13 listopada 2021

Assignment 3

Napisz (i przetestuj) opisany niżej program:

Struktura opisująca węzeł listy ma postać

```
struct Node {
    int   data;
    Node* next;
};
```

Każdy węzeł przechowuje dane — w tym przypadku po prostu liczbę całkowitą data. Napisz następujące funkcje:

```
1    Node* arrayToList(const int arr[], size_t size);
2    Node* removeOdd(Node* head);
3    void showList(const Node* head);
4    void deleteList(Node*& head);
```

gdzie

1. **arrayToList** pobiera tablicę **int**'ów i jej wymiar. Zadaniem funkcji jest utworzenie listy jednokierunkowej obiektów struktury **Node**, zawierającej w kolejnych węzłach kolejne liczby z przekazanej tablicy (w takiej samej kolejności!). Funkcja zwraca wskaźnik do „głowy" utworzonej listy.
2. **removeOdd** pobiera wskaźnik do „głowy" listy i zwraca wskaźnik do „głowy" listy powstającej z listy pierwotnej po usunięciu wszystkich węzłów, w których data jest liczbą nieparzystą.
   UWAGA: Funkcja ta *nie* powinna tworzyć żadnych nowych węzłów, tylko usuwać te zawierające nieparzyste dane. Jeśli lista zawiera same liczby nieparzyste, wszystkie węzły powinny zostać usunięte, a funkcja powinna zwrócić **nullptr**. Zapewnić, by przy każdym usuwaniu węzła funkcja drukowała wartość danej w nim zawartej, abyśmy widzieli, że rzeczywiście węzły te są usuwane.
3. **showList** drukuje zawartość listy (dane z kolejnych węzłów, w jednej linii, oddzielone znakami odstępu).
4. **deleteList** usuwa wszystkie węzły listy; wskaźnik do „głowy" przesłany jest przez referencję, aby funkcja mogła zmienić jego oryginał (na **nullptr**, co odpowiada liście pustej). Funkcja wypisuje informacje o usuwanych węzłach.

Przykładowy schemat programu:

download *ListyNoTempl.cpp*

```
#include <iostream>

struct Node {
    int   data;
```

```
    Node* next;
};

Node* arrayToList(const int arr[], size_t size) {
        // ...
}

Node* removeOdd(Node* head) {
    // ...
}

void showList(const Node* head) {
    // ...
}

void deleteList(Node*& head) {
    // ...
}

int main() {
    int arr[] = {1,2,3,4,5,6};
    size_t size = sizeof(arr)/sizeof(*arr);
    Node* head = arrayToList(arr,size);
    showList(head);
    head = removeOdd(head);
    showList(head);
    deleteList(head);
    showList(head);
}
```

Program napisany według tego schematu powinien wydrukować:

```
1 2 3 4 5 6
DEL:1 DEL:3 DEL:5
2 4 6
del:2 del:4 del:6
Empty list
```
