Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2115 – Programación como Herramienta para la Ingeniería (I/2021)

Taller 2a

Objetivos

- Utilizar estructuras de datos avanzadas.

Entrega

- **Lenguaje a utilizar:** Python 3.6 o superior
- **Lugar:** repositorio privado en GitHub. Recuerde incluir todo en una carpeta de nombre **T2a**.
- **Entrega: lunes 12 de abril a las 16:50 hrs.**
- **Formato de entrega:** archivos python notebook (**T2a.ipynb**) y python simple (**T2a.py**) con la solución a lo solicitado en el enunciado. Ambos archivos deben estar ubicados en la carpeta **T2a**. No se debe subir ningún otro archivo a la carpeta. En el archivo python notebook, utilice múltiples celdas de texto y código para facilitar la revisión de su programa. El archivo python simple debe ser exportado a partir del python notebook.
- **NO SE ADMITEN ENTREGAS POR CORREO U OTROS MEDIOS NI ENTREGAS FUERA DE PLAZO**
- **Entregas con errores de sintaxis y/o que generen excepciones serán calificadas con nota 1.0.**

Introducción

Con el fin de continuar ejercitando el uso de estructuras de datos, en este taller deberá desarrollar 2 ejercicios. El primero será guiado a través de instrucciones en el enunciado, mientras que en el segundo el desarrollo será por su cuenta.

Ejercicio 1: longitud máxima de secuencia de paréntesis balanceados

Utilizando la estructura de datos stack, construya un programa que encuentre la longitud de la secuencia de paréntesis balanceados más larga presente en un string compuesto únicamente por paréntesis. La idea central del algoritmo iterar sobre los caracteres del string mientras se actualiza el stack, siempre manteniendo en el tope la posición en el string del carácter inmediatamente anterior al último carácter de apertura de paréntesis detectado. De esta manera, al calcular la diferencia entre la el índice actual de la iteración y el tope del stack, es posible llevar eficientemente registro de la longitud de la secuencia.
Para implementar el algoritmo, siga las instrucciones descritas a continuación:

1. Cree una variable que almacenará el largo de la secuencia balanceada más extensa hasta el momento. Inicialice esta variable en 0.

2. Cree un stack e inserte inmediatemente el valor -1 (esto permite manejar el caso donde el primer abre paréntesis de la secuencia balanceada más larga se encuentra en el índice 0 del string.

3. Itere sobre cada uno de los elementos del string, de izquierda a derecha.

4. Por cada abre paréntesis: i) inserte en el stack el índice en el que se encuentra actualmente la iteración.

5. Por cada cierre paréntesis: i) quite el elemento del tope del stack (si el stack queda vacío después de esto, inserte el índice en el que se encuentra actualmente la iteración), ii) calcule la longitud de la secuencia balanceada actual, restándole al índice actual de la iteración, el valor en el tope del stack, iii) compare la longitud de la secuencia balanceada actual con el máximo registrado hasta el momento y actualice la variable la variable en caso de ser mayor la nueva longitud.

6. Empaque todo esto una función que reciba como argumento el string de paréntesis y que retorne un número natural que indique el resultado deseado. Un ejemplo de ejecución del algoritmo es el siguiente:
   Código

```
parentesis = "((()()"
max_long = calcular_max_long(parentesis)
print(max_long)
```

   Salida

```
4
```

Ejercicio 2: caminos de largo m

Dado un grafo dirigido, un nodo de origen o y un nodo de destino d, encuentre la cantidad de caminos de largo m que van de o hasta d. Se recomienda basar la solución en el algoritmo BFS. Empaque el algoritmo en una función que reciba como argumento una lista de tuplas (donde cada una denota un arco dirigido en el grafo), los índices de los nodos de origen y destino, y el largo m, y retorne un número natural que indique la cantidad de caminos encontrado. Un ejemplo de ejecución del algoritmo es el siguiente:

Código

```
grafo = [(0, 6), (0, 1), (1, 6), (1, 5), (1, 2), (2, 3), (3, 4), (5, 2), (5, 3),
        (5, 4), (6, 5), (7, 6), (7, 1)]
origen = 0
destino = 3
m = 4
num_caminos = caminos_largo_m(grafo, origen, destino, m)
print(num_caminos)
# en este caso los caminos serán:
# 0 > 1 > 5 > 2 > 3
# 0 > 1 > 6 > 5 > 3
# 0 > 6 > 5 > 2 > 3
```

Salida

```
3
```

Objetivo parcial de participación

Para verificar la participación durante la clase, terminar el ejercicio 1.

Política de Integridad Académica

Los alumnos de la Escuela de Ingeniería deben mantener un comportamiento acorde al Código de Honor de la Universidad:

"Como miembro de la comunidad de la Pontificia Universidad Católica de Chile me comprometo a respetar los principios y normativas que la rigen. Asimismo, prometo actuar con rectitud y honestidad en las relaciones con los demás integrantes de la comunidad y en la realización de todo trabajo, particularmente

en aquellas actividades vinculadas a la docencia, el aprendizaje y la creación, difusión y transferencia del conocimiento. Además, velaré por la integridad de las personas y cuidaré los bienes de la Universidad."

En particular, se espera que mantengan altos estándares de honestidad académica. Cualquier acto deshonesto o fraude académico está prohibido; los alumnos que incurran en este tipo de acciones se exponen a un procedimiento sumario. Ejemplos de actos deshonestos son la copia, el uso de material o equipos no permitidos en las evaluaciones, el plagio, o la falsificación de identidad, entre otros. Específicamente, para los cursos del Departamento de Ciencia de la Computación, rige obligatoriamente la siguiente política de integridad académica en relación a copia y plagio: Todo trabajo presentado por un alumno (grupo) para los efectos de la evaluación de un curso debe ser hecho individualmente por el alumno (grupo), sin apoyo en material de terceros. Si un alumno (grupo) copia un trabajo, se le calificará con nota 1.0 en dicha evaluación y dependiendo de la gravedad de sus acciones podrá tener un 1.0 en todo ese ítem de evaluaciones o un 1.1 en el curso. Además, los antecedentes serán enviados a la Dirección de Docencia de la Escuela de Ingeniería para evaluar posteriores sanciones en conjunto con la Universidad, las que pueden incluir un procedimiento sumario. Por "copia" o "plagio" se entiende incluir en el trabajo presentado como propio, partes desarrolladas por otra persona. Está permitido usar material disponible públicamente, por ejemplo, libros o contenidos tomados de Internet, siempre y cuando se incluya la cita correspondiente.