Mejores amigos (amigos)

Descripción

Zael ama absolutamente las salteñas. Durante sus vacaciones, visita una panadería famosa en La Paz y compra n **tipos diferentes de salteñas** para compartir con sus dos mejores amigos.

Cada salteña es única: puede ser picante, dulce, de pollo, de carne o incluso vegetariana. Si Zael le da la *i-ésima* salteña a su primer amigo, le traerá a_i unidades de felicidad. Si le da la *i-ésima* salteña a su otro amigo, le traerá b_i unidades de felicidad.

Como a Zael le gustan tanto las salteñas, quiere asegurarse de que ambos amigos las disfruten sin sentirse celosos uno del otro. Eso significa que debe encontrar distribuciones de las salteñas que minimicen la **diferencia absoluta** en la felicidad total entre los dos amigos.

Encuentra las k distribuciones que minimicen la diferencia absoluta en la felicidad. Cada salteña debe ser entregada a **exactamente uno** de los amigos.

Implementación

Debes implementar el siguiente procedimiento:

```
std::vector<long long> best_k (
  int N, int K, std::vector<long long> A, std::vector<long long> B
)
```

o el siguiente procedimiento en Python:

```
def best_k(
    N: int, K: int, A: List[int], B: List[int]
) -> List[int]:
```

Parámetros

- N: el número de salteñas
- K: el número de distribuciones a encontrar
- A: un arreglo de longitud N con la felicidad otorgada al primer amigo.
- B: un arreglo de longitud N con la felicidad otorgada al segundo amigo.

Retorna

El procedimiento debe devolver un arreglo de longitud K que corresponda a las diferencias absolutas de felicidad para las K mejores distribuciones de salteñas, ordenadas en orden ascendente.

Ejemplos

Ejemplo 1

Considera la siguiente llamada:

```
best_k (2, 3, [-1, 3], [-2, 10])
```

Existen 4 posibles distribuciones:

1. Dar la primera salteña al primer amigo y la segunda salteña al segundo amigo $|(-1) - (10)| = 11$
2. Dar la primera salteña al segundo amigo y la segunda salteña al primer amigo $|(-2) - (3)| = 5$
3. Dar todas las salteñas al primer amigo $|(-1 + 3)| = 2$
4. Dar todas las salteñas al segundo amigo $|(-2 + 10)| = 8$

Como $k = 3$, el procedimiento debe devolver:

```
[2, 5, 8]
```

Ejemplo 2

Con la llamada:

```
best_k (4, 16, [1, 2, 3, 4], [4, 3, 2, 1])
```

Debe devolver:

```
[0, 0, 0, 0, 0, 0, 5, 5, 5, 5, 5, 5, 5, 5, 10, 10]
```

Grader de ejemplo

El grader lee en el siguiente formato:

- Linea 1: NK
- Linea 2: $A[0], A[1], A[2], ..., A[N-1]$
- Linea 3: $B[0], B[1], B[2], ..., B[N-1]$

Límites

- $(1 \leq n \leq 40)$
- $(1 \leq k \leq 2^{min(20,n)})$
- $(|a_i| \leq 10^{16})$
- $(|b_i| \leq 10^{16})$

Subtareas

- (20 puntos) $n \leq 20$
- (20 puntos) $k = 1$
- (60 puntos) Sin restricciones adicionales.

Entrada

- Solo debes subir el archivo `amigos.cpp` con tu solución.
- La entrada y salida serán manejadas por el archivo `grader.cpp` o `grader.py`.
- **No debes imprimir nada por la salida estándar** ni leer de la entrada estándar en tu archivo `amigos.cpp`.
- No modifiques los archivos `grader.cpp` o `grader.py`, ya que el juez solo tomará en cuenta tu solución en `amigos.cpp`.