WYDZIAŁ
INFORMATYKI
I TELEKOMUNIKACJI

Obiegi drzew

Veronika Tronchuk

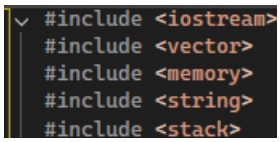
I rok Informatyki

Nr albumu: 30019

Celem

było pokazanie, jak działa obieg drzewa dowolnego rzędu w porządku in-order i pre-order oraz sprawdzenie, jak różnią się wyniki w zależności od sposobu przechodzenia węzłów.

W tym zadaniu pokazałam, jak działa obieg drzewa w dwóch wersjach: in-order i pre-order. In-order odwiedza dzieci, potem węzeł, a pre-order najpierw węzeł, potem dzieci. Struktura drzewa jest taka sama jak w poprzednim przykładzie. Na końcu wypisałam drzewo i wyniki obu obiegów. Wszystko działa poprawnie.



```
#include <iostream>
#include <vector>
#include <memory>
#include <string>
#include <stack>
```

dodajemy potrzebne biblioteki: do wejścia/wyjścia, do wektorów, do wskaźników inteligentnych shared_ptr, do stringów i stosu.

```
using namespace std;
```

używamy przestrzeni nazw std, żeby nie pisać za każdym razem std::.

```
struct Node {
    string data;
    vector<shared_ptr<Node>> children;

    Node(const string& d) : data(d) {}
};
```

struktura Node to jeden węzeł drzewa. Przechowuje tekst (data) i listę dzieci. Konstruktor ustawia data.

```
void addChild(shared_ptr<Node> parent, shared_ptr<Node> child) {
    parent->children.push_back(child);
}
```

funkcja dodaje dziecko do listy dzieci rodzica.

```
void printTree(const shared_ptr<Node>& node, int depth = 0) {
    for (int i = 0; i < depth; ++i) cout << "  ";
    cout << node->data << "\n";
    for (const auto& child : node->children) {
        printTree(child, depth + 1);
    }
}
```

funkcja printTree wypisuje drzewo w formie wizualnej. Dodaje wcięcie (depth), pokazuje węzeł i rekursywnie dzieci.

```
void inorder(const shared_ptr<Node>& node) {
    if (node == NULL)
        return;
    int total = node->children.size();
    for (int i = 0; i < total - 1; i++) {
        inorder(node->children[i]);
    }
    cout << node->data << " ";
    if (total > 0) {
        inorder(node->children[total - 1]);
    }
}
```

funkcja inorder robi obieg in-order: najpierw przechodzi wszystkie dzieci oprócz ostatniego, potem wypisuje bieżący węzeł, a na końcu ostatnie dziecko.

```
vector<string> preorder(const shared_ptr<Node>& root) {
    if (!root) return {};
    vector<string> result;
    stack<shared_ptr<Node>> nodes;
    nodes.push(root);
    while (!nodes.empty()) {
        auto current = nodes.top();
        nodes.pop();
        result.push_back(current->data);
        for (int i = current->children.size() - 1; i >= 0; --i) {
            nodes.push(current->children[i]);
        }
    }
    return result;
}
```

funkcja preorder robi obieg pre-order: zaczyna od korzenia, dodaje do wyniku, potem na stos wrzuca dzieci w odwrotnej kolejności, żeby były w poprawnej kolejności w wyniku.

```
auto director = make_shared<Node>("Director");
auto asstGrad = make_shared<Node>("Asst. Dir. of Graduate Studies");
auto assocAcad = make_shared<Node>("Assoc. Dir. for Academic Programs");
auto adminAsst = make_shared<Node>("Administrative Assistant to the Director");
auto mainOffice = make_shared<Node>("Main Office Assistant");
auto assocStudent = make_shared<Node>("Assoc. Dir. for Student Services Administrations");
auto fellow = make_shared<Node>("Faculty Fellow for Strategic Initiatives");
auto advisoryBoard = make_shared<Node>("School of Music Advisory Board");
```

w main tworzymy węzeł główny Director i inne stanowiska w organizacji.

```
addChild(director, asstGrad);
addChild(director, assocAcad);
addChild(director, adminAsst);
addChild(director, mainOffice);
addChild(director, assocStudent);
addChild(director, fellow);
addChild(director, advisoryBoard);
```

do Director dodajemy dzieci — inne osoby.

```
auto popularMusic = make_shared<Node>("Popular Music");
auto academicStudies = make_shared<Node>("Academic Studies");
auto performanceStudies = make_shared<Node>("Performance Studies");
auto staff = make_shared<Node>("Staff");

addChild(director, popularMusic);
addChild(director, academicStudies);
addChild(director, performanceStudies);
addChild(director, staff);
```

tworzymy kolejne główne sekcje organizacji i dodajemy je do Director.

```
addChild(popularMusic, make_shared<Node>("Audio Engineering"));
addChild(popularMusic, make_shared<Node>("Music Business"));
addChild(popularMusic, make_shared<Node>("Performance & Songwriting"));
addChild(popularMusic, make_shared<Node>("Jazz Studies"));
addChild(popularMusic, make_shared<Node>("Music + Entertainment"));

addChild(academicStudies, make_shared<Node>("Musicology"));
addChild(academicStudies, make_shared<Node>("Music Education"));
addChild(academicStudies, make_shared<Node>("Music Theory/Composition"));
addChild(academicStudies, make_shared<Node>("General Studies in Music"));
addChild(academicStudies, make_shared<Node>("Music Therapy"));

addChild(performanceStudies, make_shared<Node>("Conducting"));
addChild(performanceStudies, make_shared<Node>("Athletic Bands"));
addChild(performanceStudies, make_shared<Node>("Asst. Band Instructor"));
addChild(performanceStudies, make_shared<Node>("Administrative Associate"));
addChild(performanceStudies, make_shared<Node>("Staff Pianist"));

addChild(staff, make_shared<Node>("Business Office Manager"));
addChild(staff, make_shared<Node>("Operations + Events Manager"));
addChild(staff, make_shared<Node>("Marketing Communications Officer"));
addChild(staff, make_shared<Node>("Recruitment + Enrollment Officer"));
addChild(staff, make_shared<Node>("Technical Director"));
addChild(staff, make_shared<Node>("Facilities Manager"));
```

Dodawanie podsekcji Popular Music, Academic Studies, Performance Studies i Staff

```
cout << "struktura\n";
printTree(director);

cout << endl;
cout << "In-order\n";
inorder(director);
cout << endl;
cout << endl;

cout << "pre-order\n";
vector<string> res = preorder(director);
for (const auto& s : res) {
    cout << s << " ";
}
cout << endl;

cin.get();
return 0;
```

wypisujemy całe drzewo (printTree)

wykonujemy obieg in-order i wypisujemy wynik
wykonujemy obieg pre-order i wypisujemy wynik

```
struktura
Director
  Asst. Dir. of Graduate Studies
  Assoc. Dir. for Academic Programs
  Administrative Assistant to the Director
  Main Office Assistant
  Assoc. Dir. for Student Services Administrations
  Faculty Fellow for Strategic Initiatives
  School of Music Advisory Board
  Popular Music
    Audio Engineering
    Music Business
    Performance & Songwriting
    Jazz Studies
    Music + Entertainment
  Academic Studies
    Musicology
    Music Education
    Music Theory/Composition
    General Studies in Music
    Music Therapy
  Performance Studies
    Conducting
    Athletic Bands
    Asst. Band Instructor
    Administrative Associate
    Staff Pianist
  Staff
    Business Office Manager
    Operations + Events Manager
    Marketing Communications Officer
    Recruitment + Enrollment Officer
    Technical Director
    Facilities Manager

In-order
Asst. Dir. of Graduate Studies Assoc. Dir. for Academic Programs Administrative Assistant to the Director Main Office Assistant Assoc. Dir. for Student Services Administrations Faculty Fellow for Strategic In
itiatives School of Music Advisory Board Audio Engineering Music Business Performance & Songwriting Jazz Studies Popular Music Music + Entertainment Musicology Music Education Music Theory/Composition General
 Studies in Music Academic Studies Music Therapy Conducting Athletic Bands Asst. Band Instructor Administrative Associate Performance Studies Staff Pianist Director Business Office Manager Operations + Events
 Manager Marketing Communications Officer Recruitment + Enrollment Officer Technical Director Staff Facilities Manager

pre-order
Director Asst. Dir. of Graduate Studies Assoc. Dir. for Academic Programs Administrative Assistant to the Director Main Office Assistant Assoc. Dir. for Student Services Administrations Faculty Fellow for Str
ategic Initiatives School of Music Advisory Board Popular Music Audio Engineering Music Business Performance & Songwriting Jazz Studies Music + Entertainment Academic Studies Musicology Music Education Music
Theory/Composition General Studies in Music Music Therapy Performance Studies Conducting Athletic Bands Asst. Band Instructor Administrative Associate Staff Pianist Staff Business Office Manager Operations +
Events Manager Marketing Communications Officer Recruitment + Enrollment Officer Technical Director Facilities Manager
```

==========


```
struktura
Director
  Asst. Dir. of Graduate Studies
  Assoc. Dir. for Academic Programs
  Administrative Assistant to the Director
  Main Office Assistant
  Assoc. Dir. for Student Services Administrations
  Faculty Fellow for Strategic Initiatives
  School of Music Advisory Board
  Popular Music
    Audio Engineering
    Music Business
    Performance & Songwriting
    Jazz Studies
    Music + Entertainment
  Academic Studies
    Musicology
    Music Education
    Music Theory/Composition
    General Studies in Music
    Music Therapy
  Performance Studies
    Conducting
    Athletic Bands
    Asst. Band Instructor
    Administrative Associate
    Staff Pianist
  Staff
    Business Office Manager
    Operations + Events Manager
    Marketing Communications Officer
    Recruitment + Enrollment Officer
```

```
In-order
Asst. Dir. of Graduate Studies Assoc. Dir. for Academic Programs Administrative Assistant to the
 Director Main Office Assistant Assoc. Dir. for Student Services Administrations Faculty Fellow
for Strategic Initiatives School of Music Advisory Board Audio Engineering Music Business Perfor
mance & Songwriting Jazz Studies Popular Music Music + Entertainment Musicology Music Education
Music Theory/Composition General Studies in Music Academic Studies Music Therapy Conducting Athl
etic Bands Asst. Band Instructor Administrative Associate Performance Studies Staff Pianist Dire
ctor Business Office Manager Operations + Events Manager Marketing Communications Officer Recrui
tment + Enrollment Officer Technical Director Staff Facilities Manager

pre-order
Director Asst. Dir. of Graduate Studies Assoc. Dir. for Academic Programs Administrative Assista
nt to the Director Main Office Assistant Assoc. Dir. for Student Services Administrations Facult
y Fellow for Strategic Initiatives School of Music Advisory Board Popular Music Audio Engineerin
g Music Business Performance & Songwriting Jazz Studies Music + Entertainment Academic Studies M
usicology Music Education Music Theory/Composition General Studies in Music Music Therapy Perfor
mance Studies Conducting Athletic Bands Asst. Band Instructor Administrative Associate Staff Pia
nist Staff Business Office Manager Operations + Events Manager Marketing Communications Officer
Recruitment + Enrollment Officer Technical Director Facilities Manager
```

Wnioski

Kod działa poprawnie, ale można go jeszcze ulepszyć. Funkcje obiegu nie zapisują wyników do pliku, tylko wypisują wszystko w konsoli, co jest mniej praktyczne przy dużych drzewach. Funkcja preorder używa stosu, chociaż można by zrobić wersję rekurencyjną tak jak inorder. Brakuje też walidacji danych – na przykład, czy węzły nie są puste. Dodatkowo struktura może być bardziej ogólna, jeśli drzewo miałoby bardzo wiele poziomów i dzieci.

Na podstawie obiegów widać, jak zmienia się kolejność odwiedzania węzłów. Program działa zgodnie z teorią i potwierdza różnicę między in-order a pre-order.