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. José Manuel Ruiz Juárez
Ing. Dennis Stanley Barrios Gonzalez
Ing. Edwin Estuardo Zapeta Gómez
Ing. Fernando José Paz González

Tutores de curso:
Dayana Alejandra Reyes Rodríguez
Andrea María Cabrera Rosito
Paula Gabriela García Reinoso
Piter Angel Esaú Valiente de León
Mario César Morán Porras
Denilson Florentín de León Aguilar

PROYECTO 2

OBJETIVO GENERAL

Modelar, documentar e implementar una solución al problema que se plantea utilizando las herramientas de desarrollo presentadas en clase y laboratorio.

OBJETIVOS ESPECÍFICOS

- Implementar una solución utilizando el lenguaje de programación Python.
- Utilizar estructuras de programación secuenciales, cíclicas y condicionales.
- Generar reportes con la herramienta Graphviz.
- Manipular archivos XML.
- Utilizar los conceptos de TDA y aplicarlos a memoria dinámica.
- Utilizar estructuras de programación propias.
- Utilizar el paradigma de programación orientada a objetos.

ENUNCIADO

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.

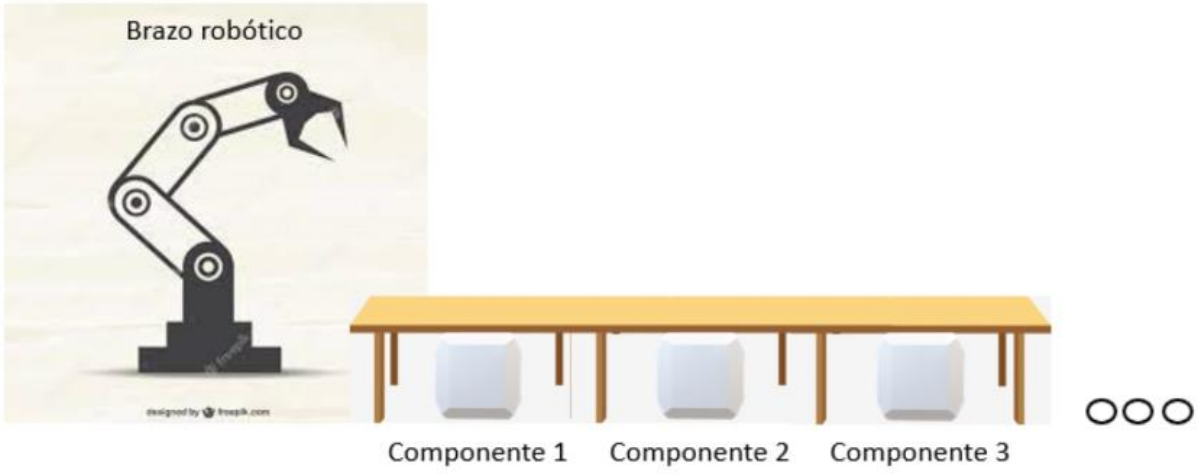


Figura No. 1 – Una línea de ensamblaje

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 debe ser ensamblado 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 para construir un SmartWatch

Archivo de Entrada

El archivo XML para configurar las máquinas y los productos que cada máquina podrá elaborar tendrá la siguiente estructura:

entrada.xml

```
<?xml version="1.0"?>
<ListaMaquinas>
  <Maquina>
    <NombreMaquina> [Texto] </NombreMaquina>
    <CantidadLineasProduccion> n </CantidadLineasProduccion>
    <CantidadComponentes> m </CantidadComponentes>
    <TiempoEnsamblaje> x </TiempoEnsamblaje>
    <ListadoProductos>
      <Producto>
        <nombre> [Texto] </nombre>
        <elaboracion>
          [Texto][1]
        </elaboracion>
      </Producto>
      …
    </ListadoProductos>
  </Maquina>
  …
</ListaMaquinas>
```

* n y m representan valores numéricos mayores que 0 y menores que 1000
** x representa un valor numérico mayor que 0

Debe considerar que puede subir más de un archivo de entrada para configurar nuevas máquinas o nuevos productos a las máquinas ya existentes.

Archivo de Salida

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

salida.xml

```
<?xml version="1.0"?>
<SalidaSimulacion>
  <Maquina>
    <Nombre> [Texto][2] </Nombre>
    <ListadoProductos>
      <Producto>
        <Nombre> [Texto] </Nombre>
```

[1] Formato: $L_{i1}C_{j1}$ $L_{i2}C_{j2}$ $L_{i3}C_{j3}$ … $L_{in}C_{jn}$ donde cada elemento $L_{in}C_{jn}$ representa la línea y componente que deben ser ensamblados para elaborar el producto.
[2] Nombre de la máquina a la que se le simulará la elaboración de sus productos.

```
        <TiempoTotal> [NumeroEntero>0] </TiempoTotal>
        <ElaboracionOptima>
          <Tiempo NoSegundo="[NumeroEntero]">
            <LineaEnsamblaje NoLinea="[NumeroEntero]">
              [Texto][3]
            </LineaEnsamblaje>
            …
          </Tiempo>
          …
        </ElaboracionOptima>
      </Producto>
      …
    </ListadoProductos>
  </Maquina>
  …
</SalidaSimulacion>
```

REPORTES

Reporte HTML - ReporteSimulación.html

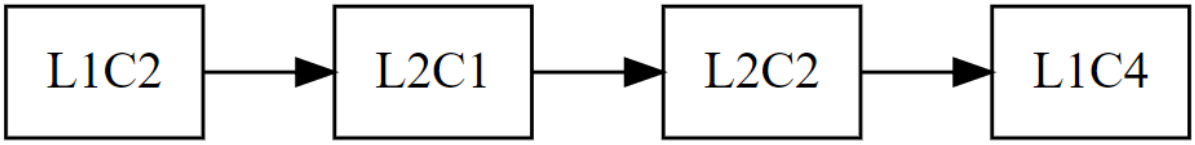
Para cada producto ensamblado dentro de cada archivo de simulación, se debe generar un reporte de simulación que detalla el proceso de ensamblado del producto; o si en dado caso se desea simular la elaboración de un producto específico en un tiempo *t* mediante la interfaz de usuario.

Este reporte se deberá generar por cada simulación realizada (generación del archivo de salida), 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 de producción y el tiempo total que llevo ensamblar el producto. Puede usar como referencia la figura No.2 - Tabla de resultados para construir un SmartWatch.

Reporte de TDAs

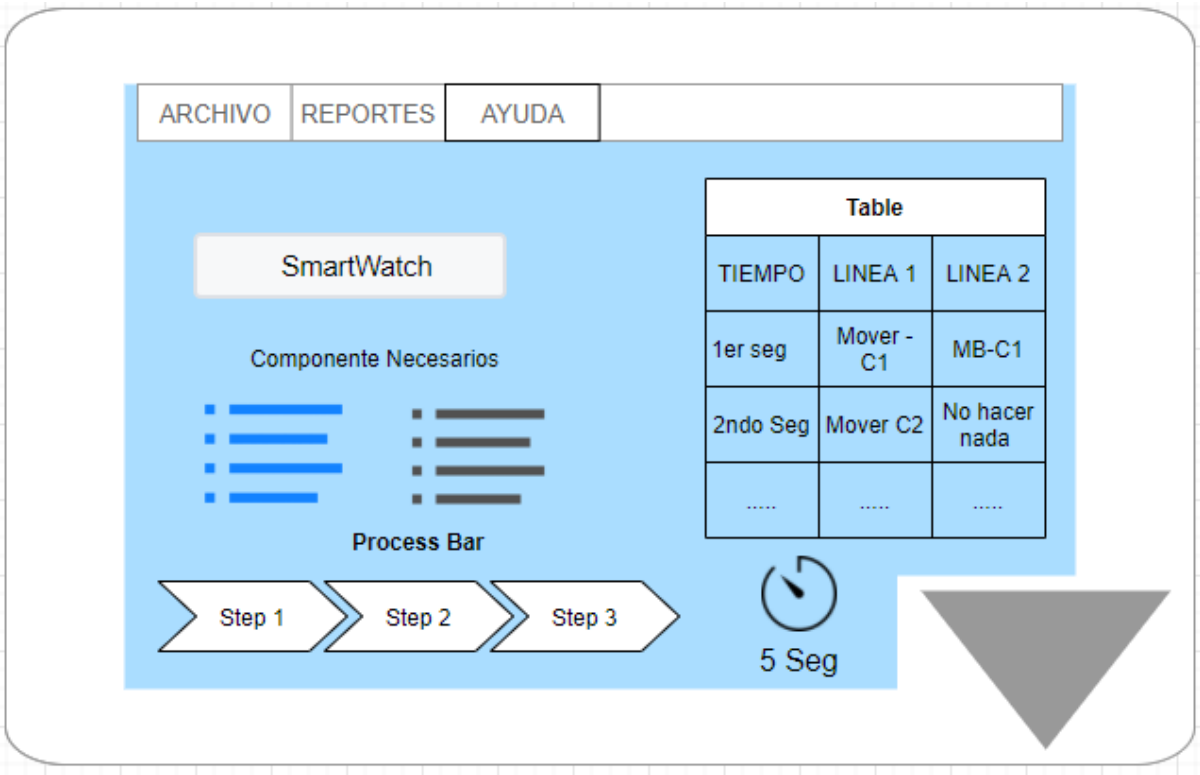
Utilizando la herramienta Graphviz, se deberá crear un grafo mostrando el estado de los TDAs utilizados para presentar la secuencia de trabajo descrita anteriormente. Este grafo debe poder generarse en cualquier momento y debe mostrar el estado de los TDAs en ese momento por medio de una opción en la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

[3] Representa la acción en la línea (mover brazo hacia componente “x”, no hacer nada o ensamblar)


L1C2
L2C1
L2C2
L1C4

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 que inicialice todos los datos del programa para poder cargar archivos de entrada en un entorno sin datos.
- Debe existir una opción para cargar archivos de la configuración de máquinas y productos (archivo de entrada).
- Se debe poder seleccionar una máquina y un producto cualquiera de los precargados mediante el archivo de entrada, para poder simular su construcción.
    - Mostrar el tiempo óptimo para construir el producto y los pasos a realizar para ensamblarlo. (ver figura 2 con ejemplo de la forma en que se muestran las instrucciones para construir un producto)
    - Opción para mostrar los pasos realizados en un tiempo “t” definido por el usuario. (ver figura 2 con ejemplo de la forma en que se muestran las instrucciones para construir un producto)
    - Generar reporte html de la elaboración de este producto
    - Generar gráfica de estado de los TDAs en un tiempo “t” utilizados para la solución para elaboración de productos.
- Generar archivo de salida con simulación de todos los productos configurados y su respectivo reporte html.

- Apartado de ayuda (Incluir información del estudiante, Acerca de la aplicación y enlace hacia la documentación).
- Se tomará en cuenta la creatividad del estudiante.
- La interfaz de usuario debe ser web utilizando Flask.

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) propios del estudiante; 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 propias 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 su respectivo auxiliar como colaborador 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. En el caso del proyecto, el ensayo debe tener entre 4 y 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 y los diagramas de actividades con los principales algoritmos implementados en la solución.

RESTRICCIONES

- Solo se permitirá la utilización de los IDEs discutidos en el laboratorio.
- 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 uno por cada semana, de esta manera se corrobora el avance continuo del proyecto. **Se definirá una penalización por cada release faltante**.
- Se calificará de los cambios realizados en el cuarto release. 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 de Ciencias y Sistemas.

- 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.
- Utilización del framework Flask para desarrollar la interfaz de usuario.
- **NO HABRÁ PRÓRROGA.**

ENTREGA

- La entrega será el **2 de octubre** a las 11:59 pm como máximo.
- 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.