27 listopada 2021

Assignment 4

Napisz (i przetestuj) opisany niżej program:
Struktura opisująca węzeł listy ma postać

```
template <typename T>
struct Node {
    T     data;
    Node* next;
};
```

(każdy węzeł przechowuje dane typu **T**).

1. Napisz szablon funkcji

   ```
   template<typename T, typename Pred>
   Node<T>* arrayToList(const T tab[], size_t size);
   ```

   pobierającej tablicę i jej wymiar. Zadaniem funkcji jest utworzenie listy jednokierunkowej obiektów struktury **Node<T>**, zawierającej (jako składowe **data**) w kolejnych węzłach kolejne elementy z przekazanej tablicy (w takiej samej kolejności). Funkcja ma zwracać wskaźnik do „głowy" utworzonej listy.
2. Napisz szablon funkcji

   ```
   template<typename T, typename Pred>
   Node<T>* extract(Node<T>*& head, Pred predicate);
   ```

   pobierającej wskaźnik do „głowy" utworzonej listy. Funkcja ma wydzielić do osobnej listy te węzły, dla których funkcja (predykat) **predicate** zastosowana do pola **data** zwraca **true**. Funkcja **extract** zwraca wskaźnik do głowy (być może pustej) listy zawierającej węzły spełniające predykat, podczas gdy **head** po powrocie z funkcji zawiera wskaźnik do (być może pustej) głowy listy węzłów *niespełniających* predykatu (czyli **head** może być przez funkcję zmodyfikowane — dlatego jest przekazane przez referencję).
   UWAGA: funkcja **extract** operuje wyłącznie na *istniejących* węzłach typu **Node**, w żadnym przypadku nie tworzy nowych obiektów.
   Przy wywołaniu funkcji **extract** drugim argumentem może być zarówno wskaźnik funkcyjny jak i lambda lub obiekt funkcyjny (funktor).
3. Napisz szablon funkcji

   ```
   void deleteList(Node<T>*& head);
   ```

   zwalniającej (za pomocą **delete**) wszystkie węzły listy do której wskaźnik przekazany został jako argument. Przy każdym usuwaniu węzła funkcja powinna drukować wartość danej z usuwanego węzła, abyśmy widzieli, że rzeczywiście węzły te są usuwane. Po powrocie z funkcji **head** powinno być wskaźnikiem pustym (bo reprezentuje listę, która stała się pusta).

4. Napisz funkcję, także w postaci wzorca, która wypisuje elementy listy (w jednej linii, oddzielone spacją).

Przykładowy schemat programu:

download *ListPredTmpl.cpp*

```
#include <iostream>
#include <string>

template <typename T>
struct Node {
        T data;
        Node* next;
};

template <typename T>
void showList(const Node<T>* head);

template <typename T>
Node<T>* arrayToList(const T a[], size_t size);

template <typename T, typename Pred>
Node<T>* extract(Node<T>*& head, Pred pred);

template <typename T>
void deleteList(Node<T>*& head);

bool isLong(const std::string& s) { return s.size() >= 5; }

int main() {
        int tabi[] = { 2, 1, 4, 3, 6, 5, 7, 8 };
        size_t sizei = sizeof(tabi)/sizeof(tabi[0]);
        Node<int>* listAi = arrayToList(tabi, sizei);
        showList(listAi);
        Node<int>* listBi = extract(listAi,
                                    [](int n) {
                                        return n % 2 == 0;
                                    });
        showList(listBi);
        showList(listAi);
        deleteList(listBi);
        deleteList(listAi);

        std::string tabs[]{ "Kasia", "Ola", "Ala",
                            "Zosia", "Ela", "Basia" };
        size_t sizes = sizeof(tabs)/sizeof(tabs[0]);
        Node<std::string>* listAs = arrayToList(tabs, sizes);
        showList(listAs);
```

```
        Node<std::string>* listBs = extract(listAs, isLong);
        showList(listBs);
        showList(listAs);
        deleteList(listBs);
        deleteList(listAs);
}
```

Program powinien wydrukować coś w rodzaju:

```
2 1 4 3 6 5 7 8
2 4 6 8
1 3 5 7
DEL 2; DEL 4; DEL 6; DEL 8;
DEL 1; DEL 3; DEL 5; DEL 7;
Kasia Ola Ala Zosia Ela Basia
Kasia Zosia Basia
Ola Ala Ela
DEL Kasia; DEL Zosia; DEL Basia;
DEL Ola; DEL Ala; DEL Ela;
```
