NO SE CÓMO ATACAR AL PROBLEMA

Si tienes problemas para resolver un problema, intenta preguntarte lo siguiente:

* Cómo se ve la solución esperada?
  + Qué características debe tener?
    - Esas características se extienden al input en general?
  + Que relaciones entre las variables de entrada genera?
    - Se cumple alguna desigualdad, igualdad o propiedad?
* Se le puede aplicar búsqueda binaria?
  + Tenemos algo que se asemeja a una función creciente o decreciente?
* Puedo generar la respuesta con fuerza bruta?
  + Puedo generar la respuesta para casos de ejemplo con fuerza bruta?
    - En ese caso, crea el programa, imprime las respuestas esperadas y busca algún patrón.
* Qué información extra se deduce del statement del problema?
  + Cuáles son los límites de las entradas?
    - Puede dar un hint sobre la complejidad esperada.
      * Límites menores a 1000 pueden sugerir soluciones de programación dinámica o de fuerza bruta
  + Los límites están en términos de potencias de 2?
    - Puede que la solución esté relacionada con el análisis bit a bit de números.
* Se parece a un problema tipo mochila (DP)?
  + Puedo poner esto en términos de una función dp(i,j,…,k)?
    - Qué elementos definen a un estado?
    - ¿Qué información es relevante mantener entre transiciones en el contexto del problema?
    - Enumera los casos/ decisiones que se pueden tomar en cada momento
      * A partir de esos casos, arma las transiciones y los valores de dp(i,j,…,k)
* Dio TLE o SIGSEV?
  + Puede ser un problema de espacio/ uso de memoria
    - En 1 segundo, podemos guardar a lo más memoria del orden 1e6
    - Si es dp y dp solamente depende de k estados anteriores, basta con hacer a dp de tamaño k e ir cambiando modulo k
      * Ejemplo, si dp[i][j] depende de dp[i-1][j] para toda j con i<=n y j<=m, no ocupas:

dp(n+1,vector<ll>(m+1);

Basta con:

dp(2,vector<ll>(m+1); e ir cambiando la i por (imod2) y a i-1 por (1-(imod2))

* + Puede ser un problema de complejidad
    - Revisa nuevamente la complejidad del algoritmo y verifica que no exceda el orden 1e8
  + Puede ser un problema de implementación
    - Analisa ciclos y verifica que las condiciones de iteración eventualmente terminen
      * Si tienes ciclos while, busca errores de concepto o alrededor de la condición que mantiene vivo al while
      * Si tienes ciclos for, busca errores en la declaración del for
        + Indices incongruentes

For(int i=0;j<n;i++)

* + - * + Variables no declaradas o leidas

No se hizo cin>>n o n=valor

* + Analiza el tamaño de los arreglos
    - Debe ser congruente con los indices que utilizas para iterar sobre el
  + Verifica que la declaración de variables sea correcta