Projeto 1: Expressões Regulares em C++

Teoria da Computação

Alunos: Fábio Melo, Luigge Lena, Valber Moreira

Implementação:

Na preparação deste trabalho, utilizamos as seguintes funções de manipulação de regex incluídas na biblioteca padrão do C++11.

```
#include <regex>
bool regex_match(string s, regex meu_regex);
// procura por padrões de expressões regulares e retorna TRUE
se achar
bool regex_search(string s, smatch m, regex meu_regex);
//procura por expressões regulares e subexpressões e retorna os
resultados em 'm'
```

Para achar os padrões especificados, utilizamos as seguintes expressões regulares:

Padrão	Expressão Regular	Exemplo de Aceitação
CPF	[0-9]{3}.[0-9]{3}.[0-9]{3}-[0-9]{2}	123.456.789-01
Telefone	[0-9]{4,5}-[0-9]{4}	1234-5678 ou 12345-5678
Placa	[A-Z]{3}-[0-9]{4}	ABC-1234

Pesquisando por Expressões Regulares em uma String

A função **regex_search()** pode ser utilizada para localizar padrões determinados por expressões regulares, como demonstrado no programa abaixo:

```
void verificar_texto(){

    //String de testes
    string texto[] = {"meu texto tem um CPF 583.384.283-
49","meu texto tem um telefone 2485-4858", "meu texto gosta de
AAA-1234" };
    //expressoes utilizadas
    regex regex_cpf("[0-9]{3}\\.[0-9]{3}\\.[0-9]{3}-[0-9]{2}");
    regex regex_telefone("[0-9]{4,5}-[0-9]{4}");
    regex regex_placa("[A-Z]{3}-[0-9]{4}");

    smatch match; //variavel que armazena os resultados

/* laço que busca pelas expressões no texto */
     for (const auto& linha : texto) {
        if(std::regex_search(linha, match, regex_cpf)) {
            for (size_t i = 0; i < match.size(); ++i)
                std::cout << "CPF: " << match[i] << '\n';
        }
        else if(std::regex_search(linha, match,
regex_telefone)) {
            for (size_t i = 0; i < match.size(); ++i)
                std::cout << "Telefone: " << match[i] << '\n';
        }
        else if(std::regex_search(linha, match, regex_placa)) {
            for (size_t i = 0; i < match.size(); ++i)
                std::cout << "Placa: " << match[i] << '\n';
        }

    }
}
```

Saída do Código

```
CPF: 583.384.283-49 Telefone: 2485-4858 Placa: AAA-1234
```

Verificando a Validade de Expressões Regulares

Para que possamos verificar a existência desses padrões na entrada do usuário em um programa de C++, precisamos declarar um objeto *regex* com o valor da expressão procurada e passá-lo como parâmetro na função **regex_match()**; e usar o booleano retornado como resultado.

CPF

```
void verifica_CPF(){
    string cpf;
    regex regex_cpf("[0-9]{3}\\.[0-9]{3}\\.[0-9]{3}-[0-9]{2}");
    cout<<"Digite o Numero do CPF:";
    cin>>cpf;
    if(regex_match(cpf, regex_cpf)){
        cout<< "Formato de CPF VALIDO\n";}
    else cout<< "Formato de CPF INVALIDO\n";
```

Exemplos de Entrada:

> 492.482.439-12 output: Formato de CPF VALIDO > 3483242.394 output: Formato de CPF INVALIDO

Numeros de Telefone (8 ou 9 dígitos)

```
void verifica_numeroTelefone(){
    string telefone;
    regex regex_telefone("[0-9]{4,5}-[0-9]{4}");
    cout<<"Digite o Numero do Telefone\n";
    cin>>telefone;
    if(regex_match(telefone,regex_telefone)){
        cout<< "Formato de Telefone VALIDO\n";
    }else cout<< "Formato de Telefone INVALIDO\n";
}
```

Exemplos de Entrada:

> 98777-3829 output: Formato de Telefone VALIDO > 991242.394 output: Formato de Telefone INVALIDO

Placas de Carro

```
void verifica_placaCarro(){
    string placa;
    regex regex_placa("[A-Z]{3}-[0-9]{4}");
    cout<< "Digite a Placa do Automovel\n";
    cin>>placa;
    if(regex_match(placa,regex_placa)){
        cout<< "Formato de Placa VALIDO\n";}
        else cout<< "Formato de Placa INVALIDO\n";
    }
```

Exemplos de Entrada:

> AER-3943 output: Formato de Placa VALIDO > II-394 output: Formato de Placa INVALIDO

Código Completo do Programa

```
#include <iostream>
#include <string>
#include <regex>
#include <vector>

using namespace std;



void menu(){
    cout<< "\nEscolha o que deseja fazer\n";
    cout<< "1 - Verificar CPF\n";
    cout<< "2 - Verificar Numero de Telefone\n";
    cout<< "3 - Verificar Placa de Carro \n";
        cout<< "4 - Verificar Texto Digitado\n";
    cout<< "5 - Sair\n";
    cout<< "Sua Escolha: ";
}

void verifica_CPF(){
    string cpf;
    regex regex_cpf("[0-9]{3}\\.[0-9]{3}\\.[0-9]{3}-[0-9]{2}");
    cout<<"Digite o Numero do CPF:";
    cin>>cpf;
    if(regex_match(cpf, regex_cpf)){
        cout<< "Formato de CPF VALIDO\n";}
    else cout<< "Formato de CPF INVALIDO\n";

}
void verifica_numeroTelefone(){
    string telefone;
    regex regex_telefone("[0-9]{4,5}-[0-9]{4}");
    cout<<"Digite o Numero do Telefone\n";
    cin>>telefone;
    if(regex_match(telefone,regex_telefone)){
        cout<< "Formato de Telefone VALIDO\n";
    }else cout<< "Formato de Telefone INVALIDO\n";
}

void verifica_placaCarro(){
    string placa;
    regex regex_placa("[A-Z]{3}-[0-9]{4}");
    cout<< "Digite a Placa do Automovel\n";
    cin>>placa;
    if(regex_match(placa,regex_placa)){
        cout<< "Formato de Placa VALIDO\n";}
        else cout<< "Formato de Placa INVALIDO\n";
    }

void verificar_texto(){

    vector<string> texto;
```

```
    string temp;

    regex regex_cpf("[0-9]{3}\\.[0-9]{3}\\.[0-9]{3}-[0-9]{2}");
    regex regex_telefone("[0-9]{4,5}-[0-9]{4}");
    regex regex_placa("[A-Z]{3}-[0-9]{4}");

    smatch match;

    cout<< "Digite o texto que deve procurar, -q para terminar
a entrada\n";

    while(true){
        cin>> temp;
        texto.push_back(temp);
        if(temp == "-q"){
            break;
        }
    }

     for (const auto& linha : texto) {
        if(std::regex_search(linha, match, regex_cpf)) {
            for (size_t i = 0; i < match.size(); ++i)
                std::cout << "CPF:" << match[i] << '\n';
        }
        else if(std::regex_search(linha, match,
regex_telefone)) {
            for (size_t i = 0; i < match.size(); ++i)
                std::cout << "Telefone:" << match[i] << '\n';
        }
        else if(std::regex_search(linha, match, regex_placa)) {
            for (size_t i = 0; i < match.size(); ++i)
                std::cout << "Placa:" << match[i] << '\n';
        }

    }
}



int main(int argc, char const *argv[])
{
    int op;
    while(true){
        menu();
        cin>>op;
        switch(op){
            case 1:
                verifica_CPF();
                break;
            case 2:
                verifica_numeroTelefone();
                break;
            case 3:
```

```
                verifica_placaCarro();
                break;
                    case 4:
                        verificar_texto();
                break;
            case 5:
                exit(0);
            default:
                cout<<"Opcao Invalida\n";
        }}

    return 0;
}
```