NAME: MEET SHAH

USN: 1PE17CS084

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$.

PREDICTIVE PARSING
PROCEDURE

1. In the beginning, the pushdown stack holds the start symbol of the grammar G. 2. At each step a symbol X is popped from the stack:
if X is a terminal then it is matched with the lookahead and lookahead is advanced one step, if X is a nonterminal symbol, then using lookahead and a parsing table (implementing the FIRST sets) a production is chosen and its right-hand side is pushed into the stack. 3. This process repeats until the stack and the input string become null (empty) .

PROGRAM CODE:

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

char prod [3][10]={"A->aBa","B->bB","B->@"};
char first[3][10]={"a","b","@"};
char follow[3][10]={"$","a","a"};
char table[3][4][10];

char input[10];
int top=-1;
char stack[25];
char curp[20];

void push(char item)
{
  stack[++top]=item;
}
void pop()
```

USN: 1PE17CS084

```
{
 top=top-1;
}
void display()
{
 int i;
 for(i=top;i>=0;i--)
 printf("%c",stack[i]);
}

int numr(char c)
{
 switch(c)
 {
   case'A':return 1;
   case'B':return 2;
   case'a':return 1;
   case'b':return 2;
   case'@':return 3;
 }
 return 1;
}

int main()
{
 char c;
 int i,j,k,n;
 for(i=0;i<3;i++){
   for(j=0;j<4;j++){
     strcpy(table[i][j],"EMPTY");
   }
 }
 printf("\nGrammar\n");

 for(i=0;i<3;i++)
 printf("%s\n",prod[i]);
```

USN: 1PE17CS084

```
   printf("\nfirst={%s,%s,%s}",first[0],first[1],first[2]);
    printf("\nfollow={%s,%s}\n",follow[0],follow[1]);
    printf("\nPredictive parsing table for the given grammar :\n");

    strcpy(table[0][0],"");
    strcpy(table[0][1],"a");
    strcpy(table[0][2],"b");
    strcpy(table[0][3],"$");
    strcpy(table[1][0],"A");
    strcpy(table[2][0],"B");

    for(i=0;i<3;i++)
    {
      if(first[i][0]!='@')
      strcpy(table[numr(prod[i][0])][numr(first[i][0])],prod[i]);
      else
      strcpy(table[numr(prod[i][0])][numr(follow[i][0])],prod[i]);
    }
    printf("\n----------------------------------------------------------------\n");
    for(i=0;i<3;i++){
      for(j=0;j<4;j++)
      {
        printf("%-30s",table[i][j]);
        if(j==3) printf("\n----------------------------------------------------------------\n");
      }
    }

    printf("Enter the input string terminated with $ to parse:-");
    scanf("%s",input);
    for(i=0;input[i]!='\0';i++){
      if((input[i]!='a')&&(input[i]!='b')&&(input[i]!='$'))
      {
        printf("Invalid String");
        exit(0);
      }
    }
```

USN: 1PE17CS084

```
  if(input[i-1]!='$')
   {
    printf("\n\nInput String Entered Without End Marker $");
    exit(0);
   }

          push('$');
   push('A');
   i=0;

          printf("\n\n");
   printf("Stack\t Input\tAction");
   printf("\n---------------------------------------------------------------------\n");

          while(input[i]!='$'&&stack[top]!='$')
   {
    display();
    printf("\t\t%s\t",(input+i));
    if(stack[top]==input[i])
    {
     printf("\tMatched %c\n", input[i]);
     pop();
     i++;
    }
    else
    {
     if(stack[top]>=65&&stack[top]<92)
     {
      strcpy(curp,table[numr(stack[top])][numr(input[i])]);
      if(!(strcmp(curp,"e")))
      {
       printf("\nInvalid String - Rejected\n");
       exit(0);
      }
      else
      {
```

USN: 1PE17CS084

```
   printf("\tApply production %s\n",curp);
          if(curp[3]=='@')
          pop();
          else
          {
            pop();
            n=strlen(curp);
            for(j=n-1;j>=3;j--)
            push(curp[j]);
          }
        }
       }
     }

   display();
   printf("\t\t%s\t",(input+i));
   printf("\n---------------------------------------------------------------\n");
   if(stack[top]=='$'&&input[i]=='$')
   {
     printf("\nValid String - Accepted\n");
   }
   else
   {
     printf("Invalid String - Rejected\n");
   }
  }
```

USN: 1PE17CS084

OUTPUT: