LAPORAN PERTEMUAN KE-4

PRAKTIKUM PERL REGEX

Laporan ini disusun untuk memenuhi Mata Kuliah Paktikum Prinsip Bahasa Pemrograman

Disusun oleh :

Suci Awalia Gardara 211524027

PROGRAM STUDI D4 TEKNIK INFORMATIKA

JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA

POLITEKNIK NEGERI BANDUNG

2022

Perl Regex

1. BasicRegex.pl

Source Code :

```
#!/usr/bin/perl -w
use strict;
my %chars;
my $Para =
"Sunflowers waiting for sunshine. \n
Violets just waiting for dew. \n
Bees just waiting for honey \n
And honey, I'm just waiting for you!";

# Matches 'for'
print "Matched \n"
if ($Para =~ m/for/);
# Matches 'And' at start of string
print "Does not match \n"
if ($Para =~ m/^And/);
# Matches 'And' at start of each line
print "Matches (using modifiers) \n"
if ($Para =~ m/^And/m);

my $group = "abcd";
# Grouping captures matched strings
# 1 2 3 4
$group =~ m/(a(b|c)(c(d)))/;
print "$1, $2, $3, $4 \n";
# Converts lowercase alphabets range [a to m]
# to uppercase
$Para =~ tr/[a-m]/[A-M]/;
print "$Para \n";

# Counts frequency of uppercase alphabets
$Para =~ s/([A-Z])/$chars{$1}++;$1/eg;
print "Frequency of '$_' : $chars{$_} \n"
foreach (sort{$chars{$b} <=> $chars{$a}}
keys %chars);
# If we use "m/avi/" then all four words will
# be matched - not GOOD!
# lookahead(?=) and lookbehind(?<=) to match
my $x = "tavi avi pavi a-avi";
print "Found avi"
```

```perl
if ($x =~ m/(?<=\s)avi(?=\s)/);
```

Output:

```
Matched
Matches (using modifiers)
abcd, b, cd, d
SunFLowErs wAItInG For sunsHInE.

VIoLEts Just wAItInG For DEw.

BEEs Just wAItInG For HonEy

AnD HonEy, I'M Just wAItInG For you!
Frequency of 'I' : 11
Frequency of 'E' : 8
Frequency of 'A' : 5
Frequency of 'F' : 5
Frequency of 'G' : 4
Frequency of 'J' : 3
Frequency of 'H' : 3
Frequency of 'D' : 2
Frequency of 'L' : 2
Frequency of 'M' : 1
Frequency of 'B' : 1
Frequency of 'S' : 1
Frequency of 'V' : 1
Found avi
```

2. Match_contact.pl

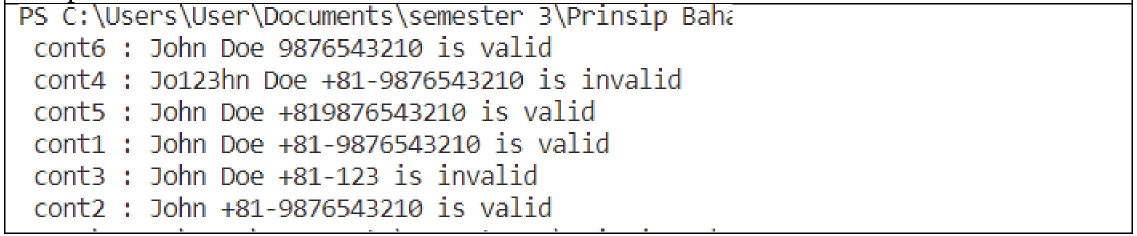
Source Code :

```perl
#!/usr/bin/perl -w
use strict;
#RegEx using DEFINE block
my $reg = qr/^(?&first_name)[\s]*(?&last_name)[\s]*(?&phone_number)$
    (?(DEFINE)
        (?<first_name> # first_name matches any number of alphabets
            ([a-zA-Z]+))
        (?<last_name>
            ([a-zA-Z]+)) # first_name matches any number of alphabets
        (?<phone_number>
            ((\+?(\d{1,3}))? # matches country code if present
            (\-)? # matches "-" between country code and number
            (\d{10}))) # matches 10 digit for a valid phone number
    )/xn; # used modifier x for free spacing

# Test contacts hash
```

```
my %contact = (
    "cont1" => "John Doe +81-9876543210",
    "cont2" => "John +81-9876543210",
    "cont3" => "John Doe +81-123",
    "cont4" => "Jo123hn Doe +81-9876543210",
    "cont5" => "John Doe +819876543210",
    "cont6" => "John Doe 9876543210",
    );
    # validating each contact
foreach my $key (keys %contact)
{
    print " $key : $contact{$key} is ";
    if ($contact{$key} =~ $reg)
    {
        print "valid \n";
        }
    else
    {
        print "invalid \n";
        }
    }
```

Output:



```
PS C:\Users\User\Documents\semester 3\Prinsip Baha
 cont6 : John Doe 9876543210 is valid
 cont4 : Jo123hn Doe +81-9876543210 is invalid
 cont5 : John Doe +819876543210 is valid
 cont1 : John Doe +81-9876543210 is valid
 cont3 : John Doe +81-123 is invalid
 cont2 : John +81-9876543210 is valid
```

3. Freq_of_chars.pl

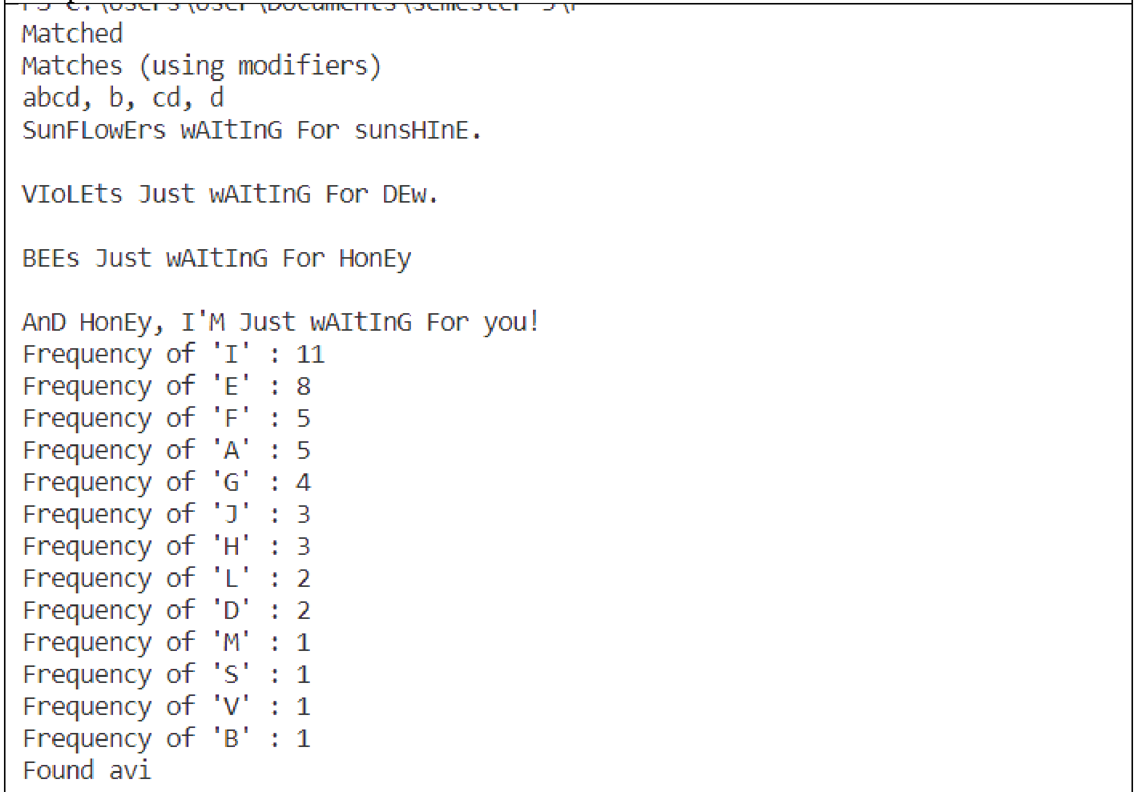
Source Code :

```
#!/usr/bin/perl -w
use strict;
my %chars;
my $Para =
"Sunflowers waiting for sunshine. \n
Violets just waiting for dew. \n
Bees just waiting for honey \n
And honey, I'm just waiting for you!";
print "Matched \n"
    if ($Para =~ m/for/);
print "Does not match \n"
    if ($Para =~ m/^And/);
```

```perl
print "Matches (using modifiers) \n"
    if ($Para =~ m/^And/m);

my $group = "abcd";
$group =~ m/(a(b|c)(c(d)))/;
print "$1, $2, $3, $4 \n";
$Para =~ tr/[a-m]/[A-M]/;
print "$Para \n";
$Para =~ s/([A-Z])/$chars{$1}++;$1/eg;
print "Frequency of '$_' : $chars{$_} \n"
    foreach (sort{$chars{$b} <=> $chars{$a}} keys %chars);
my $x = "tavi avi pavi a-avi";
print "Found avi"
    if ($x =~ /(?<=\s)avi(?=\s)/);
```

Output:



```
Matched
Matches (using modifiers)
abcd, b, cd, d
SunFLowErs wAItInG For sunsHInE.

VIoLEts Just wAItInG For DEw.

BEEs Just wAItInG For HonEy

AnD HonEy, I'M Just wAItInG For you!
Frequency of 'I' : 11
Frequency of 'E' : 8
Frequency of 'F' : 5
Frequency of 'A' : 5
Frequency of 'G' : 4
Frequency of 'J' : 3
Frequency of 'H' : 3
Frequency of 'L' : 2
Frequency of 'D' : 2
Frequency of 'M' : 1
Frequency of 'S' : 1
Frequency of 'V' : 1
Frequency of 'B' : 1
Found avi
```

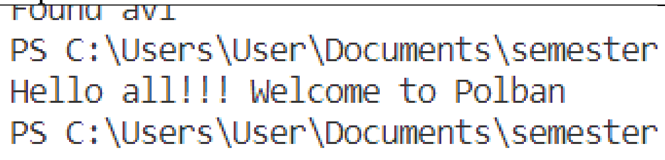
4. Substitution_regex.pl

Source Code :

```perl
#!/usr/bin/perl -w
# String in which text
$string = "Hello all!!! Welcome here";
$string =~ s/here/to Polban/;
```

```
print "$string\n";
```

Output:



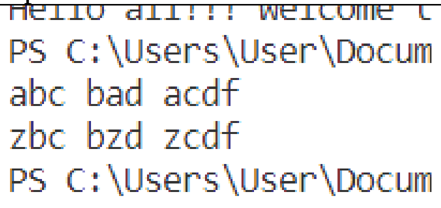
```
Found avi
PS C:\Users\User\Documents\semester
Hello all!!! Welcome to Polban
PS C:\Users\User\Documents\semester
```

5. Transliterate_01.pl

Source Code :

```
use strict;
use warnings;
use 5.010;
my $text = 'abc bad acdf';
say $text;
$text =~ tr/a/z/;
say $text;
```

Output:



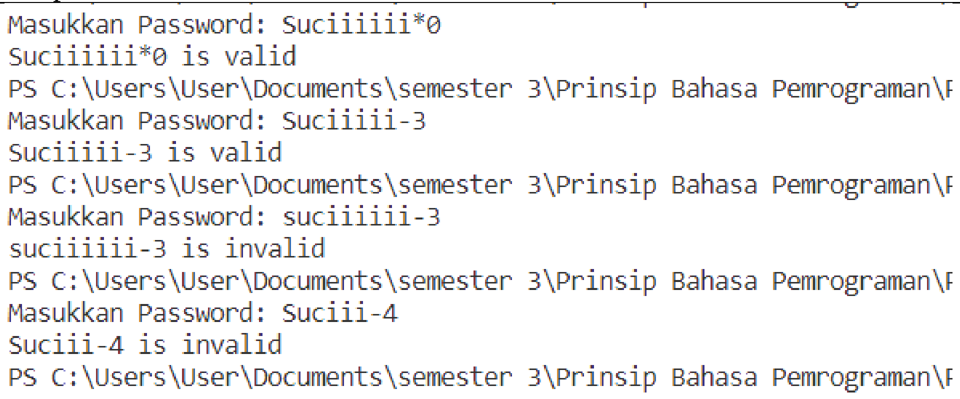
```
Hello all!!! Welcome t
PS C:\Users\User\Docum
abc bad acdf
zbc bzd zcdf
PS C:\Users\User\Docum
```

6. Password Validation

Source Code :

```
#!/usr/bin/perl -w
use strict;
print"Masukkan Password: ";
my $password = <STDIN>;
chomp $password;
print "$password is ";
if($password =~ m/^(?!.*[\s])(?=.*\d)(?=.*\W)(?=.*[a-z])(?=.*[A-
Z]).{10,}$/)
{
    print"valid\n";
}
else
{
    print"invalid\n";
}
```

Output:



```
Masukkan Password: Suciiiiii*0
Suciiiiii*0 is valid
PS C:\Users\User\Documents\semester 3\Prinsip Bahasa Pemrograman\P
Masukkan Password: Suciiiii-3
Suciiiii-3 is valid
PS C:\Users\User\Documents\semester 3\Prinsip Bahasa Pemrograman\P
Masukkan Password: suciiiiii-3
suciiiiii-3 is invalid
PS C:\Users\User\Documents\semester 3\Prinsip Bahasa Pemrograman\P
Masukkan Password: Suciii-4
Suciii-4 is invalid
PS C:\Users\User\Documents\semester 3\Prinsip Bahasa Pemrograman\P
```

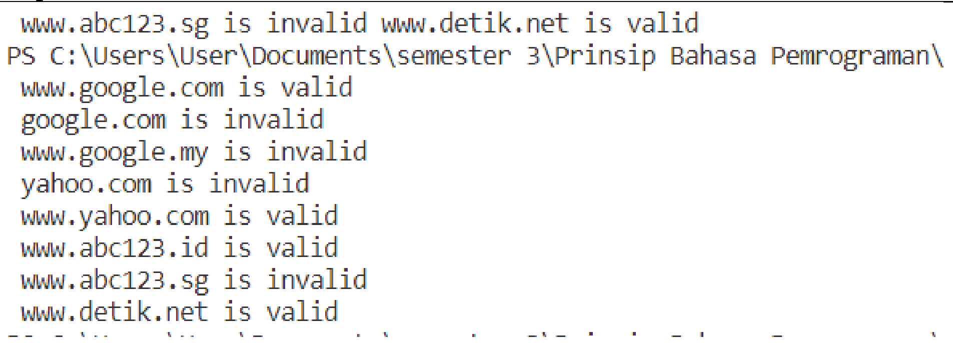
7. Domain Web Validation

Source Code :

```perl
#!/usrs/bin/perl/
my @domain = ("www.google.com", "google.com", "www.google.my",
"yahoo.com", "www.yahoo.com", "www.abc123.id", "www.abc123.sg",
"www.detik.net");

for (my $index=0; $index <= $#domain; $index++)
{
    if ($domain[$index] =~ qr/^www\.(.+[a-z0-9])\.(com|net|id)$/)
    {
        print" $domain[$index] is valid\n";
    }
    else
    {
        print" $domain[$index] is invalid\n";
    }
}
```

Output:



```
  www.abc123.sg is invalid www.detik.net is valid
 PS C:\Users\User\Documents\semester 3\Prinsip Bahasa Pemrograman\
  www.google.com is valid
  google.com is invalid
  www.google.my is invalid
  yahoo.com is invalid
  www.yahoo.com is valid
  www.abc123.id is valid
  www.abc123.sg is invalid
  www.detik.net is valid
```

8. Serial Number Matching

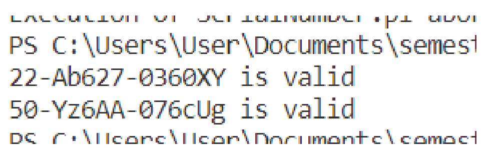
Source Code :

```perl
use strict;

my @number = ("22-Ab627-0360XY","50-Yz6AA-076cUg");

for(my $index=0; $index <= $#number; $index++)
{
    if($number[$index] =~ m/^([\d]{2})-([\w]{5})-([\w]{6})$/)
    {
        print"$number[$index] is valid\n";
    }
    else{
        print"$number[$index] is invalid\n";
    }
}
```

Output:



```
Execution of SerialNumber.pl abor
PS C:\Users\User\Documents\semest
22-Ab627-0360XY is valid
50-Yz6AA-076cUg is valid
PS C:\Users\User\Documents\semest
```