**GRUPO VERYTEL**

Prueba Desarrollador De Software Estándar

**Nombre: Andres Mauricio Garcia Mutis**

**Fecha: 17/08/2023**

1. ¿Qué es el patrón de diseño "singleton" en programación orientada a objetos?

Es un Patrón Creacional GOF, es una técnica fundamental en programación que garantiza que una clase tenga solo una instancia y proporciona un acceso global a esa instancia. Es especialmente útil cuando necesitamos una única instancia de una clase en todo el sistema, como para manejar configuraciones globales, conexiones a bases de datos o recursos compartidos. Al utilizar este patrón, podemos controlar y limitar la creación de instancias, evitando así problemas de duplicación de datos y permitiendo una gestión eficiente de recursos. Esto se logra a través de un constructor privado y un método estático que devuelve la única instancia posible, asegurando que no haya más de una copia en el programa y facilitando la reutilización del objeto creado

¿Lo haz aplicado alguna vez?

Lo use en un proyecto universitario en el cual consistía en conectar varias maquinas las cuales eran consumidores de una instancia principal, y aunque las consumidoras tenían múltiples instancias ellas tomaban información de un Log que se abría por la instancia principal del productor, y para que del Log no se tomaran dos datos iguales al ser asíncrono, tocaba dejarlo como un Singleton privado.

1. En el IDE de desarrollo y lenguaje de programación de su preferencia realice el siguiente algoritmo.

Chicos gangster

A todos nos encantan los emojis. Los emojis ASCII existen desde hace un tiempo, incluido este tipo: (-\_-). Usted puede representar una multitud de estos como este: (- (-\_ (-\_- (-\_ (-\_ (-\_-) \_-) \_-) -\_-) \_-) -).

¡Da miedo! Ese ¡La mafia tiene 11 chicos! Las reglas para crear una mafia son las siguientes:

* 1. Amob puede tener entre 1 y 255 chicos, 255 incluidos.
  2. Si hay un incluso número de chicos, entonces debería haber uno más a la izquierda que a la Derecha: por ejemplo, 4: (-\_ (-\_ (-\_-) \_-)
  3. Hay cuatro tipos diferentes de chicos: un chico completo (-\_-), un chico lateral

\_-), un chico parcial -\_-) y un chico final -).

* + 1. El tipo completo debe estar en el medio, y solo debe haber uno por turba.
    2. Un último tipo debe estar en ambos lados, siempre que haya estrictamente más de 7 tipo.
    3. Digamos el chico completo está en el k **th** posición. Cada ( k ± 3n) **th** chico debe ser un chico parcial.
    4. Todos los demás son chicos secundarios. Un ejemplo, una mafia que tiene 14 hombres es esta. (- (-\_- (-\_ (-\_ (-\_- (-\_ (-\_ (-\_-) \_-) \_-) -\_-) \_-) \_-) -) Cree una función que reciba una entero, indica cuántos tipos hay en tu mafia y devuelve una cadena (string) con la mafia resultante.

Si el parámetro no cumple con la primera regla, o si es nulo o está vacío, la función debería devolver ( O\_o)

**Observar el link de GITHUB en el documento llamado: Verytel Gangster**

1. En el IDE de desarrollo y lenguaje de programación de su preferencia realice el siguiente algoritmo.

Un grupo super secreto comparte a cada uno de sus integrantes una identificación con un número aleatorio. Para evitar la entrada de impostores o espías tienen un sistema de seguridad simple pero funcional.

Todas las identificaciones otorgadas en un día específico son pares o impares, de esta forma se puede reconocer a un espía solo si éste es el único con una identificación que no coincide con el resto.

Con este conocimiento debes crear un programa que ayude al grupo a automatizar su proceso de seguridad.

Para ello se te suministrará de un array de enteros, el cual va a representar una lista con la identificación de los miembros que desean entrar. Tu trabajó será imprimir la identificación del espía.

Nota: Debe asumir que la lista siempre contendrá más de 3 elementos y siempre tendrá un espía.

Ejemplo:

lista\_ids = [10, 8, 66, 124, 12, 11, 2602, 36]

Respuesta:

11, ya que es el único número que es impar.

**Observar el link de GITHUB en el documento llamado: Verytel Espía.**

1. En el IDE de desarrollo y lenguaje de programación de su preferencia realice el siguiente algoritmo.

Tenemos una tienda de antigüedades con productos únicos de un gran valor monetario.

Un día se acerca una persona muy particular al mostrador y nos dice.

"Tengo un show de televisión y me gustaría comprar algunas cosas de valor al azar, por favor organice un poco la tienda para que puedan entrar los equipos de grabación en lo que regrese."

Nosotros, encantados de vender más productos, decidimos acomodar la tienda y sacar nuestros productos más extraños y costosos.

Entre ellos conseguimos una pieza de antigüedad del siglo XV cuyo valor nos parece un poco más bajo del que debería ser.

No tenemos tiempo de imprimir una nueva etiqueta, así que tenemos que jugar con los números para dar el máximo valor posible con los dígitos que ya se tienen.

Entonces, nuestro trabajo será, dado un número entero, hallar el número más grande que se puede generar cambiando la posición de los dígitos.

Ejemplo:

valor = 120

Respuesta:

210.

**Observar el link de GITHUB en el documento llamado: Maximizar.js y Maximizar.py**

1. En el IDE de desarrollo y lenguaje de programación de su preferencia realice el siguiente algoritmo.

Una rana necesita cruzar un río para poder salvarse de los depredadores. Por suerte para ella, dentro del río se pueden observar un par de rocas sobre las cuales se puede aferrar en cada salto.

Al ser una rana adulta y saludable, puede saltar hasta 3 piedras en un mismo salto.

Con esto en mente, nuestro trabajo es averiguar: ¿De cuantas formas es posible que la rana pueda cruzar el río?

Para ello se le suministrará la cantidad de rocas que se pueden observar en el río.

Por suerte para nosotros, podemos asumir que siempre habrá al menos una roca en el río y que la rana siempre se podrá salvar.

Ejemplo:

Si consideramos que n es el número de rocas en el río y tenemos que n=3

Entonces los posibles comportamientos de la rana pueden ser:

1 1 1

1 2

2 1

3

Es decir, solo existen 4 formas en las que la rana pueda salvarse.

**Observar el link de GITHUB en el documento llamado: Ranas.cs y Ranas.py**

1. SQL.

Obtener utilizando una consulta SQL el conteo de personas por departamento y la suma de los sueldos por departamento. Mostrar en el listado el nombre del departamento, la cuenta y la suma, ordenado de manera ascendente por la suma de los sueldos.

Tablas:

APPX\_employee (id, firstname, lastname, department\_id, salary, educationlevel\_id)

APPX\_department (id, department\_name, department\_city)

APPX\_educationlevel (id, description)

**Respuesta:**

SELECT d.department\_name AS department\_name, COUNT(e.id) AS count\_people, SUM(e.salary) AS sum\_salaryFROM APPX\_department dJOIN APPX\_employee e ON d.id = e.department\_idGROUP BY d.department\_nameORDER BY sum\_salary ASC;