Corso di Laboratorio di Programmazione

Laboratorio 4 – Array e Liste
15/11/2021

Esercizi

1. Si implementi la funzione

```
bool is_palindrome(const std::string& s);
```

che ritorna `true` se la `std::string s` è palindroma, false altrimenti. Si implementino anche le sue due varianti:

1. `bool is_palindrome(const char s[], int length);` - gli argomenti rappresentano una C-style string e la sua lunghezza (per convenzione la lunghezza si riferisce ai caratteri validi ed esclude il terminatore '\0');

2. `bool is_palindrome(const char* first, const char* last);` - gli argomenti rappresentano il puntatore al primo e all'ultimo carattere di una stringa (riferito all'ultimo carattere valido, non al terminatore '\0').

Implementate in tre file distinti le tre funzioni, ciascuna con un main che crea varie combinazioni di input opportuni e le passa alla funzione corrispondente (devono quindi essere prodotti tre eseguibili diversi). Le stringhe di test possono essere create direttamente nel main – potete usare variabili locali automatiche o allocate dinamicamente.

Suggerimento: se usate un IDE, create tre progetti diversi

2. Considerate la struttura Link vista a lezione, che implementa un nodo di una lista concatenata:

```
struct Link {
      std::string value;
      Link* prev;
      Link* succ;
      Link(const string& v, Link* p = nullptr, Link* s = nullptr)
            : value{v}, prev{p}, succ{s} {}
};
```

Controllate gli appunti per un tempo a piacere – durante questa fase non potete copiare il codice, ma potete **disegnare le mappe della memoria**, da utilizzare durante la fase successiva, dopodiché chiudeteli ed espandete la struttura implementando:

1. Le funzioni `insert`, `add`, `erase`, `find`, `advance`, `print_all` viste a lezione;
2. Le funzioni `push_back` e `push_front` che aggiungono un elemento in coda e in testa alla lista, rispettivamente;
3. Le funzioni `pop_back` e `pop_front` che rimuovono l'ultimo e il primo elemento della lista, rispettivamente (e lo ritornano, senza liberare memoria, al chiamante – il chiamante deve liberare la memoria).

Per ciascuna funzione scegliete una combinazione opportuna di argomenti da usare.

Il punto di partenza per la lista (head) è un puntatore al primo elemento, puntatore che deve essere aggiornato in caso di `push_front` o `pop_front` usando il `return value` di queste funzioni.