensamblado dentro de cada archivo de simulación, se debe generar un reporte de simulación que detalla el proceso de ensamblado del producto; o si en dado caso se desea simular la elaboración de un producto específico en un tiempo *t* mediante la interfaz de usuario.

Este reporte se deberá generar por cada simulación realizada, y deberá ser en formato HTML; queda a discreción del estudiante como representar la información de cada producto ensamblado, pero tome a consideración que debe verse el movimiento de cada línea y el total que llevo ensamblar el producto. Puede usar como referencia la figura No.1 - Tabla de resultados.

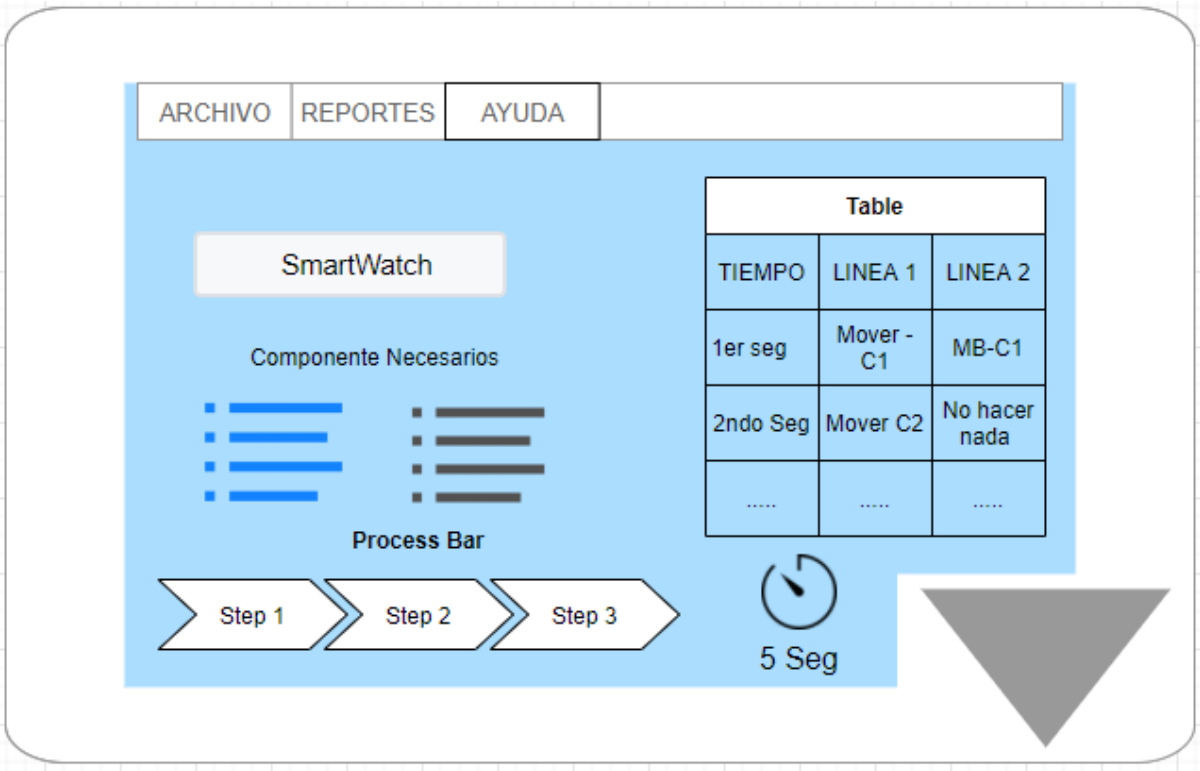
Reporte de cola de secuencia

Utilizando la herramienta Graphviz, se deberá crear un grafo mostrando el estado de la cola de secuencia de trabajo descrita anteriormente. Este grafo debe poder generarse en cualquier momento y debe mostrar el estado de la cola de secuencia en ese momento por medio de una opción en la interfaz de usuario que será descrita en la siguiente sección. El siguiente es un ejemplo de una secuencia de trabajo graficada.



INTERFAZ DE USUARIO

Se debe crear una interfaz de usuario, en la que se puedan gestionar todas las acciones necesarias para la ejecución lógica del proyecto, a continuación, se muestra una interfaz únicamente con fines demostrativos, queda a discreción del estudiante como elaborar su propio diseño para la aplicación.



Las únicas consideraciones son las siguientes:

- Debe existir una opción para cargar archivos de la configuración de máquinas.
- Debe existir una opción para poder cargar archivos de simulación, para poder ser procesados varios productos especificados por dicho archivo.
- Se debe poder seleccionar un producto cualquiera de los precargados mediante el archivo de entrada, para poder ser ensamblado.
- Mostrar de manera visual el proceso que se está ejecutando en el momento *t*, posteriormente generar el reporte de este proceso.
- Apartado de ayuda (Incluir información del estudiante, Acerca de la aplicación).
- Se tomará en cuenta la creatividad del estudiante.
- Sección de reportes.

CONSIDERACIONES

Se deberá realizar la implementación utilizando programación orientada a objetos, algoritmos desarrollados por el estudiante e implementación de estructuras a través de Tipos de Dato Abstracto (TDA); que permita almacenar la información de los archivos de entrada y poder interactuar con dicha información. El estudiante deberá abstraer la información y definir qué estructuras implementar que le faciliten la solución. Por lo que puede implementar pilas, colas, listas simples, dobles, circulares o listas de listas para poder solventar el proyecto. **No está permitido el uso de estructuras implementadas de Python(list, dict, tuple, set).**

Debe utilizarse versionamiento para el desarrollo del proyecto. Se utilizará la plataforma **Github** en la cual se debe crear un repositorio en el que se gestionará el proyecto. Se deben realizar **4 releases** o versiones del proyecto (se recomienda realizar una por semana del tiempo disponible) **mínimo**. Se deberá agregar a sus respectivos auxiliares como colaboradores del repositorio. El último release será el release final y se deberá de realizar antes de entregar el proyecto en la fecha estipulada.

DOCUMENTACIÓN

Para que el proyecto sea calificado, el estudiante deberá entregar la documentación utilizando el formato de ensayo definido para el curso. Para este ensayo, se debe de tener un mínimo de 4 y un máximo de 7 páginas de contenido, este máximo no incluye los apéndices o anexos donde se pueden mostrar modelos y diseños utilizados para construir la solución. Es obligatorio incluir el diagrama de clases que modela la solución de software presentada por el estudiante.

Formato del ensayo: https://drive.google.com/file/d/1jt3wtyTYPD-Gi27W9cJPaoKiLMIwIVVY/view?usp=sharing

RESTRICCIONES

- Solo se permitirá la utilización de los IDEs discutidos en el laboratorio.
- Uso de TDA implementados por el estudiante obligatorio; el uso de estructuras de Python(list, dict, tuple, set) resultará en penalización del 100% de la nota.
- Uso obligatorio de programación orientada a objetos (POO) desarrollada por completo por el estudiante. De no cumplir con la restricción, no se tendrá derecho a calificación
- El nombre del repositorio debe de ser **IPC2_Proyecto2_#Carnet.**
- El estudiante debe entregar la documentación solicitada para poder optar a la calificación.
- Los archivos de entrada no podrán modificarse.
- Los archivos de salida deben llevar la estructura mostrada en el enunciado obligatoriamente.
- Deben existir 4 releases mínimo, podría ser uno por cada semana, de esta manera se corrobora el avance continuo del proyecto.
- Se calificará de los cambios realizados en el último release hasta la fecha de entrega. Los cambios realizados después de ese release no se tomarán en cuenta.
- Cualquier caso de copia parcial o total tendrá una nota de 0 y será reportada a escuela.
- Para dudas concernientes al proyecto se utilizarán los foros en UEDI de manera que todos los estudiantes puedan ver las preguntas y las posteriores respuestas.
- **NO HABRÁ PRÓRROGA.**

ENTREGA

- La entrega será el **25 de septiembre,** a más tardar a las 11:59 pm.
- La entrega será por medio de la UEDI.
- La documentación debe estar subida en el repositorio en una carpeta separada.
- Para entregar el proyecto en UEDI se deberá subir un archivo de texto con el link del repositorio.