**РУСЕНСКИ УНИВЕРСИТЕТ “АНГЕЛ КЪНЧЕВ”**

**КУРСОВА РАБОТА**

по

САА

Студент:

Факултетен номер:

Група:

Специалност:

Дата: ПРОВЕРИЛ:.....................

РУСЕ

2015 г.

# Задача.

Напишете програма, която създава и поддържа двоично дърво на търсене от двойки име на държава/население. Името на държавата е ключ на подредба. Програмата трябва да изпълнява следните функции:

* Създаване на дървото като информацията се чете от клавиатурата.
* [Извеждане на информация за конкретна държава. Ако търсената държава не е открита в дървото да се извежда подходящо съобщение].
* Модифициране (промяна на населението) на възел от дървото.
* Извеждане на информацията за всички държави, подредена по азбучен ред на името.
* [Извеждане на информацията за всички държави, чиито имена са в даден интервал на азбуката, например [Е, М] (да се използва частично обхождане). Информацията да е подредена по азбучен ред имената на държавите. Да се изведе и техният брой.]

# Описание на решението.

Първо: дефинираме структура (Country), в която се включва името (Name) на държавата и нейната популация (Population). Също и два указателя към лявата и дясната част на поддървото (left и right).

Въвеждаме държавите в дървото с функцията create, първо се извършва проверка съществува ли дървото, ако не създаваме възел, след което се установява мястото на въведената държава, ако вече съществува се извежда съобщение, че вече съществува такава държава.

Извеждането на информацията за всички държави се осъществява от функцията print, ако дървото съществува отпечатваме във възходящ ред.

Извеждане информацията за дадена държава се осъществява с функцията search. Функцията се извиква рекурсивно, докато не се намери възелът или NULL.

Промяна броя на населението на зададена държава се осъществява с функцията change. Желаният елемент за модифициране се търси в дървото с функцията search. Ако не е намерена, зададената държава се извежда съответното съобщение.

Извеждане информацията за държави в определен буквен интервал се осъществява от функцията search2.

Извеждаме меню на програмата чрез главната програма, където са дефинирани име на държава (name), население (pop), символ за съгласие (c1), начало и край на интервала за извеждане (ch1 I ch2), символ за избор от менюто с опциите (c)..

# Описание на използваните функции.

(1)

Вид на функцията: int main();

Действие: главна програма (функция);

Параметри: няма;

Използвани глобални променливи: root – указател към началото на дървото (коренът)

Извиквани функции: create(), print(), search(), change(), search2().

(2)

Вид на функцията: create(string name, unsigned long population, Country \*&rootTree);

Действие: функция за създаване на двоично дърво

Параметри: name – името на държавата, population – популация (население), rootTree – указател към корена на дървото;

Използвани глобални променливи: няма

Извиквани функции: create() – рекурсия.

(3)

Вид на функцията: void print(Country \* rootTree);

Действие: функция за печатане на дървото

Параметри: rootTree – указател към корена на дървото;

Използвани глобални променливи: няма

Извиквани функции: print() – рекурсия.

(4)

Вид на функцията: Country \* search(string name, Country \* rootTree);

Действие: функция за търсене на държава

Параметри: name – името на държавата за търсене, rootTree – указател към корена на дървото;

Използвани глобални променливи: няма

Извиквани функции: search() – рекурсия.

(5)

Вид на функцията: void change(string name, Country \* rootTree);

Действие: функция за модифициране на информацията за дадена държава в дървото

Параметри: name – името на държавата за търсене, rootTree – указател към корена на дървото;

Използвани глобални променливи: няма

Извиквани функции: search().

(6)

Вид на функцията: void search2(char start, char end, Country \* rootTree);

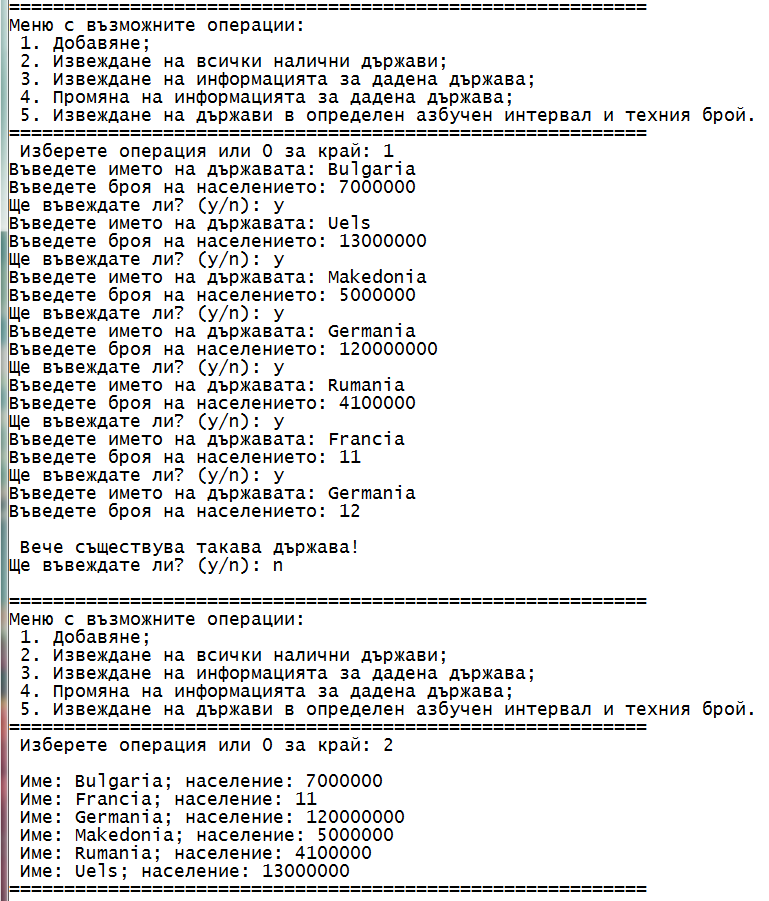
Действие: функция за търсене –> печатане на информацията за държави в даден интервал

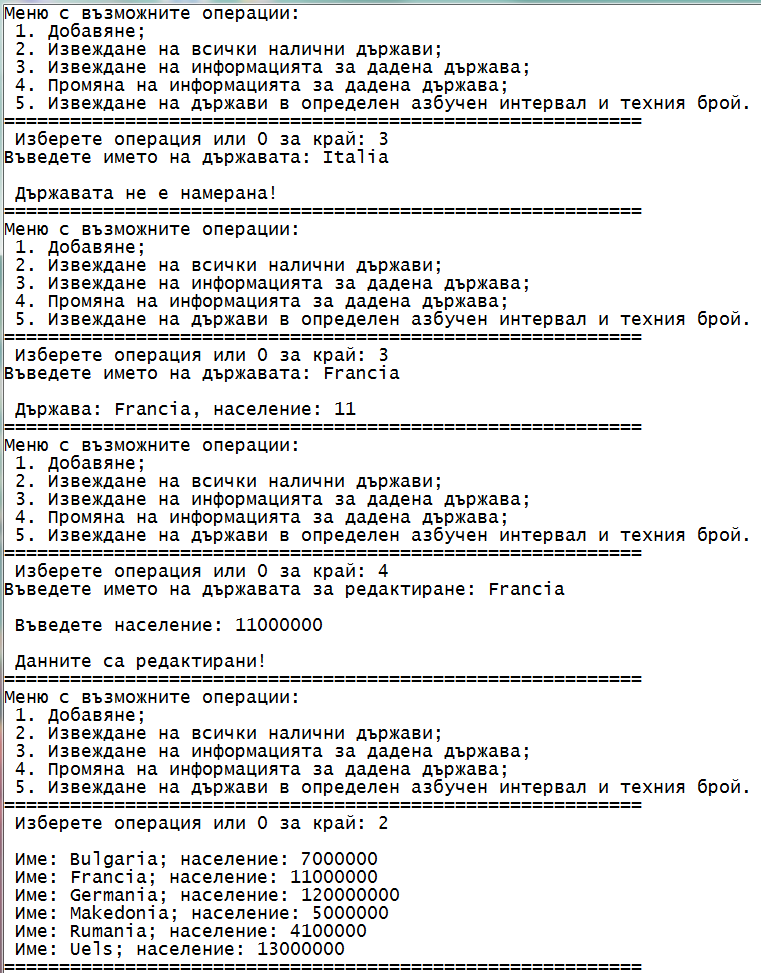
Параметри: start – начало на интервала, end – край на интервала, rootTree – указател към корена на дървото;

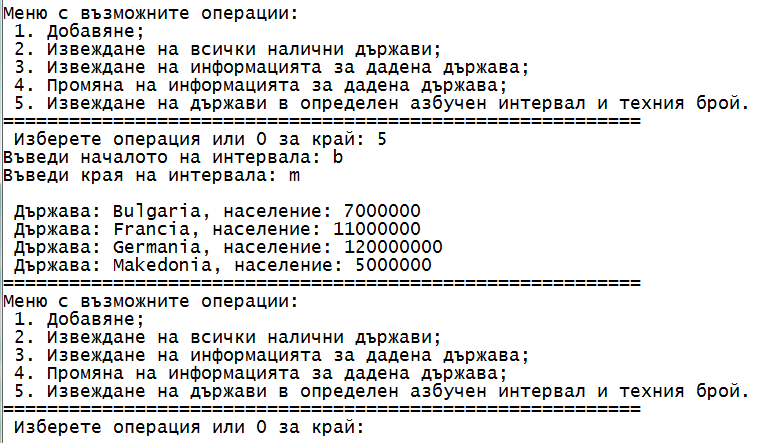
Използвани глобални променливи: няма

Извиквани функции: search2() - рекурсия.

# Тестов пример.







# Код на програмата.

#include <stdio.h>

#include <iostream>

#include <string.h>

#include <stdlib.h>

using namespace std;

//Дефиниране на дървото

struct Country {

string Name; //Име на държавата

unsigned long Population; //Брой население

Country \*left, \*right;

}

\*root = NULL;

//Въвеждане на държави в дървото

void create(string name, unsigned long population, Country \*&rootTree) {

if (rootTree == NULL) { //Ако няма въведени държави

rootTree = new Country; //се създава коренът

rootTree->Name = name;

rootTree->Population = population;

rootTree->left = rootTree->right = NULL;

}

else if (name < rootTree->Name) { //Проверява се дали нововъведената е по-голяма от корена

create(name, population, rootTree->left); //Ако да - се записва отляво

}

else if (name > rootTree->Name) {

create(name, population, rootTree->right); //В противен случай - вдясно

}

else {

cout << "\n Вече съществува такава държава!\n";

return;

}

}

//Извеждане на информацията за всички държави

void print(Country \*rootTree) {

if (rootTree) { //Ако дървото съществува

print(rootTree->left); //отпечатваме във възходящ ред

cout << "\n Име: " << rootTree->Name

<< "; население: " << rootTree->Population;

print(rootTree->right);

}

}

//Търсене на държава

Country \* search(string name, Country \*rootTree) {

if (rootTree == NULL)

return NULL;

else if (name == rootTree->Name) {

return rootTree;

}

else if (name < rootTree->Name) {

return search(name, rootTree->left);

}

else if (name > rootTree->Name) {

return search(name, rootTree->right);

}

}

//Промяна информацията за дадена държава

void change(string name, Country \*rootTree) {

Country \*result = search(name, rootTree);

if (result) {

cout << "\n Въведете население: ";

cin >> result->Population;

cout << "\n Данните са редактирани!";

}

else {

cout << "\n Държавата не е намерена!";

}

}

void search2(char start, char end, Country \*rootTree) {

if (rootTree == NULL) {

return;

}

if (toupper(start) < toupper(rootTree->Name[0])) {

search2(start, end, rootTree->left);

}

if (toupper(start) <= toupper(rootTree->Name[0]) && toupper(end) >= toupper(rootTree->Name[0])) {

cout << "\n Държава: " << rootTree->Name << ", население: " << rootTree->Population;

}

if (toupper(end) > toupper(rootTree->Name[0])) {

search2(start, end, rootTree->right);

}

}

int main() {

//system("chcp 1251"); // кирилица - Visual С++

setlocale(LC\_ALL, "BGR"); // кирилица - DEV C++

char c1, ch1, ch2;

string name;

unsigned long pop;

int c;

do {

cout << "\n==========================================================";

cout << "\nМеню с възможните операции:";

cout << "\n 1. Добавяне;";

cout << "\n 2. Извеждане на всички налични държави;";

cout << "\n 3. Извеждане на информацията за дадена държава;";

cout << "\n 4. Промяна на информацията за дадена държава;";

cout << "\n 5. Извеждане на държави в определен азбучен интервал и техния брой.";

cout << "\n==========================================================";

cout << "\n Изберете операция или 0 за край: ";

cin >> c;

switch (c) {

case 1: {

do {

cout << "Въведете името на държавата: ";

cin >> name;

cout << "Въведете броя на населението: ";

cin >> pop;

create(name, pop, root);

cout << "Ще въвеждате ли? (y/n): ";

cin >> c1;

} while (toupper(c1) != 'N');

break;

}

case 2: print(root); break;

case 3: {

cin.sync();

cout << "Въведете името на държавата: ";

cin >> name;

Country \* result = search(name, root);

if (result != NULL) {

cout << "\n Държава: " << result->Name

<< ", население: " << result->Population;

}

else {

cout << "\n Държавата не е намерана!";

}

break;

}

case 4: {

cout << "Въведете името на държавата за редактиране: ";

cin >> name;

change(name, root);

break;

}

case 5: {

cout << "Въведи началото на интервала: ";

cin >> ch1;

cout << "Въведи края на интервала: ";

cin >> ch2;

while (ch1 >= ch2) {

cout << "Въведи края на интервала: ";

cin >> ch2;

}

if (root) {

search2(ch1, ch2, root);

}

break;

}

}

} while (c != 0);

system("pause");

return 0;

}