Name of the Experiment

Write a program to determine the Union set, Concatenation set of two set of Regular Expression L and M and then find out the closure of L set.

Theory

A regular expression, regex or regexp (sometimes called a rational expression) is a sequence of characters that define a search pattern. Usually this pattern is used by string searching algorithms for "find" or "find and replace" operations on strings, or for input validation. It is a technique that developed in theoretical computer science and formal language theory. Regular expressions are used in search engines, search and replace dialogs of word processors and text editors, in text processing utilities such as sed and AWK and in lexical analysis. Many programming languages provide regex capabilities, built-in or via libraries. Regular Expression has some operators. These operations are:

1. The union of two languages Land M, denoted L u M, is the set of strings that are in either

L or M, or both. For example, if L= {001, 10,111} and M ={e,00}, then LUM={e,00,10,001,111}.

2. The concatenation of languages L and M is the set of strings that can be formed by taking any string in L and concatenating it with any string in M. For example ,

if L={001, 10,111}and M={e,001},then L.M is{001,10,111,001001,10001,111001}.

3. The closure of a language L is denoted L* and represents the set of those strings that can be formed by taking any number of strings from L, possibly with repetitions (i.e., the same string may be selected more than once) and concatenating all of them.

Code

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

void get_union(char *s1, char *s2)
{
   int m=0,n=0,k=0;
   char u[35];
   for(int i=0;i<strlen(s1);i++)
   {
      u[k]=s1[i];
```

```
        u[k+1]=' ';
        k=k+2;
    }
    for(int i=0;i<=strlen(s1);i++)
    {
        m=0;
        for(int j=0;j<strlen(s2);j++)
        {
            if(s2[i]==u[j])
            {
                m=1;
                break;
            }
        }
        if(m==0)
        {
            u[k]=s2[i];
            u[k+1]=' ';
            k=k+2;
        }
        if(i==(strlen(s1)))
            u[k]='\0';
    }
    cout<<"\nL+M = "<<"{ "<<u<<"}"<<endl;
}

void get_concat(char *s1,char *s2)
{
    char c[40];
    int k=0;
    for(int i=0 ; i<=strlen(s1) ; i++)
    {
        for(int j=0; j<strlen(s2) ; j++)
        {
            c[k] = s1[i];
            c[k+1] = s2[j];
            c[k+2] = ' ';
            k+=3;
        }
        if(i==(strlen(s1)))
        c[k]='\0';
    }
    cout<<"\nL.M = {"<<c<<"} "<<endl;
}

void get_closure(char *s)
```

```
{
   char res[1000],ress[1000],resss[1000],ressss[1000],result[1000];
   int c=0,k=4,l=0,m=0,n=0;
   res[0] = 'e';
   res[1] = 'p';
   res[2] = 's';
   res[3] = ' ';
   for(int i=0 ; i<=strlen(s) ; i++)
   {
      res[k] = s[i];
      res[k+1] = ' ';
      k+=2;
   }

   for(int i=0 ; i<=strlen(s) ; i++)
   {
      for(int j=0; j<strlen(s) ; j++)
      {
         ress[l] = s[i];
         ress[l+1] = s[j];
         ress[l+2] = ' ';
         l+=3;
      }
   }

   for(int i=0 ; i<=strlen(s) ; i++)
   {
      for(int j=0; j<strlen(s) ; j++)
      {
         for(int x=0 ; x<strlen(s) ; x++)
         {
            resss[m] = s[i];
            resss[m+1] = s[j];
            resss[m+2] = s[x];
            resss[m+3] = ' ';
            m+=4;
         }
      }
   }

   for(int i=0 ; i<=strlen(s) ; i++)
   {
      for(int j=0; j<strlen(s) ; j++)
      {
         for(int x=0 ; x<strlen(s) ; x++)
         {
```

```
            for(int y=0 ; y<strlen(s) ; y++)
            {
               ressss[n] = s[i];
               ressss[n+1] = s[j];
               ressss[n+2] = s[x];
               ressss[n+3] = s[y];
               ressss[n+4] = ' ';
               n+=5;
            }
         }
      }
   }

   strcpy(result,strcat(strcat(strcat(res,ress),resss),ressss));

   cout<<"{ "<<result<<"}"<<endl;
}

int main()
{
   int ch,ActLen1,ActLen2,c=0,d=0,op=1;
   char L[15],M[15];
   char str1[15],str2[15];

   cout<<"Enter the values of L set: ";
   gets(L);
   cout<<"Enter the values of M set: ";
   gets(M);

   ActLen1=((strlen(L)-2)/2)+1;
   ActLen2=((strlen(M)-2)/2)+1;

   for(int i=0,j=0;i<strlen(L);i++)
   {
      if(j==ActLen1){
         str1[j]='\0';
         break;
      }
      else if(L[i]!='{' && L[i]!='}' && L[i]!=','){
         str1[j]=L[i];
         j++;
      }
   }

   for(int i=0,j=0;i<strlen(M);i++)
   {
```

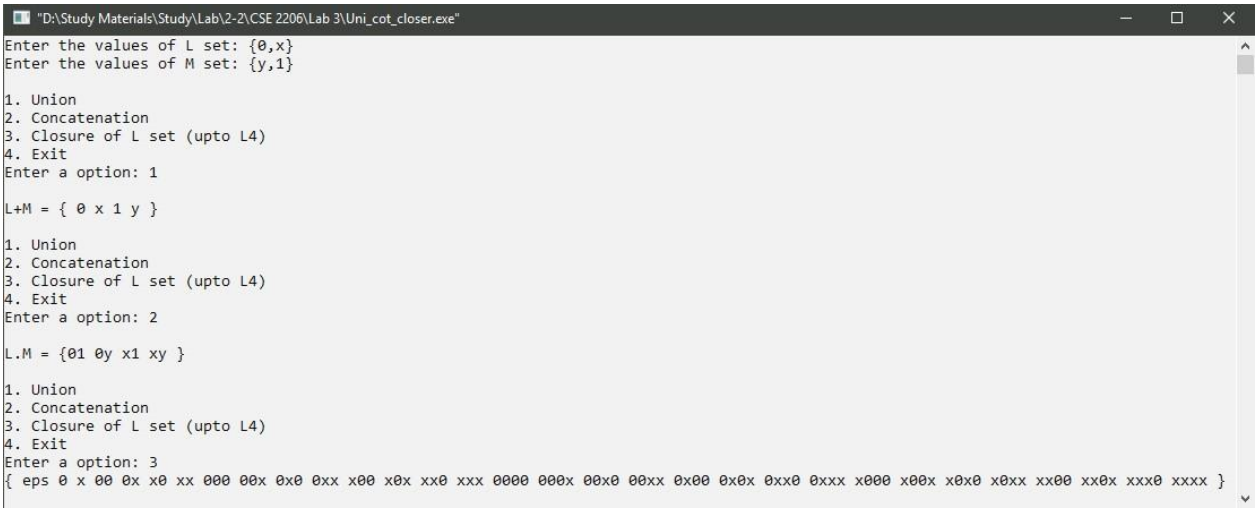
```
        if(j==ActLen2){
            str2[j]='\0';
            break;
        }
        else if(M[i]!='{' && M[i]!='}' && M[i]!=','){
            str2[j]=M[i];
            j++;
        }
    }

    sort(str1,str1+ActLen1);
    sort(str2,str2+ActLen2);

    while(op)
    {
        cout<<"\n1. Union\n2. Concatenation\n3. Closure of L set (upto L4)\n4. Exit\nEnter a
option: ";
        cin>>op;
        switch(op)
        {
            case 1:
                get_union(str1,str2);
                break;
            case 2:
                get_concat(str1,str2);
                break;
            case 3:
                get_closure(str1);
                break;
            case 4:
                cout<<"\nProgram Ended"<<endl;
                break;
            default:
                cout<<"Enter Valid option... Try Again !!!"<<endl;
        }
        if(op==4)
            break;
    }

    return 0;
}
```

Output



Discussion

Here , in this above code, a menu was created to perform the operations of union, concatenation and closure of L set. In the main function get_union(), get_concat() and get_closure() these three function was called which was used for performing union, concatenation and closure operations respectively. In those function the input was the string input of two sets. And then in the get_closure() function, closure set was determined upto L4.