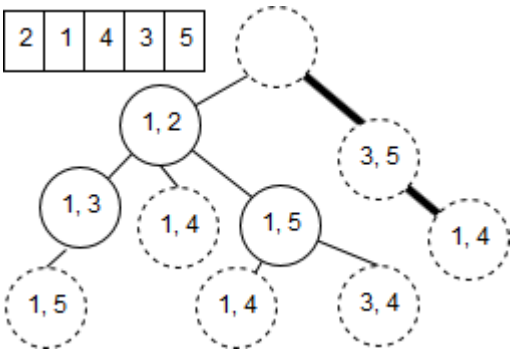
Permutacje-2

Klasa `PermutationSet` służy do przechowywania zbioru permutacji. Przechowuje ciąg elementów `seq` oraz drzewo inwersji `tree`. Permutacja jest przechowywana w postaci ścieżki w drzewie od korzenia do węzła z ustawioną flagą `End`. Pojedynczy węzeł drzewa zawiera parę liczb całkowitych `(i,j)`. W celu uzyskania odpowiadającej mu permutacji należy przestawić zamienić miejscami `i`-ty i `j`-ty wyraz permutacji odpowiadającej jego rodzicowi. Korzeniowi odpowiada permutacja `seq`.

Przykład:



Przerywaną linią zostały zaznaczone węzły z flagą `end` ustawioną na `true`. Zbiór zawiera permutacje {(2, 1, 4, 3, 5), (2, 1, 5, 3, 4), (3, 1, 5, 2, 4), (5, 2, 1, 3, 4), (3, 2, 4, 1, 5), (3, 2, 4, 5, 1), (5, 2, 3, 4, 1)}. Pogrubiona ścieżka odpowiada permutacji (3, 1, 5, 2, 4).

Zadanie polega na **umożliwieniu przeglądania** wszystkich permutacji w zbiorze.

Projekt zawiera:

1. Definicję klasy `PermutationSet`:
   - Pole `root` wskazujące korzeń drzewa
   - Metodę `Add`, która dodaje permutację do zbioru
   - Konstruktor
2. Definicję klasy `PermutationSet.Node` reprezentującą węzeł w drzewie:
   - Właściwość `Children` pozwalającą odwołać się do dzieci węzła
   - Właściwość `Value` określającą wartość w węźle
   - Właściwość End określającą, czy permutacja reprezentowana przez węzeł jest zawarta w zbiorze
3. Definicję struktury `PermutationSet.Transposition` reprezentującej wartość węzła:
   - Pola `i1`, `i2` określają indeksy zamienianych wartości (0, 0 dla korzenia)
4. Przeglądane elementy powinny być generowane na bieżąco w trakcie przeglądania. Nie wolno ich zapisać w pamięci przy jego rozpoczęciu.
5. Plik `Program.cs` zawierający metodę tworzącą przykładowe zbiory permutacji `CreatePermutationSets`

Założenia:

1. Można założyć, że w czasie przeglądania permutacji zawartość zbioru się nie zmieni
2. Powinno być możliwe równoczesne przeglądanie list permutacji różnych zbiorów i o różnej liczbie inwersji przez niezależne komponenty. Nie jest jednak konieczne tworzenie implementacji bezpiecznej dla wątków.
3. Nie wolno używać słowa kluczowego `yield`
4. Nie wolno zmieniać metody `CreatePermutationSets`. W klasie `PermutationSet` można jedynie dopisywać nowe metody, nie wolno zmieniać istniejących.

Twoje zadanie:

1. Dodaj do klasy `PermutationSet` metodę (zaproponuj nazwę i zwracany typ) pozwalającą **przeglądać permutacje** w zbiorze. Permutacja powinna być dostępna w postaci ciągu elementów.
2. W celu zaprezentowania skuteczności i uniwersalności mechanizmu przeglądania dla każdego z przykładowych zbiorów:
   a. Wypisz permutacje zawarte w przykładowym zbiorze
   b. Wypisz permutacje zawarte w przykładowym zbiorze, które na czwartym miejscu zawierają liczbę 7