**CD Practical 9:**

**Write a C program for implementing the functionalities of predictive parser for the mini language.**

#include<string.h>

#include<conio.h>

char a[10];

int top=-1,i;

void error(){

printf("Syntax Error");

}

void push(char k[]) //Pushes The Set Of Characters on to the Stack

{

for(i=0;k[i]!='\0';i++)

{

if(top<9)

a[++top]=k[i];

} }

char TOS() //Returns TOP of the Stack

{

return a[top];

}

void pop() //Pops 1 element from the Stack

{

if(top>=0)

a[top--]='\0';

}

void display() //Displays Elements Of Stack

{

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

printf("%c",a[i]);

}

void display1(char p[],int m) //Displays The Present Input String

{

int l;

printf("\t");

for(l=m;p[l]!='\0';l++)

printf("%c",p[l]);

}

char\* stack(){

return a;

}

void main()

{

charip[20],r[20],st,an;

intir,ic,j=0,k;

char t[5][6][10]={"$","$","TH","$","TH","$",

"+TH","$","e","e","$","e",

"$","$","FU","$","FU","$",

"e","\*FU","e","e","$","e",

"$","$","(E)","$","i","$"};

clrscr();

printf("\nEnter any String(Append with $)");

gets(ip);

printf("Stack\tInput\tOutput\n\n");

push("$E");

display();

printf("\t%s\n",ip);

for(j=0;ip[j]!='\0';)

{

if(TOS()==an)

{

pop();

display();

display1(ip,j+1);

printf("\tPOP\n");

j++;

}

an=ip[j];

st=TOS();

if(st=='E')ir=0;

else if(st=='H')ir=1;

else if(st=='T')ir=2;

else if(st=='U')ir=3;

else if(st=='F')ir=4;

else {

error();

break;

}

if(an=='+')ic=0;

else if(an=='\*')ic=1;

else if(an=='(')ic=2;

else if(an==')')ic=3;

else if((an>='a'&&an<='z')||(an>='A'&&an<='Z')){ic=4;an='i';}

else if(an=='$')ic=5;

strcpy(r,strrev(t[ir][ic]));

strrev(t[ir][ic]);

pop();

push(r);

if(TOS()=='e')

{

pop();

display();

display1(ip,j);

printf("\t%c->%c\n",st,238);

}

else{

display();

display1(ip,j);

printf("\t%c->%s\n",st,t[ir][ic]);

}

if(TOS()=='$'&&an=='$')

break;

if(TOS()=='$'){

error();

break;

} }

k=strcmp(stack(),"$");

if(k==0 && i==strlen(ip))

printf("\n Given String is not accepted");

else

printf("\n Given String is accepted");

getch();

}

**Program output:**

