Programación Avanzada (1113) UNLaM

Evaluación Final Virtual Marzo 02/03/2021

Nota para su resolución y entrega:

Para rendir examen debe unirse al grupo de Teams 1113-Final 02/03/2021

La cámara debe permanecer prendida durante el examen y tener un micrófono para interactuar con el docente en la parte oral y recibir la devolución del examen.

La resolución del examen será de 2 hs. A los 30 minutos de comenzado el examen debe entregar el ejercicio 1

La aprobación del ejercicio 1 y 2 son condición necesaria para la aprobación del examen

Para enviar el examen, cambie el nombre del archivo colocando su dni-apellido-nombre.

Pegue en el archivo que entrega una foto de su DNI, para acreditar su identidad.

Ejercicio 1: Java complejidad computacional

Frases dobladas

Una técnica conocida por los enamorados estudiantes para enviar mensajes de amor consiste en utilizar una frase aparentemente inocua o incluso sin sentido, escrita en una tira de papel. Esta frase, doblada por los sitios indicados, permite develar un segundo mensaje... oculto a vista de todos.

Ejemplo:

Tengo amigos conscientes de lo oscura que es la noche

Si la doblásemos por los lugares indicados:

Te|ngo |am|ig|o|s |con|scientes de |lo| os|cura| que es la noche

... obtenemos la frase...

Te amo con locura

Declaración que en épocas de nuestros padres o abuelos podría haberlos puesto en

problemas.

Se pide escribir un programa que dada una frase original de longitud N y M posibles frases ocultas, permita decidir para cada una de ellas si está contenida en la frase original, una vez doblada. **No se pide saber el sitio exacto de los dobleces**. Se pueden ignorar los espacios.

```
public boolean[] verificar(String frase, String[] posiblesOcultas);
```

La puntuación de este ejercicio dependerá en parte de la complejidad final del algoritmo utilizado. Deberás intentar ahorrar tanto memoria como tiempo de procesamiento.

Ejercicio 2: Prolog

Se quiere saber la nota final de los alumnos de un curso al final del cuatrimestre.
Para ello se pide realizar una regla en prolog llamada ***notas(Alumno, Nota)*** con el fin de obtener las notas.
La forma de calcularlo es:
- Si rindió final, la nota del final
- Si no rindió final y no rindió recuperatorio: el promedio de los dos parciales
- Si no rindió final y rindió recuperatorio: el promedio del parcial con nota más alta y el recuperatorio

Ejemplo 1:
```
parcial1(alumno1, 5).
parcial2(alumno1, 10).
notas(alumno1, N).
% N -> 7.5
```

Ejemplo 2:
```
parcial1(alumno2, 4).
parcial2(alumno2, 6).
recuperatorio(alumno2, 1).
notas(alumno2, N).
% N -> 3.5
% (6 + 1) / 2
```

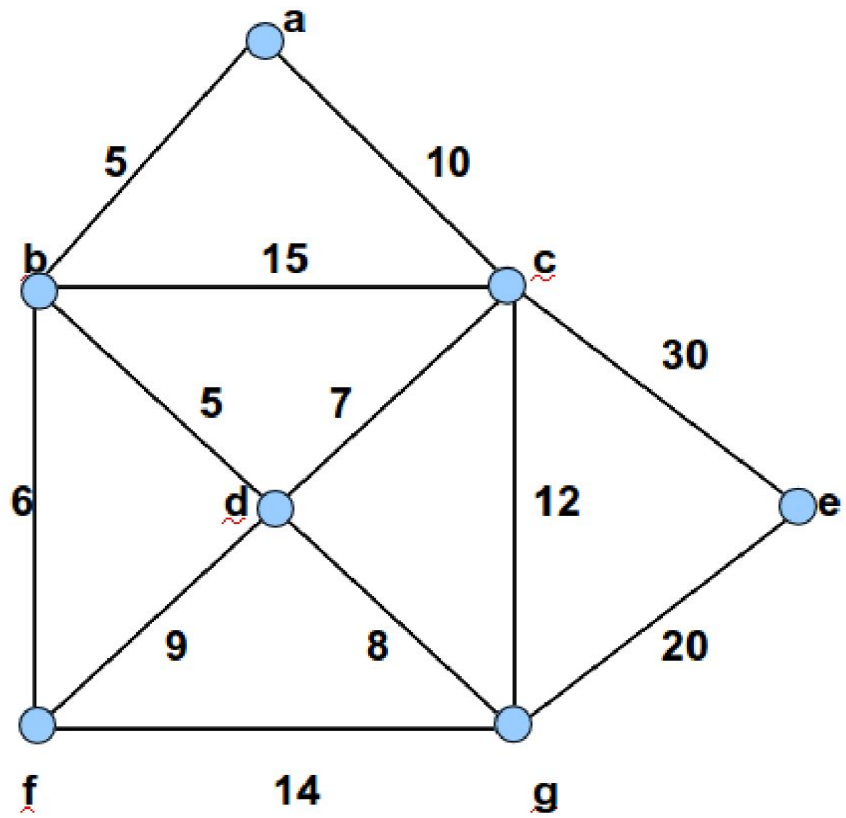
Ejemplo 3:
```
parcial1(alumno3, 4).
parcial2(alumno3, 7).
final(alumno3, 5).
notas(alumno3, N).
% N -> 5
```

Ejemplo 4:
```
parcial1(alumno4, 2).
parcial2(alumno4, 4).
notas(alumno4, N).
% N -> 3
```

Observaciones:
- La regla "notas(ALUMNO, NOTA)" debe devolver la lista entera de alumnos
- El orden de la salida no es relevante
- Los hechos de entrada son siempre válidos, es decir, solo se declaran hechos de recuperatorio y final cuando sea posible. No hay que verificar entrada
- Todos los alumnos tienen nota de parcial1 y parcial2 como mínimo
- Se recomienda usar la cantidad de reglas auxiliares extras que sean necesarias
- **Importante**: No debe haber datos duplicados

Ejercicio 3: Grafos

En el grafo de la figura los vértices representan ciudades y los números sobre las aristas representan los costos de construcción de los caminos indicados.
Determine gráficamente el sistema de carreteras más barato que una todas las ciudades.
Indique claramente qué algoritmo utilizará en la solución.



Figura

Ejercicio 4

Dado el siguiente Array:
[47, 3, 56, 32, 92, 21]
Luego de las primeras dos iteraciones completas de un algoritmo de ordenamiento, el Array ha quedado dispuesto así:
[3, 21, 56, 32, 92, 47]
¿Qué algoritmo de ordenamiento se está utilizando (selección, burbujeo o inserción)? Justifique.