Design, develop and implement YACC/C program to construct *Predictive / LL(1) Parsing Table* for the grammar rules: *A* *aBa , B* *bB |* . Use this table to parse the

sentence: *abba$.*

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

char prod[3][15]={"A->aBa","B->bB","B->@"};

char table[2][3][3]={{"aBa"," "," "},

{"@","bB"," "}};

int size[2][3]={3,0,0,1,2,0},n;

char s[20],stack[20];

void display(int i,int j)

{

int k;

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

printf("%c",stack[k]);

printf(" ");

for(k=j;k<n;k++)

printf("%c",s[k]);

printf("\n");

}

void main()

{

int i,j,k,row,col,flag=0;

printf("\nThe grammar is:\n");

for(i=0;i<3;i++)

printf("%s\n",prod[i]);

printf("\nPredictive parsing table is:\n");

printf("\ta \tb \t$ \n");

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

for(i=0;i<2;i++)

{

if(i==0)

printf("A");

else

printf("B");

for(j=0;j<3;j++)

{

printf("\t%s",table[i][j]);

}

}

printf("\nEnter the string:");

scanf("%s",s);

strcat(s,"$");

n=strlen(s);

stack[0]='$';

stack[1]='A';

i=1;

j=0;

printf("\nStack input");

printf("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

while(1)

{

if(stack[i]==s[j])

{

i--;

j++;

if(stack[i]=='$' && s[j]=='$')

{

printf("$$\nSuccess\n");

break;

}

else

if(stack[i]=='$' && s[j]!='$')

{

printf("Error\n");

break;

}

display(i,j);

}

switch(stack[i])

{

case 'A':row=0;break;

case 'B':row=1;break;

}

switch(s[j])

{

case 'a':col=0;break;

case 'b':col=1;break;

case '$':col=2;break;

}

if(table[row][col][0]=='\0')

{

printf("\nError\n");

break;

}

else if(table[row][col][0]=='@')

{

i--;

display(i,j);

}

else

{

for(k=size[row][col]-1;k>=0;k--)

{

stack[i]=table[row][col][k];

i++;

}

i--;

display(i,j);

}

}

}