3. Implementar el método `buscarMensajeMasVotadoDelUsuario` de la clase `Moderador` a partir de las siguientes especificaciones:

```
class Moderador {
  public:
    /* post: busca en la lista 'foros' el Mensaje más votado del autor 'usuarioBuscado' dentro de un Foro que incluya la
     *       temática 'tematicaBuscada'.
     */
    Mensaje* buscarMensajeMasVotadoDelUsuarioSegunTematica(Lista<Foro*>* foros,
                                                           string usuarioBuscado, string tematicaBuscada);
};
```

```
class Foro {

  public:

    /* post: inicializa el Foro sin Mensajes asignados.
     */
    Foro(string nombre);

    /* post: elimina todos los Mensajes del Foro
     */
    ~Foro();

    /* post: devuelve el nombre del Foro.
     */
    string obtenerNombre();

    /* post: devuelve todos los Mensajes del Foro.
     */
    Lista<Mensaje*>* obtenerMensajes();

    /* post: devuelve las temáticas del Foro
     */
    Lista<string>* obtenerTematicas();

};
```

```
class Mensaje {
  public:
    /* post: 'usuario' es el autor del mensaje con 'texto'
     *       como contenido, sin votos asociados.
     */
    Mensaje(string usuario, string texto);

    /* post: devuelve el nombre del usuario autor del Mensaje.
     */
    string obtenerAutor();

    /* post: devuelve el contenido del Mensaje.
     */
    string obtenerContenido();

    /* post: suma un voto al Mensaje.
     */
    void votar();

    /* post: devuelve la cantidad del Ingrediente.
     */
    unsigned contarVotos();
};
```

4. Diseñar la especificación e implementar el TDA **Restaurante**. Debe proveer operaciones para:

- Construir el Restaurante a partir de la cantidad de mesas que tiene. Cada mesa se identifica con un número.
- adicionarEnMesa: agrega el monto dado [$] a la mesa indicada, dejándola abierta.
- cerrarMesa: devuelve el total de la mesa indicada y la deja libre.
- contarMesasLibres: cuenta la cantidad de mesas que están libres.
- totalizarRecaudacion: devuelve el total

recaudado por el Restaurante.