

Автор: Єлєсін Артем , КІТ-1196

Дата: 01.06.2020

Лабораторна робота 4. РЕГУЛЯРНІ ВИРАЗИ

Мета роботи: отримати знання про базові регулярні вирази та досвід роботи із застосування їх на практиці.

Загальне завдання

Поширити попередню лабораторну роботу таким чином:

- при введенні інформації про базовий клас (нема різниці, чи з клавіатури, чи з файлу), організувати перевірку відповідності таким критеріям з використанням регулярних виразів:

- можна вводити тільки кириличні символи, латинські символи, цифри, пропуски, розділові знаки;

- не повинно бути пропусків та розділових знаків, які повторюються;

- перше слово не повинно починатися з маленького символу;

- у клас-список додати метод, що виводить на екран список усіх об'єктів, які мають одне або більше полів з щонайменше двома словами (перевірку організувати за допомогою регулярних виразів).

Опис класів

Базовий клас: Ccooperator

Клас, що має в собі масив базового класу та методи для роботи з ним: CList

Опис змінних

const char* name – ім'я.

int amount - кількість елементів

Ccooperator* fEl - 1 масив

Ccooperator* fEl1 - 2 масив

int id - Id персони

int age - вік

int salary -заробітна плата

Опис методів

void setId(const int id); - встановлює id.

void setAge(const int age);- встановлює вік.

void setSalary(const int salary); - встановлює заробітну плату.

int getId()const; - повертає id.

int getAge()const; - повертає вік.

int getSalary()const; - повертає заробітну плату.

Ccooperator(); - конструктор.

Ccooperator(int a, int b, int c, const char* d); - конструктор с параметрами.

Ccooperator(const Ccooperator& a) – конструктор копіювання.

~Ccooperator() { } – деструктор.

int averageSalary() – середня заробітна плата.

void creatMass(int a); - створює масив.

cooperator creatEl1(); - створює елемент.

cooperator creatEl2(); - створює елемент.

void Add(cooperator); - додає елемент.

void Delete(int); - видаляє елемент.

cooperator getCooperator(int a); - повертає елемент.

void showAll(); - показує всі елементи.

cooperator findCooperator(const int a); - знаходить елемент.

int getAmount(); - повертає кількість елементів.

void End(); - видаляє всі масиви.

Текст програми

Cooperator.h

```
#pragma once
#include <sstream>
#include <iostream>
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>
#include <locale>
using namespace std;
class Ccooperator
{
private:
    int id, age, salary;
    string name;
public:
    void setName(string name);
    Ccooperator();
    Ccooperator(int a, int b, int c, const char* d);
    Ccooperator(const Ccooperator& a);
    void setId(const int id);
    void setAge(const int age);
    void setSalary(const int salary);
    string getName();
    int getId()const;
    int getAge()const;
    int getSalary()const;
    ~Ccooperator() {};
};
```

Cooperator.cpp

```
#include "cooperator.h"
void Ccooperator::setName(string name) {
    this->name = name;
}
string Ccooperator::getName() {
    return this->name;
}
void Ccooperator::setId(const int id) {
    this->id = id;
}
void Ccooperator::setAge(const int age) {
    this->age = age;
}
```

```

}
void Ccooperator::setSalary(const int salary) {
    this->salary = salary;
}
int Ccooperator::getId()const {
    return this->id;
}
int Ccooperator::getAge()const {
    return this->age;
}
int Ccooperator::getSalary()const {
    return this->salary;
}
Ccooperator::Ccooperator() :id(0), age(0), salary(0), name("Ivan") {
    cout << "\nБыл вызван конструктор по умолчанию в объекте с id: " << id<<"\n";
}
Ccooperator::Ccooperator(const Ccooperator& a) :id(a.id), age(a.age), salary(a.salary),
name(a.name) {
    cout << "\nБыл вызван конструктор по умолчанию в объекте с id: " << id<<"\n";
};
Ccooperator::Ccooperator(int a , int b , int c , const char* d ) :id(a), age(b),
salary(c), name(d) {
    cout << "\nБыл вызван конструктор по умолчанию в объекте с id: " << id<<"\n";}

```

List.h

```

#pragma once
#include "cooperator.h"
#include "heder.h"
#include <regex>
class CList {
private:
    int amount;
    Ccooperator* fEl;
    Ccooperator* fEl1;
public:
    void twoWorld();
    int averageSalary();
    void writeToFile(string fileName);
    void readFromFile(string fileName);
    void creatMass(int a);
    //Ccooperator creatEl1();
    Ccooperator creatEl2();
    void Add(Ccooperator);
    void Delete(int b);
    void AddWhithString();
    string findCooperator(const int a);
    Ccooperator getCooperator(int a);
    void showAll();
    int getAmount();
    void End();
};

```

List.cpp

```

#include "list.h"
#include <sstream>
#include <iostream>

```

```

#include <fstream>

void CList::creatMass(int a)
{
    amount = a;
    //printf("Введите количество элементов ");
    //scanf("%i", &amount);
    fEl = new Ccooperator[amount];
    //int a;
    //printf("\nВыберите вариант создания элементов\n1. Создать элемент вручную\n2.
Готовый элемент\nВаш выбор: ");
    //scanf("%i", &a);
    //if (a == 1)
    //for (int i = 0; i < amount; i++) {
        //fEl[i] = creatEl1();
    //}
    //if (a == 2)
    for (int i = 0; i < amount; i++) {
        fEl[i] = creatEl2();
    }
}

string CList::findCooperator(const int a) {
    std::stringstream ss;
    string ab;
    ss << " ";
    int b = -1, count = 0;
    for (int i = 0; i < amount; i++) {
        if (a == fEl[i].getId()) {
            count++;
            b = i;
        }
    }
    if (count >= 1) {
        cout << "Есть " << count << " похожих элементов, будет возвращен последний
элемент";
        ss << "\nId: " << fEl[b].getId() << "\nAge:" << fEl[b].getAge() <<
"\nSalary: " << fEl[b].getSalary() << "\nName " << fEl[b].getName();
        ab = ss.str();
        return ab;
    }
    if (count == 0) {
        cout << "Похожих элементов нет, возвращен пустой символ";
        return ab;
    }
}

int CList::averageSalary() {
    int averageSalary = 0;
    for (int i = 0; i < amount; i++)
    {
        averageSalary = averageSalary + fEl[i].getSalary();
    }
    return averageSalary = averageSalary / amount;
};

Ccooperator CList::creatEl2() {
    Ccooperator El;
    El.setId(0);
    El.setSalary(0);
}

```

```

        El.setAge(0);
        return El;
    }
    void CList::readFromFile(string fileName) {
        End();
        creatMass(0);
        int integer;
        string line;
        string str;
        string tstr="";
        Ccooperator temp;
        ifstream file;
        file.open(fileName);
        if (!file.is_open())
        {
            cout << " Файл не открыт, давай по новой Миша\n";
            return;
        }
        std::regex regular_main(
            "([0-9]*[ \\f\\n\\r\\t\\v]{1})"
            "([0-9]*[ \\f\\n\\r\\t\\v]{1})"
            "([0-9]*[ \\f\\n\\r\\t\\v]{1})"
            "([A-Z A-Я a-z a-я 0-9]*[\\.\\,\\;\\:\\-]{0,1})"
            "([ \\f\\n\\r\\t\\v]{0,1}[A-Z A-Я a-z a-я 0-9]*[\\.\\,\\;\\:\\-]{0,1}[ \\f\\n\\r\\t\\v]{0,1})*"
        );
        std::cmatch result;
        std::stringstream ss;
        while (getline(file, line)) {

            if (regex_match(line.c_str(), result, regular_main)) {
                ss << result[1];
                ss >> integer;
                temp.setId(integer);
                ss.clear();
                ss << result[2];
                ss >> integer;
                temp.setAge(integer);
                ss.clear();
                ss << result[3];
                ss >> integer;
                temp.setSalary(integer);

                for (int i = 4; i < result.size(); i++)
                {
                    ss.clear();
                    ss << result[i];
                    ss >> str;
                    if ((str+" ")!=tstr)
                        tstr = tstr + str + " ";
                }

                temp.setName(tstr);
                tstr = "";
                Add(temp);
            }
            result.end();
        }
        file.close();
    }

```

```

}
void CList::twoWorld() {
    std::regex regular("([A-Z A-Я a-z a-я 0-9][\\.\\,\\;\\:\\-
]{0,1}[ \\f\\n\\r\\t\\v]{1}[A-Z A-Я a-z a-я 0-9][\\.\\,\\;\\:\\-
]{0,1}[ \\f\\n\\r\\t\\v]{0,1}){1}");
    std::cmatch result;
    for(int i=0;i<amount;i++)
        if (regex_match(fEl[i].getName(), regular)) {
            cout <<"\\n"<< "ID:" << fEl[i].getId() << "\\n Age: " <<
fEl[i].getAge() << "\\n Salary: " << fEl[i].getSalary() << "\\n Name: " << fEl[i].getName();
        }
}
void CList::writeToFile(string fileName) {
    ofstream file;
    string str1;
    std::stringstream ss;
    file.open(fileName);
    if (!file.is_open())
    {
        cout << " Файл не открыт, давай по новой Миша\\n";
        return;
    }
    for (int i = 0; i < amount; i++) {
        file << fEl[i].getId() << " " << fEl[i].getAge() << " " <<
fEl[i].getSalary() << " " << fEl[i].getName() << "\\n";
    }

    file.close();
}
void CList::AddWhithString() {
    Ccooperator temp;
    std::stringstream ss1, ss2, ss3, ss4;
    fEl1 = new Ccooperator[amount + 1];
    for (int i = 0; i < amount; i++) {
        fEl1[i] = fEl[i];
    }

    std::cout << "\\nВведите данные с клавиатуры в таком порядке: id, age, salary,
name\\n";
    string tid = " ", tage = " ", tsalary = " ", tname = " ";
    int tid1, tage1, tsalary1;
    string tname1 = " ";
    cin >> tid >> tage >> tsalary >> tname;
    ss1 << tid;
    ss1 >> tid1;
    ss2 << tage;
    ss2 >> tage1;
    ss3 << tsalary;
    ss3 >> tsalary1;
    ss4 << tname;
    ss4 >> tname1;
    temp.setId(tid1);
    temp.setSalary(tsalary1);
    temp.setAge(tage1);
    temp.setName(tname1);

    fEl1[amount] = temp;
    delete[] fEl;
}

```

```

        amount++;
        fE1 = new Ccooperator[amount];
        for (int i = 0; i < amount; i++) {
            fE1[i] = fE11[i];
        }
        delete[] fE11;
    }
    void CList::Add(Ccooperator E11) {
        fE11 = new Ccooperator[amount + 1];
        for (int i = 0; i < amount; i++) {
            fE11[i] = fE1[i];
        }
        fE11[amount] = E11;
        delete[] fE1;
        amount++;
        fE1 = new Ccooperator[amount];
        for (int i = 0; i < amount; i++) {
            fE1[i] = fE11[i];
        }
        delete[] fE11;
    }
    int CList::getAmount() {
        return amount;
    }
    void CList::Delete(int a) {
        Ccooperator* fE11 = new Ccooperator[amount - 1];
        for (int i = 0; i < a - 1; i++) {
            fE11[i] = fE1[i];
        }
        for (int i = a - 1, j = a; j < amount; i++, j++) {
            fE11[i] = fE1[j];
        }
        delete[] fE1;
        amount--;
        fE1 = new Ccooperator[amount];
        for (int i = 0; i < amount; i++) {
            fE1[i] = fE11[i];
        }
        delete[] fE11;
    }
    Ccooperator CList::getCooperator(const int a) {
        return fE1[a];
    }
    void CList::showAll() {
        for (int i = 0; i < amount; i++) {
            cout << "ID:" << getCooperator(i).getId() << "\n Age: " <<
getCooperator(i).getAge() << "\n Salary: " << getCooperator(i).getSalary() << "\n Name: "
<< getCooperator(i).getName();
        }
    }
    void CList::End() {
        delete[] fE1;
    }
}

```

Test.cpp

```

#include "cooperator.h"
#include "list.h"

```



```

#define _CRT_SECURE_NO_WARNINGS
#define N 5
#include <iostream>
#include <locale>
using namespace std;
int main() {
    setlocale(LC_ALL, "rus");
    Ccooperator a;
    a.setAge(0);
    a.setAge(0);
    a.setSalary(0);
    CList a1[N];
    int test[N];
    int rezult1[N];
    int rezult2[N];
    int rezult3[N];
    test[0] = 1;
    test[1] = 5;
    test[2] = 10;
    test[3] = 25;
    test[4] = 50;
    rezult1[0] = 1;
    rezult1[1] = 5;
    rezult1[2] = 10;
    rezult1[3] = 25;
    rezult1[4] = 50;
    rezult2[0] = 2;
    rezult2[1] = 6;
    rezult2[2] = 11;
    rezult2[3] = 26;
    rezult2[4] = 51;
    rezult3[0] = 1;
    rezult3[1] = 5;
    rezult3[2] = 10;
    rezult3[3] = 25;
    rezult3[4] = 50;
    for (int i = 0; i < N; i++) {
        a1[i].creatMass(test[i]);
        if (a1[i].getAmount() == rezult1[i]) {
            printf("Тест 1.%i пройден\n", i);
        }
        else {
            printf("Тест 1.%i не пройден\n", i);
        }
    }
    for (int i = 0; i < N; i++) {
        a1[i].Add(a);
        if (a1[i].getAmount() == rezult2[i]) {
            printf("Тест 2.%i пройден\n", i);
        }
        else {
            printf("Тест 2.%i не пройден\n", i);
        }
    }
    for (int i = 0; i < N; i++) {
        a1[i].Delete(test[i]);
        if (a1[i].getAmount() == rezult3[i]) {
            printf("Тест 3.%i пройден\n", i);
        }
    }
}

```

```

    }
    else {
        printf("Тест 3. %i не пройден\n", i);
    }
}
Ccooperator Obtest1, Obtest2;
Obtest1.setSalary(-200);
Obtest2.setSalary(300);
CList TestList;
TestList.creatMass(0);
TestList.Add(Obtest1);
TestList.Add(Obtest2);
if (TestList.averageSalary() == 50) {
    printf("Тест 4 пройден\n");
}
else {
    printf("Тест 4 не пройден\n");
}
TestList.readFromFile("File.txt");
if (TestList.getAmount() == 3)
    printf("Тест 5 пройден\n");
else
    printf("Тест 5 не пройден\n");
}

```

Висновок

При виконанні даної лабораторної роботи було набуто практичного досвіду роботи з регулярними виразами.

Було розроблено програму що перевіряє читання з файлу за допомогою регулярних виразів.

Регулярні вирази дуже корисна бібліотека яка допомагає шукати потрібні данні в тексті(потоці), за допомогою шаблонних виразів.

Програма протестована, витоків пам'яті немає, виконується без помилок.