NHF3 – n dimenziós rendezés

1. Feladat: *dual_sort* (40 pont)

Készítsd el a *dual_sort-h*-ban a *dual_sort* implementációját a megadott keretrendszerben! Az algoritmus implementálásához felhasználhatod az órai kódokat is illetve a standard library által nyújtott lehetőségeket kivéve az std::sort-ot! A *dual_sort* egy hibrid rendezőalgoritmus. Alapvetően egy gyorsrendezőként indul, de ha a rekurzió túl mélyre jut egy kupacrendezőre vált a worst-case $O(n^2)$ elkerülése érdekében. Tehát a *dual_sort*:

- Ha a gyorsrendező rekurziójának mélysége elér egy határt akkor a *dual_sort* kupacrendezőre vált és a maradék részt kupacrendezéssel rendezi. Ez a határ $2*\log_2(N)$.

Megvalósítandó feladat: *dual_sort.h*-ban implementáld:

- *void dual_sort(std::vector<int> &data, int N)* eljárást
  - bemenete egy int típusú vektor és a vektor hossza
  - a vektort növekvő sorrendbe rendezetten adja vissza

Egyéb segédfüggvényeket lehet használni.

2. Feladat: *n_dim_sort* (60 pont)

Készítsd el az *n_dim_sort.h*-ban a *sort_2D* és *sort_3D* implementációját a *dual_sort* felhasználásával! A *sort_2D* és *sort_3D* algoritmusok az elemeket adott dimenzión az origóhoz viszonyított **euklideszi távolság** függvényében rendezi:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Nézzük a következő példát!

- Legyen a rendezetlen tömb (vektor) a következő:

```
 Unsorted matrix
32 21 51 55 70 40 36 6 97 25 96 24 25 69 49 71 30 14 28 99 3 26 60 97 50 41 5 3 27 34
```

- A *dual_sort*-tal való rendezés után a tömb, ami az 1D rendezésnek felel meg:

```
 1D sorted matrix
3 3 5 6 14 21 24 25 25 26 27 28 30 32 34 36 40 41 49 50 51 55 60 69 70 71 96 97 97 99
```

- A *sort_2D*-vel való rendezés után a tömb, ami a 2D rendezésnek felel meg (bal kép) és a távolságnégyzetek (jobb kép):

```
 2D sorted matrix
 3  3 14 26 36 60  0
 5  6 24 28 41 96  0
21 25 25 32 51 97  0
27 30 34 50 69  0  0
40 49 55 70  0  0  0
71 97 99  0  0  0  0
 0  0  0  0  0  0  0
```

```
 Distance square matrix in 2D
 0  1  4  9 16 25 36
 1  2  5 10 17 26 37
 4  5  8 13 20 29 40
 9 10 13 18 25 34 45
16 17 20 25 32 41 52
25 26 29 34 41 50 61
36 37 40 45 52 61 72
```

  - **Megjegyzés**: a 0 az üres helyet jelenti. Előre le van foglalva egy 2D mátrix és utána van feltöltve elemekkel. Az előre lefoglalt mátrix mérete a megadott rendezetlen tömb

méretének négyzetgyökével arányos. Jelen esetben a rendezetlen tömb mérete 30. A 2D mátrix mérete: sqrt(30) ~ 6 → 7. Nem hiba, ha a mérete 10x10, de ne legyen 20x20-as!

- A távolságnégyzet mátrixban látható, hogy vannak azonos távolságú pontok. Ekkor a sorrendet először a nagyobb x, majd a nagyobb y koordináta határozza meg. Pl.: 25 távolságnégyzet 4-szer fordul elő (lásd jobb ábra). Emiatt (x,y) koordináta esetén (5,0) helyen 60, (4,3) helyen 69, (3,4) helyen 70 és (0,5) helyen 71 érték szerepel.

- A *sort_3D*-vel való rendezés után a tömb, ami a 3D rendezésnek felel meg (bal kép) és a távolságnégyzetek (jobb kép):

```
 3D sorted matrix
z = 0
 3  3 25 69  0
 5 14 28 99  0
26 30 51  0  0
70  0  0  0  0
 0  0  0  0  0


z = 1
 6 21 32  0  0
24 25 41  0  0
34 49 71  0  0
 0  0  0  0  0
 0  0  0  0  0


z = 2
27 36 55  0  0
40 50 96  0  0
60 97  0  0  0
 0  0  0  0  0
 0  0  0  0  0


z = 3
97  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0


z = 4
 0  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0
 0  0  0  0  0
```

```
 Distance square matrix in 3D
z = 0
 0  1  4  9 16
 1  2  5 10 17
 4  5  8 13 20
 9 10 13 18 25
16 17 20 25 32


z = 1
 1  2  5 10 17
 2  3  6 11 18
 5  6  9 14 21
10 11 14 19 26
17 18 21 26 33


z = 2
 4  5  8 13 20
 5  6  9 14 21
 8  9 12 17 24
13 14 17 22 29
20 21 24 29 36


z = 3
 9 10 13 18 25
10 11 14 19 26
13 14 17 22 29
18 19 22 27 34
25 26 29 34 41


z = 4
16 17 20 25 32
17 18 21 26 33
20 21 24 29 36
25 26 29 34 41
32 33 36 41 48
```

- **Megjegyzés**: a 0 az üres helyet jelenti. Előre le van foglalva egy 3D mátrix és utána van feltöltve elemekkel. Az előre lefoglalt mátrix mérete a megadott rendezetlen tömb méretének köbgyökével arányos. Jelen esetben a rendezetlen tömb mérete 30. A 3D mátrix mérete: köbgyök(30) ~ 3 → 5. Nem hiba, ha a mérete 7x7, de ne legyen 20x20-as!

- A távolságnégyzet mátrixban látható, hogy vannak azonos távolságú pontok. Ekkor a sorrendet először a nagyobb x, majd a nagyobb y, végül a nagyobb z koordináta határozza meg. Pl.: 2 távolságnégyzet 3-szor fordul elő (lásd jobb ábra). Emiatt (x,y,z) koordináta esetén (1,1,0) helyen 4, (1,0,1) helyen 21, és a (0,1,1) helyen 24 érték szerepel.

Megvalósítandó feladat*: n_dim_sort.h*-ban implementáld:

- *std::vector<std::vector<int>> sort_2D(std::vector<int> v)* függvény
  - bemenete egy int típusú 2D vektor
  - kimenete a 2D rendezett vektor
- *std::vector<std::vector<std::vector<int>>> sort_3D(std::vector<int> v)* függvény
  - bemenete egy int típusú 3D vektor
  - kimenete a 3D rendezett vektor

Egyéb segédfüggvényeket lehet használni. Nem kell kiíratni a 2D, 3D mátrix elemeit és a távolságnégyzeteket.