Practica 2

Alumno: *Ángel López Manríquez*
Tema: *Expresiones regulares*
Fecha: *17 de junio de 2019*
Grupo: *2CV1*

M. en C. Luz María Sánchez García

Instituto Politecnico Nacional
Escuela Superior de Cómputo

Expresiones regulares

Lopez Manriquez Angel 2CV1

Marzo 2018

1. Introducción

En esta práctica implementaremos un programa que, dada una expresión regular sobre un alfabeto Σ, verifique si diversas cadenas pertenecen o no al lenguaje generado por dicha expresión.

Una expresión regular es una forma abreviada de representar cadenas de caracteres que se ajustan a un determinado patrón. Al conjunto de cadenas representado por la expresión r se lo llama lenguaje generado por la expresión regular r y se escribe como $L(r)$.

Una expresión regular se define sobre un alfabeto Σ y es una cadena formada por caracteres de dicho alfabeto y por una serie de operadores también llamadas metacaracteres, tales como paréntesis, corchetes, llaves, guión, unión (|), cerradura opcional (?), cerradura positiva (+) y de Kleene (*).

En esta práctica, la expresión regular podrá contener operaciones de:

1. Concatenación. Ejemplo: 01.
2. Unión. Ejemplo: (01)|(10)
3. Cerradura (opcional, positiva y de Kleene). Ejemplo: $aa(a|b)^*$.

2. Planteamiento del problema

En la practica se pide hacer un programa que verifique si una directiva de preprocesamiento del lenguaje C esta bien escrita. por ejemplo `#include <stdio.h>`, `#include<string.h>` (note la falta de espacio), `#include <stdlib.h>` son correctas, a diferencia de `import java.util.*;`, `#include <iostream>`. Para la practica se asume que el archivo cabecera existe en el directorio donde se instalaron las bibliotecas de C.

3. Implementación de la solución

Para dar solucion al problema, utilizaremos la biblioteca `regex` de C++ que viene para versiones mayor a la 11. En esta biblioteca viene una clase `regex` cuyo constructor recibe una cadena en la cual viene la expresion regular.

```
1 /*  Practica 2: expresiones regulares
2     Autor: Angel Lopez Manriquez
3     Grupo: 2CV1
4     ================================================================================
5         Programa de consola que determina si la cabecera para incluir bibliotecas en
6       C es correcta. Se asume que la cabecera esta en la carpeta donde se instala
```

```
7         el lenguaje.
8
9       Caracteristicas ------------------------------------------------------------------
10        Hacemos uso de la clase regex en conjunto con sus funciones amigas. Al crear
11        la variable de instancia le pasamos al constructor la expresion regular.
12
13      Compilacion
    ↪   ------------------------------------------------------------------------------
14        g++ pr2.cpp -std=c++11
15        es importante que la version del compilador g++ sea 11 o superior, puesto
16        que desde esta version se incluyo la biblioteca regex. */
17
18  #include <iostream>
19  #include <regex>
20  #include <string>
21
22  using namespace std;
23
24  int main(void) {
25      // expresion regular que acepta cualquier caracter del alfabeto ingles o digito
26      string valid_char = "([A-Z]|[a-z]|\\d)",
27             include = "#include\\s?((<((" + valid_char + ")+).h>))\\s*",
28             input; // cadena que guardara la expresion a validar
29      regex pattern(include); // creamos una variable de instancia tipo regex
30      getline(cin, input); // leemos todo el texto de una linea ingresado por teclado
31      bool accepted = regex_match(input, pattern);
32      cout << (accepted ? "Cadena valida" : "cadena invalida") << endl;
33      return 0;
34  }
```

```
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
ang3l@n4m3l3ss:~/dev/theory-of-computation/practice2$ ls
a.out  Makefile  pr2.cpp
ang3l@n4m3l3ss:~/dev/theory-of-computation/practice2$ make
g++ pr2.cpp -std=c++14
ang3l@n4m3l3ss:~/dev/theory-of-computation/practice2$ ./a.out
#include <stdio.h>
Cadena valida
ang3l@n4m3l3ss:~/dev/theory-of-computation/practice2$ ./a.out
#include <stdlib.h>
Cadena valida
ang3l@n4m3l3ss:~/dev/theory-of-computation/practice2$ ./a.out
#include<string.h>
Cadena valida
ang3l@n4m3l3ss:~/dev/theory-of-computation/practice2$ ./a.out
#include <iostream>
cadena invalida
ang3l@n4m3l3ss:~/dev/theory-of-computation/practice2$ ./a.out
#inc <complex.h>
cadena invalida
ang3l@n4m3l3ss:~/dev/theory-of-computation/practice2$ ./a.out
#include <assert.h
cadena invalida
ang3l@n4m3l3ss:~/dev/theory-of-computation/practice2$ ./a.out
import java.util.*;
cadena invalida
ang3l@n4m3l3ss:~/dev/theory-of-computation/practice2$ ./a.out
library ieee;
cadena invalida
ang3l@n4m3l3ss:~/dev/theory-of-computation/practice2$
```