

Lista jednokierunkowa

Autor	Daniel Cogieł
Data	15.01.2022

1. Analiza tematu.

Opis zadania

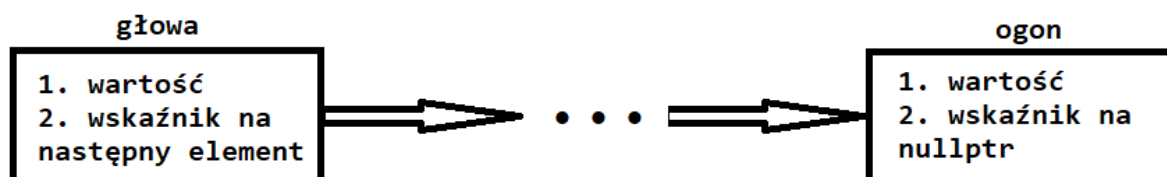
1) Należy zaimplementować listę jednokierunkową. Należy w szablonie klasy zawrzeć:

- inteligentne wskaźniki,
- konstruktory kopiujące/przenoszące, konstruktor bezargumentowy,
- destruktor
- operatory przypisania/przeniesienia,
- dodawanie elementów do kontenera,
- usuwanie wybranego elementu,
- wyszukiwanie elementu,
- sortowanie zawartości (różne kryteria sortowania, wybór czy rosnąco, czy malejąco),
- serializacja i deserializacja (zapis i odczyt z plików binarnych);

2) Należy zaimplementować dodatkowe klasy, na których będzie testowana w/w klasa (szablon). Dla tych dodatkowych klas należy dopisać metodę umożliwiającą wczytywanie z pliku tekstowego danych.

Analiza postawionego zadania

Lista jednokierunkowa będzie się składać z członów zawierających wartość danego typu oraz wskaźnik na element następny. Zostało to przedstawione na poniższym rysunku.



Do realizacji takiej struktury zostaną stworzone dwie klasy (szablony): klasa `List_element`, która reprezentuje jeden element listy oraz klasa `My_forward_list` reprezentująca całą strukturę i zawierająca dwie wartości: wskaźnik na głowę oraz wskaźnik na ogon. Umieszczenie wskaźników właśnie na te dwa elementy: początkowy i końcowy, ułatwi zaimplementowanie funkcji dodawania i usuwania elementów z początku i końca. W projekcie zostaną użyte inteligentne wskaźniki `shared_ptr`, z powodu swojej prostoty w użytkowaniu i braku potrzeby dealokowania pamięci.

2. Specyfikacja zewnętrzna programu.

Instrukcja działania

Program jest wywoływany w konsoli. Aby wczytywać do niego pliki wejściowe i wyjściowe oraz dokonać wyboru, dla jakiego typu bądź klasy chcemy testować program, zostały dodane odpowiednie przełączniki:

- c [1;4] – wybór, dla jakiego typu chcemy testować listę (1 – `double`, 2 – `string`, 3 – `Czlowiek`, 4 – `Samochod`)
- ti „nazwa_pliku.txt” – podanie nazwy wejściowego pliku tekstowego
- to „nazwa_pliku.txt” – podanie nazwy wyjściowego pliku tekstowego
- bi „nazwa_pliku.bin” – podanie nazwy wejściowego pliku binarnego
- bo „nazwa_pliku.bin” – podanie nazwy wyjściowego pliku binarnego

Pliki, na których będziemy testować program, należy umieścić w tym samym folderze, co plik wykonywalny `.exe`.

Przykłady działania

Przykład wywołania programu (testowanie dla klasy Samochod) z argumentami: `-c 4 -ti Cars_input.txt -to Output.txt -bi Cars_input.bin -bo Output.bin`

```
Windows PowerShell
PS C:\Users\Daniel\source\repos\Mikroprojekt_forward_list\Debug> .\Mikroprojekt_forward_list.exe
-c 4 -ti Cars_input.txt -to Output.txt -bi Cars_input.bin -bo Output.bin

=====
Testowanie listy jednokierunkowej dla klasy Samochod
=====
Lista wczytana z pliku Cars_input.txt
Fiat 126p W0V7D5EB3K426G2D9 123101
Nissan Micra Q0U7D2EB6K5062279 324897
Fiat Punto C2V1R91B6K2062375 20000
Hyundai i30 T7C7D5ED3K42632C9 100000
Volkswagen Polo G4U7R2EC6K5262371 56489
Ford Escort D7V1C9126K2D62325 300000

=====
Dodanie do listy za pomoca push_back
=====
Dodawane obiekty:
Subaru Impreza W0V7D9EB6K4062279 250000
Renault Megane_III L1D7C9WB7K5062289 60000

=====
Fiat 126p W0V7D5EB3K426G2D9 123101
Nissan Micra Q0U7D2EB6K5062279 324897
Fiat Punto C2V1R91B6K2062375 20000
```

Przykład wywołania programu (testowanie dla typu double) z argumentami: `-c 1 -ti Doubles_input.txt -to Output.txt -bi Doubles_input.bin -bo Output.bin`

```
Windows PowerShell
PS C:\Users\Daniel\source\repos\Mikroprojekt_forward_list\Debug> .\Mikroprojekt_forward_list.exe
-c 1 -ti Doubles_input.txt -to Output.txt -bi Doubles_input.bin -bo Output.bin

=====
Testowanie listy jednokierunkowej dla typu double
=====
Lista wczytana z pliku Doubles_input.txt
12.2 13.4 17.5 87.5 65.1 32.3 78.5 98.3 32.1 56.32 24.6 -12.3 -23.1 23.1 -23.1 -76.6

=====
Dodanie do listy za pomoca push_back
12.2 13.4 17.5 87.5 65.1 32.3 78.5 98.3 32.1 56.32 24.6 -12.3 -23.1 23.1 -23.1 -76.6 23.7 -27.2

=====
Dodanie do listy za pomoca push_front
-8.2 43.1 12.2 13.4 17.5 87.5 65.1 32.3 78.5 98.3 32.1 56.32 24.6 -12.3 -23.1 23.1 -23.1 -76.6 2
3.7 -27.2

=====
Uzycie pop_back()
-8.2 43.1 12.2 13.4 17.5 87.5 65.1 32.3 78.5 98.3 32.1 56.32 24.6 -12.3 -23.1 23.1 -23.1 -76.6 2
3.7

=====
Uzycie pop_front()
43.1 12.2 13.4 17.5 87.5 65.1 32.3 78.5 98.3 32.1 56.32 24.6 -12.3 -23.1 23.1 -23.1 -76.6 23.7

=====
```

3. Specyfikacja wewnętrzna programu.

Ogólna struktura programu i schemat działania

W programie utworzone zostały w sumie 4 klasy: szablon klasy `List_element` reprezentujący element listy, szablon klasy `My_forward_list` reprezentujący listę, klasy `Czlowiek` i `Samochod` służące do testowania stworzonego kontenera. Dla każdej klasy zostały dodane odpowiednie metody zapewniające wymagane funkcjonalności tworzonego kontenera. Program zawiera również 5 funkcji, które służą do wczytywania odpowiednich elementów podanych jako argumenty programu dzięki przełącznikom. Po wywołaniu programu, jest on testowany dla wybranego typu (`double`, `string`, `Czlowiek` bądź `Samochod`). Prezentacja wszystkich funkcjonalności programu dla wybranego typu została zawarta w funkcji `main()` i wyświetla się po jego wywołaniu.

Szczegółowy opis klas i funkcji

Szczegółowy opis klas i funkcji został zawarty w załączniku „Opis klas i funkcji.pdf”.

4. Testowanie i uruchamianie.

Program został przetestowany dla podstawowych typów oraz klas stworzonych specjalnie dla testowania i działał poprawnie. W przypadku gdy nie zostaną podane żadne argumenty wejściowe, test zawarty w funkcji `main()` po prostu się nie wykona. Niepodanie pliku wejściowego skutkuje tym, że program wykona test na początkowo pustym kontenerze (w czasie testowania program dodaje do listy nowe elementy, a więc lista wyjściowa będzie po prostu o wiele krótsza). Niepodanie pliku wyjściowego skutkuje tym, że program nie zapisze stanu listy do pliku.

Program został także przetestowany pod kątem wycieków pamięci przy pomocy biblioteki `<crtdbg.h>`, a konkretnie instrukcji `_CrtDumpMemoryLeaks()`, zbierającej wszystkie wycieki pamięci.

Program nie generuje żadnych wycieków ze względu na użycie inteligentnych wskaźników `shared_ptr`, co potwierdza załączony zrzut ekranu.

PlikEdycjaWidokProjektKompilowanieDebugowanieTestAnalizaNarzędziaRozszerzeniaOknoPomocWyszukaj (Ctrl+Q)

Debugx86Lokalny debugger Windows

main.cppfunkcje.hfunkcje.cppLista.hCzlowiek.hSamochod.h

Mikroprojekt_forward_list(Globalny zasięg)

```
1 #include "Czlowiek.h"
2 #include "Samochod.h"
3 #include "Lista.h"
4 #include "funkcje.h"
5
6
7
8 int main(int argc, char* argv[])
9 {
10     {
11         string text_input = get_text_input(argc, argv);
12         string bin_input = get_bin_input(argc, argv);
13         string text_output = get_text_output(argc, argv);
14         string bin_output = get_bin_output(argc, argv);
15         int choice = get_choice(argc, argv);
16
17         //Blok DOUBBLE
18         if (choice == 1){ ... }
19
20         //Blok STRING
21         else if (choice == 2){ ... }
22
23         //Blok CZLOWIEK
24         else if (choice == 3){ ... }
25
26         //Blok SAMOCHOD
27         else if (choice == 4){ ... }
28     }
29
30     _CRTDumpMemoryLeaks(); //wykrywanie wycieków pamięci
31 }
```

99 %Nie znaleziono żadnych problemów

Dane wyjściowe

Pokaż dane wyjściowe z: Debugowanie

```
„Mikroprojekt_forward_list.exe” (Win32): załadowano „C:\Users\Daniel\source\repos\Mikroprojekt_forward_list\Debug\Mikroproj...
„Mikroprojekt_forward_list.exe” (Win32): załadowano „C:\Windows\SysWOW64\ntdll.dll”.
„Mikroprojekt_forward_list.exe” (Win32): załadowano „C:\Windows\SysWOW64\kernel32.dll”.
„Mikroprojekt_forward_list.exe” (Win32): załadowano „C:\Windows\SysWOW64\KernelBase.dll”.
„Mikroprojekt_forward_list.exe” (Win32): załadowano „C:\Windows\SysWOW64\msvcrt.dll”.
„Mikroprojekt_forward_list.exe” (Win32): załadowano „C:\Windows\SysWOW64\vcruntime140d.dll”. Symbole zostały załadowane.
„Mikroprojekt_forward_list.exe” (Win32): załadowano „C:\Windows\SysWOW64\ucrtbased.dll”.
Wątek 0x2360 zakończył działanie z kodem 0 (0x0).
„Mikroprojekt_forward_list.exe” (Win32): załadowano „C:\Windows\SysWOW64\kernel.appcore.dll”.
„Mikroprojekt_forward_list.exe” (Win32): załadowano „C:\Windows\SysWOW64\msvcp140d.dll”.
„Mikroprojekt_forward_list.exe” (Win32): załadowano „C:\Windows\SysWOW64\rpcrt4.dll”.
Wątek 0x2f84 zakończył działanie z kodem 0 (0x0).
Wątek 0x4fc0 zakończył działanie z kodem 0 (0x0).
Program „[7644] Mikroprojekt_forward_list.exe” zakończył działanie z kodem 0 (0x0).
```

5. Wnioski.

Lista jednokierunkowa jest jednym z prostszych do utworzenia kontenerów, jednak jej struktura czasami utrudnia implementację niektórych funkcji, takich jak np. `sort()`, z powodu braku możliwości cofania się, takiej, jaka jest zawarta w liście dwukierunkowej.