concat ba

```
#include"iostream.h"
#include"stdio.h"
#include"string.h"
void main()
{
int i,j,k,n1,n2,n3;
char
a[10][10],b[10][10],c[10][10],r[10][10];
cout<<"input the size of a array=";
cin>>n1;
cout<<"input the size of b array=";
cin>>n2;
cout<<"input element of(a)array\n";
for(i=0;i<n1;i++)
gets(a[i]);
cout<<"input element of(b)array\n";
for(i=0;i<n2;i++)
gets(b[i]);
n3=n1*n2;
k=0;
for(i=0;i<n2;i++)
for(j=0;j<n1;j++)
{
strcpy(r[i],b[i]);
strcat(b[i],a[j]);
strcpy(c[k],b[i]);
strcpy(b[i],r[i]);
k++;
}
cout<<"the result array\n";
cout<<"c={";
for(i=0;i<n3;i++)
if(i<n3-1) cout<<(c[i])<<",";
else cout<<(c[i]);
cout<<"}";
}
```

Concat a and b

```
#include<iostream.h>
#include<string.h>
#include<stdio.h>
#include<conio.h>
int con(char a[10][20],char b[10][20],char c[20][40],int n1,int n2)
{int k=0,i,j,n3;
for(i=0;i<n1;i++)
for(j=0;j<n2;j++)
{strcpy(c[k],a[i]);
strcat(c[k],b[j]);
k++;}
n3=n1*n2;
return(n3);}
main()
{clrscr();
char a[10][20],b[10][20],c[20][40];
int i,m1,m2,m3;
cout<<"enter m1\n";
cin>>m1;
for(i=0;i<m1;i++)
```

```
gets(a[i]);
cout<<"enter m2 \n";
cin>>m2;
for(i=0;i<m2;i++)
gets(b[i]);
m3=con(a,b,c,m1,m2);
cout<<"c={"<<c[0];
for(i=1;i<m3;i++)
cout<<","<<c[i];
cout<<"}";
getch();
}
```

الابدالي Concat

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
#include<stdio.h>
void out(char a[20][40],int n1)
{int i;
cout<<a[0];
for(i=1;i<n1;i++)
{cout<<","<<a[i];}
cout<<"}";}
int cct(char a[20][40],char b[20][40],char c[20][40],int n1,int n2)
{int k=0,i,j,n3;
for(i=0;i<n1;i++)
for(j=0;j<n2;j++)
{strcpy(c[k],a[i]);strcat(c[k],b[j]);k++;}
n3=n1*n2; return(n3);}
main()
{clrscr();
char a[20][40],b[20][40],c[20][40],d [20][40];
```

```
int i,m1,m2,m3,m4,p=1;
cout<<"\n enter m1\n";cin>>m1;cout<<"enter elements\n";
for(i=0;i<m1;i++)
gets(a[i]);
cout<<"\n a={";out(a,m1);
cout<<"\n enter m2\n";cin>>m2;cout<<"enter elements \n";
for(i=0;i<m2;i++)
gets(b[i]);
cout<<"\n b={";out(b,m2);
m3=cct(a,b,c,m1,m2);
cout<<"\n c={";out(c,m3);
m4=cct(b,a,d,m2,m1);
cout<<"\n d={";out(d,m4);
for (i=0;i<m3;i++)
if(!strcmp(c[i],d[i]))
{p=0;break;}
if(p==0)
cout<<"\n equal";
else
cout<<"\n not equal";
getch();}
```

التجميعي concat

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
#include<stdio.h>
void out(char a[20][40],int n1)
{int i;
cout<<a[0];
for(i=1;i<n1;i++)
{cout<<","<<a[i];}
cout<<"}";}
int cct(char a[20][40],char b[20][40],char c[20][40],int n1,int n2)
{int k=0,i,j,n3;
for(i=0;i<n1;i++)
for(j=0;j<n2;j++)
{strcpy(c[k],a[i]);strcat(c[k],b[j]);k++;}
n3=n1*n2; return(n3);}
main()
{clrscr();
```

```
char
a[20][40],b[20][40],c[20][40],s1[20][40],s2[20][40],l1[20][40],l2[20][40]
;

int i,m1,m2,m3,m4,m5,m6,m7,p=1;

cout<<"\n enter m1\n";cin>>m1;cout<<"enter elements\n";

for(i=0;i<m1;i++)

gets(a[i]);

cout<<"\n a={";out(a,m1);

cout<<"\n enter m2\n";cin>>m2;cout<<"enter elements \n";

for(i=0;i<m2;i++)

gets(b[i]);

cout<<"\n b={";out(b,m2);

cout<<"\n enter m3 \n";cin>>m3;cout<<"enter elements \n";

for(i=0;i<m3;i++)

gets(c[i]);

cout<<"\n c={";out(c,m3);

m4=cct(a,b,s1,m1,m2);

cout<<"\n s1={";out(s1,m4);

m5=cct(s1,c,s2,m4,m3);

cout<<"\n s2={";out(s2,m5);

m6=cct(b,c,l1,m2,m3);

cout<<"\n l1={";out(l1,m6);
```

```
m7=cct(a,l1,l2,m1,m6);
cout<<"\n l2={";out(l2,m7);
for(i=0;i<m7;i++)
{if(!strcmp(s2[i],l2[i]))
p=0;}
if(p==0)
cout<<"\n equal";
else
cout<<"\n not equal";
getch();}
```

S*

```
#include<iostream.h>
#include<string.h>
#include<stdio.h>
#include<conio.h>
int con(char a[10][20],char b[10][20],char c[20][40],int n1,int n2)
{int k=0,i,j,n3;
for(i=0;i<n1;i++)
for(j=0;j<n2;j++)
{strcpy(c[k],a[i]);
strcat(c[k],b[j]);
k++;}
n3=n1*n2;
return(n3);}
main()
{clrscr();
char s[10][20],s1[10][20],s2[10][20],s3[10][20],sstar[20][40];
int i,n,n1,n2,n3,h=0;
cout<<"enter n\n";
cin>>n;
```

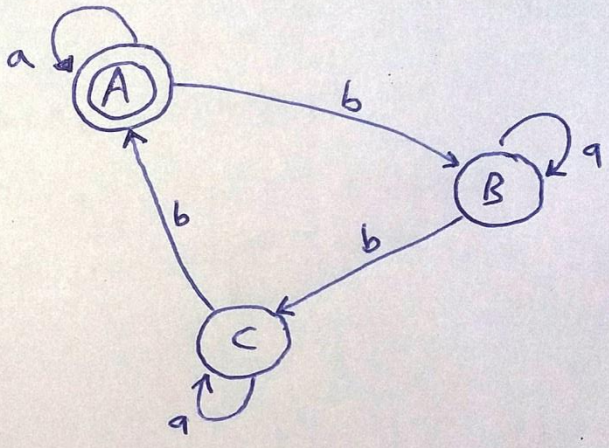
```
for(i=0;i<n;i++)
gets(s[i]);
n1=n;
for(i=0;i<n1;i++)
strcpy(s1[i],s[i]);
cout<<"s1={";
for(i=0;i<n1;i++)
{
cout<<s1[i];
if (i< n1-1)
cout<<",";
}
cout<<"}\n";
n3=con(s2,s,s3,n2,n);
cout<<"s3={";
for(i=0;i<n3;i++)
{
cout<<s3[i];
if(i<n3-1)
cout<<",";
}
```

```
cout<<"}\n";
strcpy(sstar[h],"null");
h++;
for(i=o;i<n1;i++)
{
strcpy(sstar[h],s1[i]);
h++;
}
for(i=o;i<n2;i++)
{
strcpy(sstar[h],s2[i]);
h++;
}
for(i=o;i<2;i++)
{
strcpy(sstar[i],s3[i]);
h++;
}
cout<<"sstar={";
for(i=0;i<h;i++)
{
```

```
cout<<sstar[h];
if(i<h-1)
cout<<"}";
}
```

FSA

```
#include<stdio.h>
#include<conio.h>
#include<iostream.h>
#include<string.h>
main()
{clrscr();
char a[3][2]={{'A','B'},{'B','C'},{'C','A'},s[20],k;
int i,j,l,n,f=0;
j=0;n=0;
gets(s);
l=strlen(s);
for( i=0;i<l;i++)
{if ((s[i]!='a')&&(s[i]!='b'))
{cout<<"The expression not accepted\n";f=1;break;}
else
if (s[i]=='a')
k=a[n][j];
else
if(s[i]=='b')
{k=a[n][j+1];n++;}
if(n==3)
n=0;
}
if(f==0)
if(k=='A')
cout<<"Accepted";
else
cout<<"not accepted";
 getch();
return 0;
}
```



Token

```
#include <stdio.h>
#include<stdlib.h>
#include<ctype.h>
#include<string.h>
#include<iostream.h>
#include<conio.h>
main()
{ char s[20];
clrscr();
cout<<"Assumption: the expression contains single letter
variables and single digit constants only.\n";
cout<<"\n enter expression:";
gets(s);
int l=strlen(s);
for (int i=0;i<l;i++)
{if (isdigit(s[i]))
cout<<"\n"<<s[i]<<" digit";
if (isalpha(s[i]))
cout<<"\n"<<s[i]<<" identifier";
if
(s[i]=='+'||s[i]=='-'||s[i]=='*'||s[i]=='/'||s[i]=='^'||s[i]=='
('||s[i]==')')
cout<<"\n"<<s[i]<<" operator";
if (s[i]=='=')
cout<<"\n"<<s[i]<<" assignment";
}
getchar();
}
```

Stack

```
#include<iostream.h>
#include<conio.h>
#include<stdio.h>
#include<string.h>
void push(char item,int &top,char st[30])
{
if (top<30)
{ top++;
st[top]=item;}
else
cout<<"stack is full";
}
void pop(char &item,int &top,char st[30])
{ if (top==-1)
cout<<"stack is empty\n";
else
{item=st[top];
top--;
}
}
main()
{char s[30],item,st1[30],st2[30];
int i,top=-1,top2=-1;
clrscr();
gets(s);
int l=strlen(s);
for (i=0;i<l;i++)
push(s[i],top,st1);
for (i=0;i<l;i++)
{
pop(item,top,st1);
if (item=='a')
push(item,top2,st2);
else
cout<<"\n item="<<item;
}
cout<<"\n";
for(i=top2;i>=0;i--)
cout<<st2[i];

getch();
}
```

infix - Prefix - Postfix

```
#include <stdio.h>
#include<stdlib.h>
#include<ctype.h>
#include<string.h>
#include<iostream.h>
#include<conio.h>
const int size=100;
char stack[size];
int top=-1;
  void push(char item)
  {
   if(top>=size-1)
   cout<<"\n stack overflow";
   else
   {
    top=top+1;
    stack[top]=item;
   }
  }
  char pop()
  {
   char item;
   if (top<0)
    {
     cout<<"stack under flow: invalid infix expression";
     getchar();
     exit(1);
    }
    else
    {
     item=stack[top];
     top=top-1;
     return(item);
    }
   }
   int is_operator(char symbol)
   {
    if (symbol=='^' || symbol=='*' || symbol=='/' ||
symbol=='+' || symbol=='-')
    return 1;
    else
    return 0;
```

```
}
int precedence(char symbol)
{
if(symbol=='^')
return (3);
else if (symbol=='*' || symbol=='/')
        return(2);
     else if (symbol=='+' || symbol=='-')
             return(1);
          else
           return(0);
}
void infixtopostfix(char infix_exp[],char postfix_exp[])
{
int i,j;char item,x;
push ('(');
strcat(infix_exp,")");
i=0;j=0;
item=infix_exp[i];
while(item !='\0')
{
if(item=='(')
push(item);
else if (isdigit(item)||isalpha(item))
      {postfix_exp[j]=item;
      j++;
      }
      else if(is_operator(item)==1)
           {
            x=pop();
            while(is_operator(x)==1 &&
precedence(x)>=precedence(item))
            {
            postfix_exp[j]=x;
            j++;
            x=pop();
            }
            push(x);
            push(item);
           }
           else if (item==')')
                 {x=pop();
```

```
                        while(x!='(')
                        {postfix_exp[j]=x;
                        j++;
                        x=pop();}
                       }
                       else{cout<<"\ninvlid infix expression.\n";
                            getchar();
                            exit(1);}
   i++;
   item=infix_exp[i];
   }
   if(top>0)
   {cout<<"\ninvalid infix expression.\n";
    getchar();
    exit(1);}
   postfix_exp[j]='\0';
  }
int main()
{   clrscr();
  char infix[size],postfix[size];
  cout<<"Assumption: the infix expression contains singl letter
variables and single digit constants only.\n";
  cout<<"\n ente infix expression:";
  gets(infix);
  infixtopostfix(infix,postfix);
  cout<<"postfix expression: " ;
  puts(postfix);
   getchar();
  return 0;

  }
```