Task URL-Parser Explanation

As per the task I was asked to parse the URL into various components like SCHEME,HOST,PORT,PATH and PARAMS through Regex and State Machine and compute performance of individual algorithm. I have taken the assumption that URL given will be valid and all the parameters will be there in order to parse them successfully.

Part1 RegEx:

I have constructed RegEx `(\\w+)://([a-zA-Z0-9\\.-_]+?):?(\\d+)?/([^?]+)\\??(.*)?"` which will parse the URL into groups: first group: initially it will accept any word character "`(\\w+)`" uptill "://" ; second group will accept the all the English alphabet either small or capital and with this it can also accept the number and all weird character like (\ . - _);The third group parses the port which I've set optional `:?(\\d+),` the forth group; `/([^?]+)` it accepts all the character after the / .In the fifth group I'm parsing the query `\\??(.*)?` it can precede with "?" as optional.

```
values=new LinkedHashSet<>();
```

For storing the parsed elements of groups I am using Linked Hashset reason being is, LinkedHashSet lets you iterate through the elements in the order in which they are inserted.

```
long t=System.currentTimeMillis();
for(int i=0;i<loopCount;i++)

	regexParse(p,url);

return System.currentTimeMillis()-t;
```

The heart of the RegEx program is this which calculates the performance. For each iteration in the forloop method `regexParse(p,url)` is called and passed with pattern object and the url object.

```
public static void regexParse(Pattern p,String url) {

		Matcher matcher = p.matcher(url);/creating a matcher object which is accepting text value
		matcher.find(); //getting the Stings based on groups

		String s=matcher.group(1);    //s="http"
		if(s!=null&&!s.isEmpty())
			values.add(s);     // Appending "http" to s

		s=matcher.group(2);                  //s= "host"
		if(s!=null&&!s.isEmpty())
			values.add(s);

		s=matcher.group(3);                  //s=port
		if(s!=null&&!s.isEmpty())
			values.add(s);
		 . . . . . .
```

When `regexParse(p,url)` is called ate every iteration it creates the matcher object of class Matcher and with the help of this matcher objects it groups the

string based on different groups.After that in the subsequent code the strings in the group is added to the *vallues* which is object of LinkedHashSet.

PartII State Machine

For parsing the URL I'm using the java.netURL library.
Performance Calculation:

```
long t=System.currentTimeMillis();

            for(int i=0;i<loopCount;i++)

                  stateParse(0,new URL(url));

            return System.currentTimeMillis()-t;

      }
```

In every loop of iteration this ,,stateParse(0,new URL(url))" will create a new object and it'll take some clock ticks in creation and do background parsing because of that we'll see change in the performance. If I make the URL object before the for loop then it will always prompt me 1ms.

As this stateParse will iterate it will do recursive call to itself changing from State 0..4 and eventually to -1 (termination state)

```
public static void stateParse(int state, URL url) {
            /*
             * In this state=0 is starting state and state=-1 is termination
                  state
             *
             */
            if (state < 0)
                  return;

            switch (state) {
            case 0:// get scheme here
                  stateValues.add(url.getProtocol());
                  //url.getProtocol();
                  state = 1;
                  break;

            case 1:// get host here
                  stateValues.add(url.getHost());
                  //url.getHost();
                  state = 2;
                  break;

            case 2:// get port here
                  stateValues.add(url.getPort()+"");
                  //url.getPort();
                  state = 3;
                  break;
            case 3:// get path here
                  stateValues.add(url.getPath());
                  //url.getPath();
                  state = 4;
                  break;
            case 4:// get parameters here
                  stateValues.add(url.getQuery());
```

```
                        //url.getQuery();
                        state = -1;
                        break;

                default:
                        state = -1;
                        break;
                }

                stateParse(state, url); //calling it recursively
        }
```

PartIII Printing the values

```
private static void printValues() {

                System.out.println("Output from RegEx:");
                for(String s:values)
                        System.out.println(s);

                System.out.println("\nOutput from StateMachine:");
                for(String s: stateValues)
                        System.out.println(s);
                System.out.println("\nPerformance of Two
        Algorithms:\n"+"Regex:"+regexTime+"msec");
                System.out.println("State:"+stateMachineTime+"msec");

         }
```

PartIV Output

```
<terminated> UrlParser [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (Nov 19, 2017, 3:38:38 AM)
Output from RegEx:
http
host
8090
path
params

Output from StateMachine:
http
host
8090
/path
params

Performance of Two Algorithms:
Regex:76msec
State:61msec
```

References:

1)http://tutorials.jenkov.com/java-regex/matcher.html#java-matcher-example
2)http://www.vogella.com/tutorials/JavaRegularExpressions/article.html#pattern-and-matcher
3)https://docs.oracle.com/javase/7/docs/api/java/net/URL.html