- ## Opis problemu:

Permutacja zbioru n-elementowego - to dowolny n-wyrazowy ciąg utworzony ze wszystkich elementów tego zbioru.

Liczbę permutacji zbioru n-elementowego możemy obliczyć ze wzoru:

$$\mathbf{P_n = n!}$$

- ## Opis algorytmu:

Generuje wszystkie (n-1)! permutacji zbioru {2,...,n}.
np. dla n=4

Dla każdej z wygenerowanych permutacji umieszczam liczbę 1 na kolejnych n możliwych pozycjach.

1**234**	1**324**	1**342**	1**432**	1**423**	1**243**
2134	3124	3142	4132	4123	2143
2314	3214	3412	4312	4213	2413
2341	3241	3421	4321	4231	2431

(n-1)! permutacji zbioru {2,…,n}

Wygenerowany w ten sposób ciąg można przedstawić za pomocą grafu, którego wierzchołkami są permutacje zbioru {1,...n}, a krawędzie łączą jedynie te wierzchołki a, b, dla których od permutacji a do permutacji b można przejść przez zamianę **dwóch sąsiednich pozycji ciągu**. Wtedy ciąg wygenerowany przez algorytm wyznacza ścieżkę przechodzącą przez wszystkie wierzchołki grafu i to przez każdy **tylko raz** (przemieszczając się po kolumnach na zmianę z góry na dół i z dołu na górę).

Taką drogę nazywamy **cyklem Hamiltona** w grafie. Znalezienie cyklu lub ścieżki Hamiltona w grafie jest bardzo trudne obliczeniowo (problem komiwojażera). Problem ten zalicza się to tzw. problemów NP zupełnych, co oznacza, że dla dużej liczby wierzchołków jest on praktycznie nierozwiązywalny w sensownym czasie.

Przykład grafu dla n=3

$$\begin{tikzcd}
123 \arrow[d] & 132 \arrow[l] \\
213 \arrow[d] & 312 \arrow[u] \\
231 \arrow[r] & 321 \arrow[u]
\end{tikzcd}$$

Generowanie permutacji zbioru n-elementowego:

- Tworzę 2 tablice dynamiczne o rozmiarze n (tab1 i tab2),
- wypełniam tab1 ciągiem kolejnych liczb naturalnych od 2 do n,
- w tab1 będę generował (n-1)! permutacji zbioru {2,...,n}. poprzez zamianę sąsiednich liczb, 2 z 3 3 z 4 … n-1 z n,
- po każdej zamianie (wygenerowaniu permutacji) sprawdzam czy powstały ciąg nie jest równy ciągowi liczb naturalnych, jest to warunek zakończenia rekurencji, np.:

  234 3**24** **34**2 4**32** **42**3 2**43**

- do każdej z wygenerowanych permutacji umieszczam liczbę 1 na kolejnych n możliwych pozycjach w tab2, np.:
  1234
  2**1**34
  23**1**4
  234**1**
- wygenerowane w tab2 permutacje wyświetlam w konsoli Windows.

- ## Fragment kodu:

Funkcja wywoływana rekurencyjnie:

```
void permutation(int k, int l){
    int p;
    if(k!=n){
        show();
        tab2[k] = tab2[k+1];
        tab2[k+1] = 1;
        permutation(k+1, l);
    }else if(l!=n-1){
        p = tab1[l];
        tab1[l] = tab1[l+1];
        tab1[l+1] = p;
        for(int i=0; i<n; i++){
            tab2[i] = tab1[i];
        }
        if(check());
        else permutation(0, l+1);

    }
    else if(l==n-1){
        permutation(n,1);
    }
}
```

Przykładowe wywołanie programu:

```
Permutacje zbioru n-elementowego
podaj dlugosc zbioru: 3
123
213
231
132
312
321
```

```
Permutacje zbioru n-elementowego
podaj dlugosc zbioru: 4
1234
2134
2314
2341
1324
3124
3214
3241
1342
3142
3412
3421
1432
4132
4312
4321
1423
4123
4213
4231
1243
2143
2413
2431
```

- ## Wnioski:

  Program poprawnie generuje permutacji zbioru n-elementowego i wyświetla je w konsoli Windows. Dodatkowo waliduje dane wprowadzone przez użytkownika i spełnia wszystkie wstępnie założone kryteria. Ciekawym spostrzeżeniem jest, że wyniki można przedstawić w formie grafu Hamiltona.

Źródła:

[1]. Kokosiński Z.: On generation of permutations through decomposition of symmetric groups into cosets, BIT , Vol.30, pp. 583-591 (1990).

[2]. Permutacje - edu.pjwstk.edu.pl/wyklady/mad/scb/mad11/main11_p4.html

[3]. Znajdowanie cyklu lub ścieżki Hamiltona - eduinf.waw.pl/inf/alg/001_search/0136.php