Homework Assignment 2

Name: **Asrar Syed**

Due by: **October 04, 2024**

Course Section: **CSC 3320-034**

1. Assume we are using extended regex (?, +, { }, \*, [ ], |, and () are special characters NOT literals). Now, answer the following questions:

i) (2 points) Which of the following strings match(es) with **‘a[ab]\*a’**? Multiple matches are possible. Give your explanation.  
A. ababa  
B. aaba  
~~C. aabab~~  
D. aabbaa  
E. aa

**Answer**: A,B,D,E

**Explanation**: The matched strings must begin and end with an **‘a’** and there can be zero or more occurances of **‘a’** or **‘b’** in between, irregardless of any combination of amount of those letters.

ii) (2 points) Which of the following strings match(es) with ‘a(bc)?’? Multiple matches are possible. Give your explanation.  
A. abc  
B. a(bc)?  
C. a(bc  
D. a  
E. abcbc

**Answer**: A

**Explanation**: The matched strings must begin with an **‘a’**, must find the exact letters within the parenthesis in same order and it can only occur once or zero times.

iii) (2 points) Which of the following strings match(es) with ‘.[ind]\*’? Multiple matches are possible. Give your explanation.

A. wind  
B. window  
C. end  
D. good  
E. will

**Answer**: A, B, C, D, E

**Explanation**: The **‘.’** matches all characters except except line breaks, then it looks for any character that matches the characters withen the set **‘[ind]\*’** a checks if it exits zero or more occurances of using **‘\*’**.

iv) (2 points) Which of the following strings match(es) with ‘[a-z]+[a-z]\*’? Multiple matches are possible. Give  
your explanation.  
A. a+b+c  
B. +b  
C. x  
D. x+1  
E. xc

**Answer**: C, E

**Explanation**: **‘[a-z]+’** This matches one or more lowercase alphabetic characters (from a to z).

**‘[a-z]\*’** This part matches zero or more lowercase alphabetic characters (from a to z).

v) (3 points) Which of the following strings match(es) with ‘[a-z](\+[a-z])+’? Multiple matches are possible. Give your explanation.  
A. a+b+c  
B. +b  
C. x  
D. x+1  
E. x + a

**Answer**: A

**Explanation**: **‘[a-z]’** This matches one lowercase alphabetic characters (from a to z). **‘\+’** This matches and actual plus sign character, **‘[a-z]’** This matches one lowercase alphabetic characters (from a to z). Together **‘(\+[a-z])+’** this looks for one or more repetitions of the pattern within the parentheses.

vi) (3 points) Which of the following strings match(es) with ‘[a-z]+[\.\?!]’? Multiple matches are possible. Give your explanation.  
A. good!  
B. bad!  
C. yes?  
D. book.  
E. all of the above.

**Answer**: A, B, C, D

**Explanation**: **‘[a-z]+’** This matches one or more lowercase alphabetic characters (from a to z). **‘[\.\?!]’** this matches for period, question mark and exclamation mark. The period and question mark are special characters and need to be backspaced.

vii) (3 points) Which of the following strings match(es) with ‘(very )+(hot )?(good|bad|stormy)weather’? Multiple matches are possible. Give your explanation.  
A. good weather  
B. very very hot weather  
C. very good weather  
D. very hot bad weather  
E. very stormy weather

**Answer**: C, D, E

**Explanation**: **‘(very )+’** Matches the word "very" one or more times. **‘(hot )?’** Matches the word "hot" zero or one times and the question mark makes "hot" optional. **‘(good|bad|stormy)**’ Matches either "good", "bad", or "stormy" (due to the alternation |). Lastly it matches the exact word **‘weather’** at the end.

viii) (3 points) Which of the following strings match(es) with ‘-?[0-9]\*\.?[0-9]\*’? Multiple matches are possible.  
Give your explanation.  
A. -123  
B. . (just a decimal point)  
C. 123.456  
D. -123.456  
E. all of the above  
F. none of the above

**Answer**: A, B, C, D

**Explanation**: **‘-?’** This looks for an optional minus sign, **‘[0-9]\*’** this part matches zero or more numeric characters. It then looks for an optional period/dot **‘\.?’** and it then searches to match zero or more numeric characters **‘[0-9]\*’**.

2. (10 points) Write a regular expression for variable names which can be between 1 and 10 character long. Variable names must start with a letter or an underscore. The following characters can be letters or underscores or digits.  
Examples of valid variable names include **number**, **\_name1**, **na021b1a2**. Explain your answer.

**Answer**: [a-zA-Z\_][\w]{0,9}

**Explanation**: The items in the first bracket checks if the first character is a letter or an underscore. The second pair of brackets and braces ‘[\w]{0,9}’ matches if the next 9 characters are letters, underscores or digits.

3. (10 points) Write a regular expression for passwords. Assume that the passwords are strings that contain at least 4 but not more than 10 characters. The passwords should have at least one upper case letter, one lower case letter, and one digit. Explain your answer.

**Answer**: : (?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)[a-zA-Z\d]{4,10}

**Explanation**: There is 5 parts to this regex command. I will break down starting with **‘?=.\*’**, that is essentially a combination of a positive lookahead **‘?=’** which matches the expression by looking ahead and sees if there is any pattern that matches my search. This also has ‘.=’ which ensures that the pattern can exist anywhere in the in the string and contain mulptiple of any characters except line breaks. This pattern is combined to search for three things specifically which is at least one upper case letter, one lower case letter, and one digit i.e. ‘**(?=.\*[a-z])’** and **‘(?=.\*[A-Z])’** and **‘(?=.\*\d)’**. Lastly I check if there is a minimum of at least four to a maximum of ten characters using **‘[a-zA-Z\d]{4,10}’**.

4. (10 points) Write a regular expression for valid dates in the format: MM/DD/YYYY. Note 12/04/2005 is a valid date, but 13/20/1998 and 08/32/2012 are not. Explain your answer.

**Answer**: (0[1-9]|1[012])([/])(0[1-9]|[12][0-9]|3[01])\2(19|20)\d\d

**Explanation**: **‘(0[1-9]|1[012])’** this matches the month part of the date and ensures the month is between 01 and 12 by either letting the months match 01-09 or 10, 11, 12. The next part **‘[(/)]’** matches the separator between the parts of the date so it only checks for MM/DD/YYYY using **‘\2’** which is a backreference that ensures the previous symbol occurs twice. **‘(0[1-9]|[12][0-9]|3[01])’** this matches the days and ensures it is between 01 and 31. And the last part checks for the years and matches the 19th and 20th century and the exact year using \d\d.

5. (10 points (bonus)) Write a regular expression for phone numbers in any of the following formats: 999-999-9999 or (999)-999-9999. (Note: these two formats should be matched by a single regular expression).

**Answer**: \(?\d{3}\)?[-]\d{3}[-]\d{4}

**Explanation**: This checks for an option parenthesis **‘\(?’ and ‘\)?’** and in between you will see it check for 3 digits by **‘d{3}’** or 4 digits using **‘d{4}’**. I also check for dashes between using a set **‘[-]’**.