Estructuras de Datos I

PROYECTO FINAL DE CLASE: IMPLEMENTACIÓN DE ESTRUCTURAS DE DATOS

OBJETIVOS

Fortalecer los conocimientos sobre las estructuras de datos para cada tipo de dato abstracto, a través de la implementación de un proyecto de software.

DESCRIPCIÓN DE ACTIVIDAD

Fecha máxima para publicación en foro de proyecto: **Domingo 02 de Febrero 2020 23h00**

Fecha de entrega primer avance: **Lunes 24 de Febrero 2020 10h00**

Fecha de entega final: **Lunes 23 de Marzo 2020 10h00**

En ambos casos, no se evaluarán trabajos recibidos después de la fecha.

Valor: **17.5% :**

- **Participación en Foro: 1.0 %**
- **Primer Avance: 2.5%**
- **Entrega Final: 14 %**

Tipo de Entrega: **Individual**

Para desarrollar el proyecto, debería seguir los siguientes pasos:

1. Determinar caso o escenario para el desarrollo de proyecto.
2. Definición de estructura de almacenamiento y campos.
3. Desarrollo. Creación de interfaz entre archivo y estructura.
4. Documentación del proyecto.

Paso 1: Determinar caso o escenario para el desarrollo de proyecto.

Un componente importante para el desarrollo del proyecto es la investigación sobre las estructuras que utilizará ya que todos los escenarios planteados utilizan dos estructuras de datos, lo que hace imprescindible que el alumno investigue con suficiente tiempo sobre el comportamiento de estas estructuras (todas manejan las operaciones de ingreso, eliminación y consulta de una manera diferente). No espere a que la estructura sea vista en clase o hasta la última semana para empezar su proyecto.

Para el desarrollo del proyecto de clase debe seleccionar uno de los tres casos siguientes:

Caso A: Organigrama y tareas

Estructuras a utilizar:

- Árbol (organigrama)
- Cola (Tareas por puesto)

Descripción:

Toda organización mantiene una descripción visual del orden jerárquico de todos los puestos de todos los empleados (organigrama). Cada puesto tiene tareas específicas que debe desarrollar.

Su trabajo consiste en representar mediante un árbol (no árbol binario, es decir se puede tratar de un árbol donde cada nodo tiene uno o más hijos) el organigrama de una empresa y sus diferentes puestos. Para cada puesto se podrán ingresar un conjunto de tareas que corresponden a las tareas asignadas a ese puesto específico.

Deberá desarrollar como mínimo los siguientes métodos:

1. Ver organigrama
    a. Este método debe desplegar el organigrama con todos los puestos disponibles donde se indique las relaciones jerárquicas.
2. Crear subordinado (***puesto superior***, ***puesto nuevo***)
    a. Dado un ***puesto superior*** debe crear un ***puesto nuevo*** que dependa del puesto superior indicado.
3. Eliminar subordinado (***puesto***)
    a. Dado un ***puesto***, debe eliminar el puesto del organigrama. Si se elimina un puesto que tenga dependientes todos los puestos dependientes deben pasar a depender del puesto jefe del eliminado.
4. Contar puestos
    a. Devuelve la cantidad de puestos que tiene el organigrama.
5. Agregar tarea (***puesto***, ***tarea***)
    a. Para un ***puesto*** específico, agrega una ***tarea***.
6. Eliminar tarea (***puesto***)
    a. Para un ***puesto*** específico, elimina la tarea correspondiente al orden en que se eliminan tareas en una cola.
7. Grado organigrama
    a. Devuelve la el grado del organigrama. Grado es la cantidad máxima de hijos presente en el árbol.
8. Grado puesto (***puesto***)
    a. Para un ***puesto*** específico devuelve la cantidad de puestos subordinados (hijos en el árbol).
9. Listar tareas (***puesto***)
    a. Lista todas las tareas que pertenecen a un ***puesto*** específico.
10. Determinar empleado mas bajo
    a. Buscar en todo el árbol el puesto que se encuentre en el nivel mas profundo, es decir que a partir del primer puesto del organigrama tenga que pasar por más subordinados para llegar a él. Devuelve el puesto y la cantidad de puestos recorridos.

Tenga en cuenta lo siguiente:

- En el nivel superior del organigrama solo debe tener 1 puesto.
- Todo puesto puede tener 1 o más dependientes.
- El manejo de puestos debe basarse en el manejo de un árbol no binario.
- El manejo de tareas debe basarse en el manejo de colas.

Caso B: Clientes herederos y herencias

Estructuras a utilizar:

- Montículo (Herederos)
- Pila (Clientes/ Familias)

Considere una pila donde se almacenan varios listados de clientes (familias) y el monto de una herencia en particular para cada familia. Al acceder a cada cliente es posible almacenar el listado de todos los herederos de esa familia (un montículo por cada familia). Dentro del montículo los herederos se ubican basados en el monto heredado. Este monto heredado puede ser modificado en el tiempo de ejecución del programa por lo que cada montículo puede cambiar.

Deberá desarrollar como mínimo los siguientes métodos:

1. Consultar clientes
    a. Devuelve la pila con todos los clientes y su respectivo monto de herencia.
2. Agrega cliente (***familia, monto***)
    a. Agrega una ***familia*** y su ***monto*** a la pila de clientes.
3. Borra cliente (***familia***)
    a. Elimina una ***familia*** de la pila de clientes. La familia a eliminar debe ser la del orden específico en la pila (el último dato ingresado).
4. Agregar heredero (***familia***, ***monto_heredero***, ***nombre_heredero***)
    a. Para una familia que ya se encuentra dentro de la pila, agrega un heredero cuyo nombre es ***nombre_heredero***, así como su monto heredado (***monto_heredero***). Este monto debe ser menor al total de monto de la familia (que se encuentra en el registro de la pila de cada). Si al momento del ingreso el montículo ya se encuentra con algún dato debe reacomodar el montículo si fuera necesario.
5. Verificar monto heredado (***familia***, ***monto***)
    a. Para una ***familia*** específica verifica si el ***monto*** ingresado puede ser heredado por alguno de los herederos. Para esto debe analizar el monto registrado en la pila menos la suma de los montos de todos los herederos que se encuentren dentro del montículo de esa familia.
6. Consulta herederos (***familia***)
    a. Devuelve la lista de todos los herederos y el monto heredado que pertenecen a una ***familia*** específica.
7. Eliminar heredero (***nombre_heredero***, ***familia***)
    a. Elimina el heredero con nombre_heredero del montículo de su familia. Recuerde que si es necesario reacomodar el montículo debe hacerlo.
8. Mejor / peor herencia
    a. Recorre todos los montículos y retorna quien es el heredero que tiene la mejor o peor herencia, así como la familia a la que pertenece.
9. Cambiar herencia (***nombre_heredero***, ***familia***, ***nueva_herencia***)

   a. Para una ***familia*** específica, cambia el monto de la herencia a ***nueva_herencia*** para el ***nombre_heredero.*** Recuerde que si es necesario tendrá que reacomodar el montículo.
10. Buscar heredero (***familia o heredero***)
   a. Busca una ***familia*** o ***heredero*** en la pila o nombre montículo y devuelve el monto a heredar y si es familia o heredero.

Tenga en cuenta lo siguiente:

- El manejo de clientes (familias) debe hacerse en una pila
- El manejo de herederos debe hacerse en montículos, recuerde que hay un montículo por cada familia.

Caso C: Municipio y registro de personas

Estructuras a utilizar:

- Grafo no dirigido (municipio)
- Listas enlazadas (registro de personas)

Descripción:

En un determinado país, las diez principales ciudades se representan a través de un grafo no dirigido. Aunque no todas las ciudades se encuentran conectadas directamente unas con otras, si es posible recorrer todas las ciudades utilizando ciudades intermedias. La distancia entre cada par de ciudades debe ser diferente. Al interior de cada ciudad (un nodo del grafo) es posible almacenar una lista enlazada con los habitantes de esa ciudad, para fines prácticos se limitará a 20 personas por ciudad.

Deberá desarrollar como mínimo los siguientes métodos:

1. Ver ciudades
   a. Despliega un grafo no dirigido con todas las ciudades (deben estar predefinidas) y las distancias entre cada uno si la hubiera. Puede representarse a través de una matriz.
2. Agregar camino (***ciudad a***, ***ciudad b***, ***distancia***)
   a. Dadas dos ciudades, conecta ***ciudad a*** con ***ciudad b*** con una arista que tiene un peso igual a ***distancia***. Si el camino ya existiera reemplaza el valor de la arista con la nueva ***distancia***.
3. Eliminar camino ***(ciudad a***, ***ciudad b***)
   a. Elimina el camino existente de la ***ciudad a*** a la ***ciudad b***, debe verificar que si elimina el camino directo siempre existe alguna manera de llegar desde la ciudad a hasta la ciudad b.
4. Existe camino (***ciudad a***, ***ciudad b***)
   a. Verifica si existe un camino desde la ***ciudad a*** hasta la ***ciudad b***. Devuelve verdadero o falso.
5. Agregar persona (***ciudad***, ***persona***)
   a. Para una ciudad ***específica***, agrega una ***persona*** (nombre e identificador) a la lista de personas de la ciudad.

6. Eliminar persona (***ciudad***, ***persona***)
    a. Data una ***ciudad*** específica, elimina una ***persona***.
7. Buscar persona (***persona***)
    a. Busca una ***persona*** en todas las ciudades. Devuelve la ciudad a la que pertence, el identificador y el orden en la lista.
8. Contar personas (***ciudad***)
    a. Regresa la cantidad de personas que hay registradas en la ***ciudad.***
9. Ver personas (***ciudad***)
    a. Devuelva todas las personas que se encuentran registradas en la ***ciudad*** en el orden de la lista.
10. Ordenar personas (***ciudad***)
    a. Devuelve todas las personas que se encuentran registradas en la ***ciudad*** ordenadas de acuerdo al identificador de las personas.

Tenga en cuenta lo siguiente:

- Al iniciar el programa debe tener precargada varias ciudades (al menos 7).
- En todo momento todas las ciudades deben permanecer conectadas.
- Las ciudades deben manejarse utilizando las operaciones de un grafo no dirigido.
- Las personas dentro de cada ciudad deben de manejarse utilizando operaciones de lista.

Paso 3: Desarrollo. Creación de interfaz entre archivo y estructura.

Basado en el caso o escenario seleccionado, cada alumno también deberá definir el tipo de estructura de almacenamiento que utilizará dentro de su proyecto para guardar la información del primer paso. Puede utilizar archivos planos, arreglos, base de datos, etc… la elección dependará de su experiencia. El lenguaje de programación también queda a libre elección del alumno.

Una vez que se haya seleccionado el tipo de estructura de almacenamiento debe presentar (en el mismo foro del proyecto) el diseño que tendrá cada registro que se almacenará en la estructura seleccionada. Todos los campos deberán almacenarse en la estructura de almacenamiento, pero el enlace entre el almacenamiento (Archivo / arreglo / tabla) y las estructuras de datos será ese campo único numérico. La idea es que los datos siempre se encuentren cargados y no sea necesario el reingreso de datos cada vez que se ejecute el programa.

Si decide utilizar base de datos debe proporcionar en la entrega final del proyecto los scripts para la creación de la base de datos, tablas, procedimientos y poblado de datos. Debe también configurar su programa de manera que pueda conectarse a un servidor local sin rutas absolutas.

En esta etapa debe realizar toda la codificación para que el programa sea funcional y se ejecuten los métodos que se apliquen al caso que usted seleccionó. Se sugiere la incorporación de un menu para interactuar con cada una de las opciones del programa.

Paso 4: Documentación del proyecto

Como parte del proyecto deberán entregar un documento de proyecto que contenga lo siguiente:

1. Portada (Asignatura, Docente, Periodo, Numero de cuenta y nombre, fecha de entrega Campus CEUTEC)
2. Escenario o caso seleccionado y descripción del tipo de almacenamiento seleccionado, así como sus campos: archivo / arreglo / tabla y para cada campo descripción, tipo de campo, tamaño aproximado, etc.. – una página
3. Lenguaje de programación utilizado y requerimientos mínimos para ejecución del programa. – media página.
4. Guía de instalación: pasos necesarios para instalar el software. Debe ser lo mas explícito posible.
5. Pseudocódigo o diagrama de flujo por cada una de los métodos de las diferentes estructuras. Recuerde que el pseudocódigo no es el código fuente del programa.
6. Breve descripción del código: principales rutinas utilizadas: nombre en código y descripción de lo que hacen. **No es necesario poner todo el código del programa, eso estará incluido en el código fuente.** Lo que se requiere es una breve descripción de lo que hace cada rutina, que parámetros recibe y que devuelve.
7. Bitácora del proyecto: detalle de fecha, actividad, duración de cada una de las tareas realizadas para el desarrollo del proyecto.
8. **Enlace a vídeo en youtube con instalación y ejecución de opciones del programa: deberán incluir un vídeo de no mas de 8 minutos que incluya el proceso de instalación del programa, así como la prueba de las opciones.**
9. Finalmente incluir una sección de Dificultades encontradas en la elaboración del proyecto y Recomendaciones sobre el desarrollo del proyecto.

Recomendación: *No espere el final del periodo para empezar la documentación. La documentación del proyecto debe empezar el día en que seleccione el escenario o caso y su publicación en el foro.*

RUBRICA DE CALIFICACIÓN

Primer Avance: Lunes 23 de Febrero2020 10h00

Valor: 2.5 %

Criterio	Ponderación
Documento del proyecto (Portada 5 %, Datos a utilizar 15%, Caso seleccionado y descripción del archivo de datos (campos, descripción, tipo, longitud) 20%, Lenguaje de programación 10 %, Estructuras seleccionadas y la manera en que se utilizarán en el proyecto 40%, Bitácora del proyecto 10%)	100%
Total	100%

Entrega Final: Lunes 24 de Marzo 2020 10h00

Valor: 14.0 %

Criterio	Ponderación
Documento del proyecto (Datos a utilizar, caso a resolver, descripción de archivo/tabla/campos 5%, Lenguaje de programación y requerimientos para ejecución 5 %, Guía de Instalación 5%, Pseudocódigo o diagrama de flujo de los métodos 30%, Breve descripción del código 15%, Bitácora del proyecto 5% , Vídeo en youtube con instalación y operaciones específicas 35%)	10%
Estructura del Grupo A y sus métodos	35%
Estructura del Grupo B y sus métodos	40%
Enlace estructuras de datos con estructura de almacenamiento de registros	15%
Total	100%

ESPECIFICACIONES DE ENTREGA

El proyecto de clase se entregará en tres fases:

1. Publicación en el Foro
   a. Fecha: : **Domingo 03 de Febrero 2020 23h00**
   b. Favor consultar el Foro de Asignación de Proyecto en el Aula Virtual (Semana 1)

2. Entrega de Primer Avance
   a. Fecha: **Lunes 23 de Febrero 2020 23h00 – No habrá prórrogas**
   b. Entrega: subir en Blackboard en el espacio correspondiente (Semana 5 / Actividades). Se harán correcciones de ser necesarias en la misma plataforma.
   c. Deberán subir un documento que incluya lo siguiente:
      i. Portada
      ii. Conjunto de datos abiertos, sitio de origen y descripción de los datos a utilizar.
      iii. Caso seleccionado y descripción de archivo de datos (campo, descripción, tipo, longitud)
      iii. Lenguaje de programación seleccionado
      iv. Estructuras seleccionadas y la manera en que se utilizarán en el proyecto. Debe indicar cuales son las estructuras que se utilizarán y de que manera. También deberá indicar cuales serán los métodos que desarrollará para las estructuras seleccionadas.
      v. Bitácora del proyecto hasta el momento. (fecha, actividad, duración)

2. Entrega final del Proyecto
   a. Fecha **Lunes 24 de Marzo 2020 23h00 – No habrá prórrogas**
   b. Entrega: subir en Blackboard en el espacio correspondiente (Semana 10 / Actividades)
   c. Presentación: En el aula de clase, cada alumno será responsable de llevar el equipo
   software para su prueba. Solo algunos proyectos serán seleccionados para realizar su presentación en el aula.
   d. En el aula virtual deberán subir un ZIP por cada proyecto conteniendo lo siguiente:
      - Archivos fuentes

- Archivo ejecutable / instalador (**es responsabilidad de quien entrega el proyecto asegurarse de que se entrega un ejecutable corriendo en plataforma Windows 7 en adelante y que se cuentan con todas las librerías integradas).** Archivos ejecutables que no funcionan, se considerarán como proyectos incompletos y esto tendrá una incidencia en la nota final del proyecto).
- Guía de instalación: deben incluir una guía rápida y completa de los pasos que se deben seguir para que se pueda instalar el proyecto. (máximo 2 páginas) Asuma que el usuario que lo ejecutará tiene un nivel básico de conocimientos de computación.
- Documento del proyecto de acuerdo a lo siguiente:
  - Portada
  - Caso seleccionado y descripción del archivo de datos
  - Lenguaje de programación y requerimientos para ejecución
  - Pseudocódigo o diagrama de flujo
  - Breve descripción del código
  - Bitácora del proyecto
  - Dificultades y recomendaciones para el desarrollo del proyecto
  - Link a vídeo en youtube con instalación

- Dudas o comentarios Via FORO: Proyecto I en el Aula Virtual.