Regular expressions are kind of like a whole programming language that is in characters. They are used for string matching and extraction. Let’s say you have a huge file that has a bunch of email addresses and then you want to find actual home addresses in there, you can use regular expressions to do that just by specifying a set of characters it should match. It’s like a wild card for matching and parsing strings. To use regular expressions in python, you have to import regular expressions in to your code.

<code>import re</code>

**Quick Guide**

^ - matches the beginning of a line

$ - matches the end of a line

. – matches any characters

\s – matches whitespace (e.g. )

\S – matches non-whitespace characters

\* - repeats a character zero or more times

\*? - repeats a character zero or more times (non-greedy)

+ - repeats a character one or more times

+? - repeats a character one or more times (non-greedy)

[aeiou] – matches a single character in the listed set

[^XYZ] – matches a single character not in the listed set

[a-z0-9] – the set of characters can include a range

(- indicates where to start extracting the string

) – indicates where to top extracting the string

**Examples**

Let’s say we have a file that has a million lines and we want to get only the lines that start with “F” and then also have “:” in them, we can use <code>^F.\*:<code>. In plain English, what this line is saying is find all the lines that start with (^) F, include any character zero or more times (.\*) and then stop at : so we can get back lines like “From:”, “Fr0m:”, “F:” etc. we can fine tune this more.

For the regular expression characters that have the square brackets, you can look at that square bracket and all its content as one single character. It is like a filter that matches a single character of what is inside it. If you want to match more than one, you can add the <code>+</code>

Import re

X = “My 2 favorite numbers are 7 and 23”

Y = re.findall(“[0-9]+”, x)

Print (y)

#[“2”, “7”, “23”]

<code>re.findall()</code> method takes in two arguments here. The first one is the regular expression and the second one is the variable containing the string we want to search through. In simple English what that line of code <code> Y = re.findall(“[0-9]+”, x)</code> is doing is you are looking for a number in the string(<code>[0-9+</code>) one or more times hence the plus after the [0-9]. The result is a list that contains strings that match the criteria. It is important to note that it is searching through a string so it returns strings. You can convert the strings to numbers later if you like.<br />

<strong>Greedy matching</strong>

The repeat characters (\* and +) push outward in both directions (greedy) to match the largest possible string.

Import re

X = “From: using the : character”

Y = re.findall(“^F.+:”, x)

Print(y)

#[“From: using the :”]

It did not return “From:” because it was a greedy matching. What the regular expression is saying above is match any line that starts with F (^) followed by any character one or more times (.+) and then ends at “:”. The result we got back matched that criteria given however if we want just the “From:” we can make it non-greedy by just adding in the “?” character.

Import re

X = “From: using the : character”

Y = re.findall(“^F.+?:”, x)

Print(y)

#[“From:”]

This is going to match any character that starts with F followed by any character one or more times but don’t be greedy so it would stop at the first “:”.

<br /><br />

You can fine-tune string extraction by using parenthesis. Parenthesis tells the regular expression where to start extracting and where to stop.

Import re

X = “From [Daniel.dumebi@stu.cu.edu.ng](mailto:Daniel.dumebi@stu.cu.edu.ng) Thursday 14 January 2021”

Y = re.findall(“\S+@\S+”, x)

Print(y)

#[“[Daniel.dumebi@stu.cu.edu.ng](mailto:Daniel.dumebi@stu.cu.edu.ng)”]

The code above would only print out the email address but let’s say we have a file that has many email addresses but we only want to print out the ones that are on a line that starts with from, we can do that by finding the lines that have the from and then start our extraction from the point we want like so.

Import re

X = “From [Daniel.dumebi@stu.cu.edu.ng](mailto:Daniel.dumebi@stu.cu.edu.ng) Thursday 14 January 2021”

Y = re.findall(“From (\S+@\S+)”, x)

Print(y)

#[“[Daniel.dumebi@stu.cu.edu.ng](mailto:Daniel.dumebi@stu.cu.edu.ng)”]

In simple English, it would match the line that has “From” followed by space and then start extracting from after the space then it should find any non-blank characters one or more times (\S+) followed by an @ sign followed by another non blank character. It cuts off at .ng because after it is a whitespace character. This matching however is greedy. If we added question mark to our +, our result would have been i@s.

Import re

X = “From [Daniel.dumebi@stu.cu.edu.ng](mailto:Daniel.dumebi@stu.cu.edu.ng)”

Y = re.findall(“@([^ ]\*)”, x)

Print(y)

#[“stu.cu.edu.ng”]

What the code above is doing is it is going to find where the is an @ sign and then start extracting after the @ sign anything that is non blank ([^ ]) zero or more times. In regards to <code>[^ ]</code>, when you have a “^” inside the square bracket it means <em>not</em>. Here is it followed by an empty space so it means not empty space. Of course this is the same thing as <code>\S</code> but then you get the idea of what “^” does inside of the square brackets.

<br /><br />

You can escape certain regular expression characters. For example:

Import re

X = “we just received $10.00 for cookies”

Y = re.findall(“\$[0-9.]+”, x)

Print(y)

#[“$10.00”]

<code>$</code> normally is a regular expression character but since we want to get the “$10.00” we have to escape it so the program doesn’t see it as line end. <code>[0-9.]+</code> is get any character that is a number or dot (because we want the decimal) one or more times.<br />

<br />

Cheers 🥂