Examen Parcial de Algoritmos y Estructuras de Datos

Profesor: Walter Cueva Chávez

Duración: 170 minutos

Hora Inicio: 15:20pm

Nro. Examen: 17

Instrucciones:

- El examen tiene 2 partes: un cuestionario para marcar basado en un banco de preguntas aleatorio (20 minutos) y desarrollo de un caso práctico (150 minutos).

- Es libre de usar el entorno de desarrollo integrado (IDE) que Ud. considere.
- En la duración del examen se están contemplando ***10 últimos minutos*** para adjuntar el archivo de código fuente. Tenga en cuenta este aspecto en caso de contar con dificultades de acceso a internet.
- En el proceso de corrección de exámenes se utilizará un sistema de detección de plagio (sintaxis y semántica) así como corrección, por lo que es indispensable que tenga las siguientes consideraciones:
    - Verifique que su archivo sea adjuntado en la actividad indicada mediante aula virtual (dentro del horario establecido), para que su examen pueda ser calificado, Además es su responsabilidad guardar apropiadamente su examen.
    - Todas las soluciones deben contar con código de comprobación.
    - Archivo con código que no compila, no será considerado en su totalidad.
    - Soluciones misteriosas o supuestas invalidan la respuesta.
    - El nombre del archivo solución debe estar conformado por el #Caso_, primer nombre del alumno y el primer apellido del alumno**".cpp"** (por ejemplo.: 1_JorgeMejia.cpp).
    - **Solamente** enviar un archivo de solución con extensión **.cpp**.

- El alumno debe dedicar los primeros 20 minutos a revisar las preguntas del examen y de presentarse alguna duda enviar un correo los profesores según la sección

  Secciones:
  CC31, CC32, CC33: Walter Cueva Chavez **pcsiwcue@upc.edu.pe**
  SS31, SS32: Carlos Jara García: **pcsicjar@upc.edu.pe**
  SV31, SX31: Luis Vives Garnique: **pcsilviv@upc.edu.pe**
- Los profesores en mención solo recibirán correos provenientes de las cuentas **UPC**, de

ninguna manera se recibirán correos de cuentas públicas.

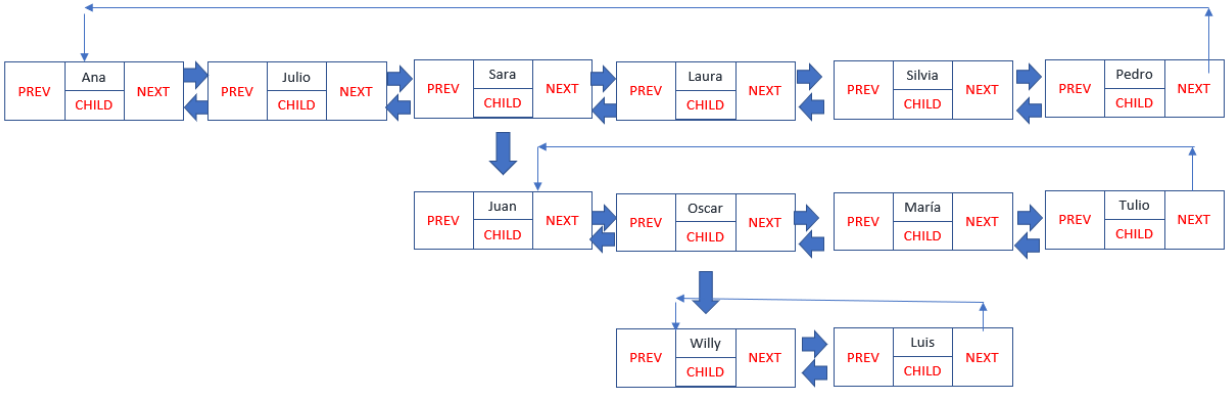
- Cuando realice una pregunta al equipo técnico, indicar el nro de examen que le corresponde para identificarlo con mayor rapidez.
- Ante problemas técnicos, debe de forma obligatoria adjuntar evidencias de este, como capturas de pantalla, videos, fotos, etc. Siendo requisito fundamental que, en cada evidencia se pueda apreciar claramente la fecha y hora del sistema operativo del computador donde el alumno está rindiendo el examen.
- Puede usar el código desarrollado en clases
- Los problemas técnicos se recibirán **como máximo 15 minutos** culminado el examen.
- Estamos seguros de que cada uno realizará su examen. Sin embargo, para evitar ¿cualquier perspicacia, le recomendamos leer sus reglamentos de estudios y disciplina del alumno, en el cual se indican las faltas y las sanciones en el caso de la copia de exámenes (falta contra la probidad académica). La detección de plagio parcial o totalmente será penalizada con la nota de **cero en todo el examen (parte 1 y 2) para todos los alumnos que participen**. Alentamos el desarrollo honesto de su examen.
- En el momento de la revisión, ante dudas razonable de plagio, puede ser citado mediante el correo institucional con un día de anticipación para la exposición oral (con cámara activa) de su desarrollo del examen mediante el aula virtual y ésta será grabada.
- Se habilitará una actividad en el aula virtual con Nombre **"ExamenParcial"** para el envío de la solución de la segunda parte.

- **No usar biblioteca STL en su desarrollo y solamente considerar los métodos necesarios en su implementación además es mandatorio el uso de POO.**

Lista Doble Circular

En una lista doblemente enlazada circular especializada donde tiene punteros al siguiente, anterior y un puntero hijo (puede apuntar a una lista doblemente enlazada separada). Estas listas hijas pueden tener uno o más hijos propios, y así sucesivamente, para producir una estructura de datos multinivel, como se muestra en el ejemplo siguiente.

Entrada: Se proporciona el inicio del primer nivel de la lista.



Salida: Lista única.
[Ana,Julio,Sara,Juan,Oscar,Willy,Luis,Maria,Tulio,Laura,Silvia,Pedro]

Convierta la lista para que todos los nodos aparezcan en una lista doblemente enlazada de un solo nivel.

Ejemplo 1:

Entrada:
Inicio = [Ana, Julio, nullptr,Sara]
Salida: [Ana,Sara,Julio]
Explicación:
La lista enlazada multinivel de entrada es la siguiente:
Ana--Julio—nullptr
|
Sara-- nullptr

Ejemplo 2:

Entrada:
inicio = [ ]
Salida: []

Explicación:

Si contamos con la lista jerárquica

Entrada:

```
Ana—Julio –Sara –Laura –Silvia—Pedro—
        |
       Juan –Oscar –Maria—Tulio—
           |
          Willy—Luis—
```

La serialización de cada fila es la siguiente:

```
[Ana,Julio,Sara,Laura,Silvia,Pedro]
[Juan,Oscar,Maria,Tulio]
[Willy,Luis]
```

Para serializar todos los niveles juntos añadiremos nulos en cada nivel para significar que ningún nodo se conecta al nodo superior del nivel anterior.

La serialización de cada fila es la siguiente:

```
[Ana,Julio,Sara,Laura,Silvia,Pedro]
[nullptr, nullptr,Juan,Oscar,Maria,Tulio]
[nullptr, Willy, Luis]
```

Uniendo las filas obtenemos, que es la entrada del algoritmo:

```
[Ana,Julio,Sara,Laura,Silvia,Pedro, nullptr, nullptr, Juan,Oscar,Maria,Tulio, nullptr, Willy, Luis]
```

Puntaje

1. (1 punto) leer desde archivo e insertar elementos en la lista, dicho archivo con nombre **input.txt** debe contener la entrada, basado en el ejemplo anterior sería:

```
Ana,Julio,Sara,Laura,Silvia,Pedro, nullptr, nullptr, nullptr,Juan,Oscar,Maria,Tulio, nullptr, Willy, Luis]
```

2. (4 punto) Realizar la serialización de los datos según el caso descrito, considerar la menor complejidad en la implementación, la estructura de datos debe ser genérica.
3. (2 punto) Usar un iterador para recorrer la lista.
4. (2 punto) El resultado debería guardar en un archivo **output.txt** en la primera línea debe estar el resultado y en la segunda línea mostrar su complejidad detallada del método anterior, por ejemplo:

Ana,Julio,Sara,Juan,Oscar,Willy,Luis,Maria,Tulio,Laura,Silvia,Pedro
n^2+15n+10