Implementacja dwukierunkowej listy wiązanej w C++

Autor: *Zachariasz Jażdżewski*

Spełnione wymagania podstawowe

1. Dwukierunkowa lista wiązana

- Spełnione w pliku: `doublylinkedlist.cpp`
- Implementacja znajduje się w przestrzeni nazw `AiSD`, w klasie `DLL<T>` (linia `18`)
- Klasa `DLLNode<T>` (linia `8`) reprezentuje węzeł listy i zawiera:
  - pole `data` przechowujące wartość typu `T`
  - wskaźniki `prev` i `next` do poprzedniego i następnego elementu listy
- Lista przechowuje wskaźniki do `head` (pierwszy element), `tail` (ostatni element) oraz zmienną `size` określającą rozmiar listy

2. Operacje na liście Zaimplementowane w klasie `DLL<T>`:

- `PushFront(const T& el)` – wstawia element na początek listy (linia `31`)
- `PopFront()` – usuwa pierwszy element listy (linia `44`)
- `PushBack(const T& el)` – wstawia element na koniec listy (linia `59`)
- `PopBack()` – usuwa ostatni element listy (linia `72`)
- `Front()` – zwraca referencję do pierwszego elementu listy (linia `87`)
- `Back()` – zwraca referencję do ostatniego elementu listy (linia `91`)
- `IsEmpty()` – zwraca true, jeśli lista jest pusta (linia `95`)
- `Size()` – zwraca rozmiar listy (linia `99`)
- `Clear()` – usuwa wszystkie elementy listy, czyszcząc ją całkowicie (linia `103`)
- `DisplayFromFront()` – wypisuje zawartość listy od początku do końca (linia `109`)
- `DisplayFromBack()` – wypisuje zawartość listy od końca do początku (linia `118`)
- `SaveToFile(const string& filename)` – zapisuje zawartość listy do pliku (linia `127`)

Testowanie

Kod testujący znajduje się w funkcji `main()` (linia `141`) w tym samym pliku, co implementacja.

Demonstracja spełnionych wymagań:

- Tworzenie listy `dll` typu `int`
- Dodawanie elementów z przodu: `PushFront(3)`, `PushFront(2)`, `PushFront(1)`
- Wyświetlanie listy od początku po `PushFront`: `DisplayFromFront()`
- Dodawanie elementów z tyłu: `PushBack(4)`, `PushBack(5)`
- Wyświetlanie listy od przodu i od tyłu: `DisplayFromFront()`, `DisplayFromBack()`
- Odczyt elementów z początku i końca: `Front()`, `Back()`
- Usuwanie elementów: `PopFront()`, `PopBack()`
- Sprawdzenie, czy lista jest pusta: `IsEmpty()`
- Odczyt rozmiaru listy: `Size()`
- Zapis do pliku: `SaveToFile("output.txt")`
- Czyszczenie listy: `Clear()` oraz ponowne sprawdzenie `IsEmpty()`

Sposób uruchomienia

1. **Kompilacja**

   ```
   g++ doublylinkedlist.cpp -o doublylinkedlist
   ```

2. **Uruchomienie**

   ```
   ./doublylinkedlist
   ```

Wynik działania pojawi się w terminalu oraz zostanie zapisany w pliku `output.txt`.