|  |  |
| --- | --- |
| **Estructura de Datos y Algoritmos**  **Grado Ingeniería Informática**  **Universidad Carlos III de Madrid**  **CURSO 2021-2022** | **Logotipo de la Universidad Carlos III de Madrid** |

**Imagen que contiene dibujo, señal, plato

Descripción generada automáticamente**

**Examen Final. 20 Mayo 2022.**

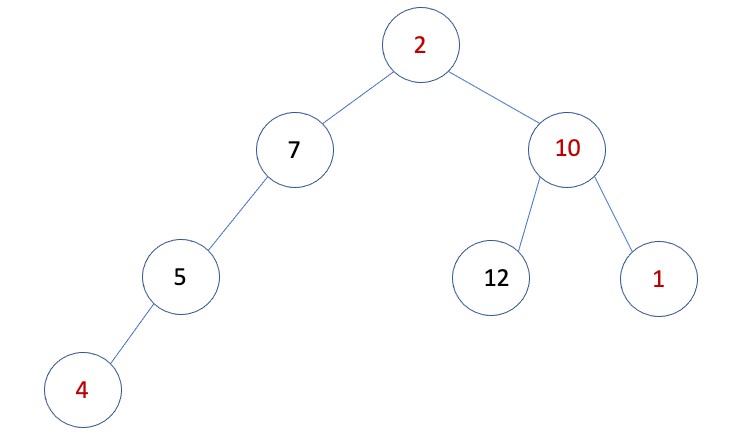
**Nombre:**

Instrucciones:

* Duración del examen: 60’.
* Descarga el zip de Aula Global y descomprimelo.
* En el fichero con el esqueleto (.py) deberás escribir tu solución. Además, añade un comentario python al principio del fichero con tu nombre y apellidos.
* NO MODIFIQUES el fichero con el unittest (te ayuda a testear tu solución).
* Durante el examen, en tu equipo únicamente podrás tener abierto el entorno de desarrollo de Python (por ejemplo, spyder o pycharm). No puedes consultar ningún material (código, apuntes, libros, etc.).
* Cuando queden 5 minutos el profesor te avisará para que subas tu solución. Podrás abrir el navegador y entrar en tu grupo reducido de aula global. Sube únicamente el fichero con el código a Aula Global.
* Es tu responsabilidad comprobar que has subido la versión correcta del fichero.
* Los móviles deben estar apagados y nunca pueden estar encima de la mesa.
* No puedes salir del aula antes de que pasen 30 minutos.
* No está permitido ir al baño.

**Enunciado:**

Te hemos proporcionado una implementación básica de una clase BinaryTree donde almacenaremos números enteros no negativos. Se debe implementar un algoritmo, right\_sum(), que **calcule la suma de todos los nodos que se encuentran en la posición más a la derecha de cada nivel del árbol**. Por ejemplo, dado el siguiente árbol (que contiene 4 niveles) la función debería dar 17, ya que los nodos 2, 10, 1 y 4 están colocados en la posición más a la derecha de cada nivel:



Si el árbol está vacío, el método debe devolver 0.

NO está permitido utilizar listas de Python, colas o cualquier otra estructura de datos. El código propuesto debe resolver el problema, ser capaz de manejar entradas inesperadas y ser eficiente en términos de complejidad de tiempo y espacio. El código debe ser legible y fácil de seguir.