CD PRACTICALS

Practical No. 1

```c
#include <stdio.h>
int main()
{
   char str[20],opp[10],id[10],numb[10];
   int i,j=0,k=0,l=0,n=5;
   printf("Enter the equation:\n");
   scanf("%s",str);
   for(i=0;i<=n;i++)
   {
       if((str[i]>='a' && str[i]<='z') || (str[i]>='A' && str[i]<='Z'))
       {
           id[j]=str[i];
           j++;

       }
        else if(str[i]>='0' && str[i]<='9')
        {
            numb[l]=str[i];
            l++;
        }
        else
          {
            opp[k]=str[i];
            k++;
        }
   }
   printf("\nOPERATOR:");
   for(i=0;i<k;i++)
   {
     printf("%c",opp[i]);
    printf(",");

   }
   printf("\nIDENTIFIER:");
   for(i=0;i<j;i++)
   {
       printf("%c",id[i]);
        printf(",");
   }
   printf("\nNUMBER:");
   for(i=0;i<l;i++)
   {
       printf("%c",numb[i]);
```

```
        printf(",");
    }

     return 0;
}
```

Practical No.2:

```
#include <stdio.h>
#include<string.h>
int main()
{
    char str[20];
    int l;
    printf("Enter the string");
    scanf("%s",str);
    l=strlen(str);
   // printf("%s",str);
    if(str[0]=='/' && str[1]=='/' && str[2]!='/')
    {
    printf("It is single line comment");

    }
  else if(str[l-2]=='*' && str[l-1]=='/' && str[0]=='/' && str[1]=='*')
  {
  printf("It is a multiline comment");
  }
  else
  printf("It is not comment");
    return 0;
}
```

Practical No.3:

```
#include <bits/stdc++.h>
using namespace std;
bool areBracketsBalanced(string expr)
{
    stack <char> s;
    char x;
    for(int i=0;i<expr.length();i++)
{
    if(expr[i]=='(' || expr[i]=='[' || expr[i]=='{')
```

```
    {
        s.push(expr[i]);
        continue;
    }
    if(s.empty())
    return false;
    switch(expr[i])
    {
    case ')':
        x=s.top();
        s.pop();
        if(x=='{' || x=='[')
        return false;
        break;
    case '}':

        x=s.top();
        s.pop();
        if(x=='(' || x=='[')
        return false;
        break;
    case ']':
        x=s.top();
        s.pop();
        if(x=='(' || x=='{')
        return false;
        break;
    }
}
    return(s.empty());
}
int main()
{
    string expr="{()}[]";
    if(areBracketsBalanced(expr))
    cout<<"Balanced";
    else
    cout<<"Not Balanced";
    return 0;
}
```

Practical No.4:

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

```
class Trans
{
    public:
    char ip_state,op_state,ip_symbol;
    Trans()
    {
    ip_state=op_state=ip_symbol='\0';
    }
};
int main()
{
    int nos,ips,notr,i;
    char state[10],ipsmb[5],str[20],istate,fstate,extra;
    Trans tr[20];
    cout<<"Enter no of states";
    cin>>nos;
    for(i=0;i<nos;i++)
    {
    cout<<"Enter state:"<<i+1<<":";
    cin>>state[i];
    }
    cout<<"Enter initial state";
    cin>>istate;
    cout<<"Enter final state";
    cin>>fstate;
    cout<<"Enter no of input symbol";
    cin>>ips;
    for(i=0;i<ips;i++)
    {
        cout<<"Enter ip symbol"<<i+1<<":";
        cin>>ipsmb;
    }
    cout<<"Enter no of transitions";
    cin>>notr;
    for(i=0;i<notr;i++)
    {
        cout<<"Enter transitions"<<i+1<<":";
        cin>>tr[i].ip_state,tr[i].ip_symbol,tr[i].op_state;
    }
        for(i=0;i<notr;i++)
            cout<<tr[i].ip_state<<"->"<<tr[i].ip_symbol<<"-
>"<<tr[i].op_state<<"\n";
            extra=istate;
            cout<<"Enter string";
            cin>>str;
            int k=0;
             for(i=0;i<notr;i++)
            {
```

```
            if((extra==tr[i].ip_state) && (str[k]==tr[i].ip_symbol))
             {
                 extra=tr[i].op_state;
                 k++;
             }
            }
            if(extra==fstate)
            cout<<"String is accepted";
            else
            cout<<"String is not accepted";
             return 0;
}
```

Practical No.5:

```
#include<iostream>
#include<stdlib.h>
using namespace std;
struct symbol
{
    char alpha;
    int value;
};
void display(symbol tab[],int n);
void modify(symbol tab[],int n);
void search(symbol tab[],int n);
void Delete(symbol tab[],int n);
 int main()
 {
     symbol *table;
     int n,i;
     char alpha;
     cout<<"How many Symbol You Have to Enter:";
     cin>>n;
     table=new symbol[n];
     cout<<"Symbol\tValue\n";
     for(i=0;i<n;i++)
     {
         cin>>table[i].alpha>>table[i].value;
     }
     while(1)
     {
         int choice;
         cout<<"1.Display"<<endl;
         cout<<"2.Modify"<<endl;
         cout<<"3.Search"<<endl;
```

```
        cout<<"4.Delete"<<endl;
        cout<<"Enter your choice"<<endl;
        cin>>choice;
        switch(choice)
        {
            case 1:
            display(table,n);
            break;
             case 2:
            modify(table,n);
            break;
             case 3:
            search(table,n);
            break;
             case 4:
            Delete(table,n);
            break;
            default:
            exit(0);
        }
        }
        return 0;
}
void display(symbol tab[],int n)
{
    int i;
    for(i=0;i<n;i++)
   {
       cout<<tab[i].alpha<<"\t"<<tab[i].value<<endl;
   }
}
void modify(symbol tab[],int n)
{
    int temp,i;
    char alpha;
    cout<<"Enter the symbol to be modified";
    cin>>alpha;
    cout<<"Enter the value after modification";
    cin>>temp;
    for(i=0;i<n;i++)
    {
        if(tab[i].alpha==alpha)
        {
            tab[i].value=temp;
            cout<<endl;
            display(tab,n);
        }
    }
```

```
}
void search(symbol tab[],int n)
{
    int i;
    char alpha;
    cout<<"Enter the symbol to be searched";
    cin>>alpha;
    for(i=0;i<n;i++)
    {
        if(tab[i].alpha==alpha)
        {
            cout<<tab[i].alpha<<"\t"<<tab[i].value<<endl;
        }

    }
}
void Delete(symbol tab[],int n)
{
    int i;
    char alpha;
    cout<<"Enter the Symbol to be delete";
    cin>>alpha;
    for(i=0;i<n;i++)
    {
        if(tab[i].alpha==alpha)
        {
           tab[i].alpha=tab[i+1].alpha;
           tab[i].value=tab[i+1].value;
        }
    }
    n=n-1;
    display(tab,n);
}
```

Practical No.6:

```
#include <iostream>
#include<string.h>
#define MAX 20
#include<ctype.h>
#include<stdlib.h>
using namespace std;
int i;
char s[MAX];
int E();
int getint()
{
```

```
    int j=0;
    char w[MAX];
    if(!isdigit(s[i]))
    return(-1);
    while(isdigit(s[i]))
    w[j++]=s[i++];
    w[j]='\0';
    return(atoi(w));
}
int F()
{
    int v;
    while(s[i]=='(')
    {
        i++;
        if(i==strlen(s))
        v=E();
        if(s[i]!=')')
        return(-1);
        else{
            i++;
            return(v);
        }
    }
    v=getint();
    return(v);
}

int T()
{
    int v;
    v=F();
    while(s[i]=='*')
    {
        i++;
        if(i==strlen(s))
        return(-1);
         v=v*F();
    }

    return(v);
}
int E()
{
    int v;
    v=T();
    while(s[i]=='+')
    {
```

```
        i++;
        if(i==strlen(s))
        return(-1);
         v=v+T();
    }

    return(v);
}
int main()
{
    int ans;
    cout<<"Grammar:\nE->E+T|T\nT->T*F|F\nF->(E)|0|1|.....|9\n";
    cout<<"Enter the expression:";
    cin>>s;
    ans=E();
    if(ans==-1)
    cout<<"\nError in passing the expression";
    else
    cout<<"Result is "<<ans;
    return 0;
}
```

Practical No.7:

```
#include <iostream>
using namespace std;
int main()
{
    char str[20];
    char stack[20];
    int top=0;
    cout<<"Given grammar is:\nS->aABe\nA->Abc\nB->d\n"<<endl;
    cout<<"Enter the String:"<<endl;
    cin>>str;
    cout<<"Stack\t\tinput buffer\t\tAction"<<endl;
    cout<<"_____"<<endl;
    stack[top]='$';
    top++;
    cout<<"$\t\tabbcde\t\tShift"<<endl;
    stack[top]=str[0];
    top++;
    cout<<"$a\t\tbbcde$\t\tShift"<<endl;
    stack[top]=str[1];
    if(stack[top]='b')
```

```
    {
    cout<<"$ab\t\tbcde$\t\tReduce"<<endl;
    stack[top]='A';
    top++;
    cout<<"$aA\t\tbcde$\t\tShift"<<endl;
    stack[top]='b';
    top++;
    cout<<"$aAb\t\tcde$\t\tShift"<<endl;
    stack[top]='c';
    top++;
    cout<<"$aAc\t\tcde$\t\tReduce"<<endl;
    }
    if(stack[top]=='b' && stack[top-1]=='c')
    {
        top=top-2;
        cout<<"$aA\t\tde$\t\tShift"<<endl;
        top++;
        stack[top]='d';
        cout<<"$aAd\t\te$\t\tShift"<<endl;
    }
    if(stack[top]='d')
    {
        stack[top]='B';
        cout<<"$aAB\t\te$\t\tShift"<<endl;
        top++;
        stack[top]='e';
        cout<<"$aABe\t\t$\t\tReduce"<<endl;
        top=top-4;
        stack[top]='S';
        cout<<"$S\t\t$\t\tACCEPT"<<endl;
    }
    if(stack[top]=='S')
    {
        cout<<"String is valid"<<endl;
    }
    return 0;
}
```

Practical No.8:

```
#include <iostream>
#include<string.h>
using namespace std;
int dfa=0;
void start (char c)
{
```

```
    if(c=='a'){
    dfa=1;
    }
    else if( c=='b'){
        dfa=3;
    }
    else {
    dfa=-1;
    }
}
void state1(char c)
{
    if( c=='a'){
    dfa=2;
    }
    else if( c=='b'){
        dfa=4;
    }
    else {
    dfa=-1;
    }
}
void state2(char c)
{
    if( c=='b'){
    dfa=3;
    }
    else if( c=='a'){
        dfa=1;
    }
    else {
    dfa=-1;
    }
}
void state3(char c)
{
    if(c=='b'){
    dfa=3;
    }
    else if( c=='a'){
        dfa=4;
    }
    else {
    dfa=-1;
    }
}
void state4(char c)
{
```

```
    dfa=-1;
}
int isAccepted(char str[])
{
    int i,len=strlen(str);
    for(i=0;i<len;i++)
    {
        if(dfa==0)
        start(str[i]);
        else if(dfa==1)
         state1(str[i]);
        else if(dfa==2)
         state2(str[i]);
       else if(dfa==3)
         state3(str[i]);
     else if(dfa==4)
         state4(str[i]);
         else
         return 0;
    }
    if(dfa==3)
    return 1;
    else
    return 0;
}
int main()
{
    char str[20];
    cout<<"Enter the string:";
    cin>>str;
    if(isAccepted(str))
    cout<<"String is ACCEPTED";
   else
   cout<<"Not ACCEPTED";
    return 0;
}
```

Practical No.10:

```
#include<stdio.h>
#include<string.h>
//#include<conio.h>
void pm();
void plus();
void div();
int i,ch,j,l,addr=100;
```

```
char ex[10],abc[10],exp1[10],xyz[10],id1[5],op[5],id2[5];
int main()
{
//  clrscr();
    while(1)
    {
        printf("\n1.assignment\n2.arithmetic\n3.relational\n4.Exit\nEnter the
choice:");
        scanf("%d",&ch);
        switch(ch)
        {
        case 1:
            printf("\nEnter the expression with assignment operator:");
            scanf("%s",&abc);
            l=strlen(abc);
            xyz[0]='\0';
            i=0;
            while(abc[i]!='=')
            {
                i++;
            }
            strncat(xyz,abc,i);
            strrev(abc);
            exp1[0]='\0';
            strncat(exp1,abc,l-(i+1));
            strrev(exp1);
            printf("\nThree address code:\ntemp=%s\n%s=temp\n",exp1,xyz);
            break;

        case 2:
            printf("\nEnter the expression with arithmetic operator:");
            scanf("%s",&ex);
            strcpy(abc,ex);
            l=strlen(abc);
            exp1[0]='\0';
            for(i=0;i<l;i++)
            {
                if(abc[i]=='+'||abc[i]=='-')
                {
                    if(abc[i+2]=='/'||abc[i+2]=='*')
                    {
                        pm();
                        break;
                    }
                    else
                    {
                        plus();
                        break;
```

```
                    }
                }
                else if(abc[i]=='/'||abc[i]=='*')
                {
                    div();
                    break;
                }
            }
            break;

        case 3:
            printf("Enter the expression with relational operator:");
            scanf("%s%s%s",&id1,&op,&id2);
            if(((strcmp(op,"<")==0)||(strcmp(op,">")==0)||(strcmp(op,"<=")==0)
||(strcmp(op,">=")==0)||(strcmp(op,"==")==0)||(strcmp(op,"!=")==0))==0)
            printf("Expression is error");
            else
            {
                printf("\n%d\tif %s%s%s goto %d",addr,id1,op,id2,addr+3);
                addr++;
                printf("\n%d\t T:=0",addr);
                addr++;
                printf("\n%d\t goto %d",addr,addr+2);
                addr++;
                printf("\n%d\t T:=1",addr);
            }
            break;
        case 4:
            //exit (0);
            break;
        }
    }
return 0;
}
void pm()
{
    strrev(abc);
    j=l-i-1;
    strncat(exp1,abc,j);
    strrev(exp1);
    printf("Three address
code:\ntemp=%s\ntemp1=%c%ctemp\n",exp1,abc[j+1],abc[j]);
}
void div()
{
    strncat(exp1,abc,i+2);
    printf("Three address
code:\ntemp=%s\ntemp1=temp%c%c\n",exp1,abc[i+2],abc[i+3]);
```

```
}
void plus()
{
    strncat(exp1,abc,i+2);
    printf("Three address
code:\ntemp=%s\ntemp1=temp%c%c\n",exp1,abc[i+2],abc[i+3]);
}
```