Problem Solving using c and c++

Lab Assignment 5

Patil Amit Gurusidhappa
19104004
B11

Q1. Considering the input telephone number format as (AA)-BBB-CCCCCCCC, where A, B, C belongs to a number set {0, 1, 2, ... 9}. Write a program to extract the ISD code, STD code and subscriber number using the input entered by user. Also display the details in the output.

```cpp
#include <iostream>
#include <regex>
using namespace std;

bool isValid(string s)
{

    const regex pattern("(0|91)?[7-9][0-9]{9}");

    if (regex_match(s, pattern))
    {
        return true;
    }
    else
    {
        return false;
    }
}

int main()
{
    string s = "347873923408";
    if (isValid(s))
    {
        cout << "Valid";
    }
    else
```

```
    {
        cout << "Invalid";
    }
    return 0;
}
```

```
PROBLEMS    OUTPUT    TERMINAL    ...

 ($?) { g++ q1.cpp -o q1 } ; if (
$?) { .\q1 }
Invalid
PS E:\Work\JIIT\sem_6\JIIT-SEM-6\
Problem_solving_lab\Lab5>
```

Q2. Write a program to search for file in the present working directory using regular expression. Also, list the
names of files matching the expression on the console. [You can assume the list of file in the directory as
abc.docx, a1,ppts, 123.pdf, def,docx, important.docx, test.docx, present.pptx]
The search operations should be performed based on the following issues,
a) Find all the files with same file type i.e., docx type.
b) Find all files that start with alphabets.
c) Find all the files that start with ab.
d) Find all files that end with numeric.

```
#include <string>
#include <iostream>
#include <filesystem>
namespace fs = std::filesystem;

int main()
{
    std::string path = "/path/to/directory";
    for (const auto &entry : fs::directory_iterator(path))
        std::cout << entry.path() << std::endl;
}
```

```
void PrintMatches2(string str, regex reg)
```

```
{
    sregex_iterator currentMatch(str.begin(), str.end(), reg);
    sregex_iterator lastMatch;
    while (currentMatch != lastMatch)
    {
        smatch match = *currentMatch;
        cout << match.str() << endl;
        currentMatch++;
    }
    cout << endl;
}
```

Q3. In an input paragraph, write a program to remove those words that has more than one occurrence of letter
'a' and has suffix as 'tion'.

```
#include <bits/stdc++.h>
using namespace std;

template <size_t N>
void splitString(string (&arr)[N], string str)
{
    int n = 0;
    istringstream iss(str);

    for(auto it = istream_iterator<string>(iss);
            it != istream_iterator<string>() && n < N;
        ++it, ++n)
        arr[n] = *it;
}

inline bool ends_with(std::string const& value,
                      std::string const& ending)
{
    if (ending.size() > value.size())
        return false;

    return std::equal(ending.rbegin(),
                      ending.rend(),
                      value.rbegin());
```

```
}



int endingWith(string str, string suff)
{


    int cnt = 0;
    const int size = 50;
    string words[size];


    splitString(words, str);


    for(int i = 0; i < size; i++)
    {


        if (ends_with(words[i], suff))
            cnt++;
    }
    return cnt;
}

int main()
{
    cout << endingWith(str, suff);
}
```

Also extract date field mentioned in the paragraph [mentioned as MM/DD/YYYY] and display the date in standard format as DD-MM-YYYY. There is possibility that only month and year in
mentioned in the paragraph too.

Q4. Write a program to take input as mathematical expression and extract the operands to perform the operation
mentioned in the expression. You must use regex to complete it. Consider only four mathematical binary
operators +, -, * and /.

```cpp
#include <iostream>
#include <regex>
#include <string>
using namespace std;

void PrintMatches2(string str, regex reg)
{
    sregex_iterator currentMatch(str.begin(), str.end(), reg);
    sregex_iterator lastMatch;
    while (currentMatch != lastMatch)
    {
        smatch match = *currentMatch;
        cout << match.str() << endl;
        currentMatch++;
    }
    cout << endl;
}

int main()
{
    string str = "2+&2";
    regex reg("(ape[^ ]?)");
    PrintMatches2(str, reg);

    string str1 = "2*3-9+1";
    regex reg1("(pick[^ ]+?)");
    PrintMatches2(str1, reg1);
}
```