|  |
| --- |
| /\*Description: Conversion of infix expression to postfix expression |
|  | \* Learner: ARSHEE QURESHI |
|  | \* created on: 20th JULY 2017 |
|  | \*/ |
|  |  |
|  | #include<stdio.h> |
|  | #include<ctype.h> //for isalpha() function |
|  |  |
|  | typedef struct conversion |
|  | { |
|  | char a[30]; |
|  | int top; |
|  | }stack; |
|  |  |
|  | void push(stack\*,char); |
|  | char pop(stack\*); |
|  | void convert(char[],char[]); |
|  | int priority(char); |
|  |  |
|  | int main() |
|  | { |
|  | char infix[30],postfix[30]; |
|  | printf("\nEnter INFIX Expression\n"); |
|  | scanf("%s",infix); |
|  | convert(infix,postfix); |
|  | printf("\n\nPostfix Expression\n%s",postfix); |
|  | return 0; |
|  | } |
|  | void convert(char in[],char post[]) |
|  | { //This function is used to convert infix expression to postfix |
|  | char opr; |
|  | stack s1; |
|  | int j=0,i; |
|  | s1.top=-1; |
|  | for(i=0;in[i]!='\0';i++) |
|  | { |
|  | if(isalpha(in[i])) |
|  | post[j++]=in[i]; |
|  | if(in[i]=='(') |
|  | push(&s1,in[i]); |
|  | if(in[i]=='+'||in[i]=='-'||in[i]=='/'||in[i]=='\*') |
|  | { |
|  | if(s1.top!=-1) |
|  | { |
|  | opr=pop(&s1); |
|  | while(priority(opr)>=priority(in[i])) |
|  | { |
|  | post[j++]=opr; |
|  | opr=pop(&s1); |
|  | } |
|  | push(&s1,opr); |
|  | push(&s1,in[i]); |
|  | } |
|  | else |
|  | push(&s1,in[i]); |
|  | } |
|  | if(in[i]==')') |
|  | { |
|  | opr=pop(&s1); |
|  | while(opr!='(') |
|  | { |
|  | post[j++]=opr; |
|  | opr=pop(&s1); |
|  | } |
|  | } |
|  | } |
|  | while(s1.top!=-1) |
|  | post[j++]=pop(&s1); |
|  | post[j]='\0'; |
|  | } |
|  |  |
|  | int priority(char c) |
|  | { //This function checks the priority of different symbols |
|  | if(c=='$') |
|  | return 3; |
|  | if(c=='\*'||c=='/') |
|  | return 2; |
|  | if(c=='+'||c=='-') |
|  | return 1; |
|  | else |
|  | return 0; |
|  | } |
|  |  |
|  | void push(stack \*s,char opr) |
|  | { //inserts an element into stack |
|  | s->top++; |
|  | s->a[s->top]=opr;//increment the top and set top of stack equal to opr |
|  | } |
|  |  |
|  | char pop(stack \*s) |
|  | { |
|  |  |
|  | //deletes an element from the top of stack |
|  | if (s->top==-1) |
|  | { |
|  | printf("Stack is empty\n"); |
|  |  |
|  | //message stack uderflow |
|  | return 0; |
|  | } |
|  | else |
|  | { |
|  | char data=s->a[s->top--];//set top of stack equal to data and decrement the top |
|  | return data; |
|  |  |
|  | } |
|  | } |
|  | /\* |
|  | Enter INFIX Expression |
|  | a+(b\*c-d/f)\*e |
|  |  |
|  |  |
|  | Postfix Expression |
|  | abc\*df/-e\*+ |
|  |  |
|  | ------------------ |
|  | (program exited with code: 0) |
|  | Press return to continue |
|  |  |
|  | \*/ |