Experiment Name / No.: 3

Camlin | Page No. 9

Date / /

IMPLEMENTATION OF STACK

AIM

write a menu driven program to implement Stack using 1-D array.

ALGORITHM

1 Start

2 Display the stack menu and ask the user to select Push, pop, peek or display

3 If user selects Push.

3.1 Ask the user to enter the element he wants to push

3.2 Call the function push() with the element as parameter.

4 If user selects pop

4.1 call the pop() function

5 If user selects peek

5.1 call the peek() function

6 If user selects display

6.1 call the display() function

7 Stop

→ push() function

1. start

2 if top == SIZE - 1

2.1 print stack is full and exit

Teacher's Signature:

3 else :
3.1 increment top by 1
3.2 stack [top] = x
5 3.3 Display message showing push successful
4 Stop

→ POP() function
1 Start
2 10 If top = -1
2.1 Display underflow !! stack empty
3. Else
3.1 Display Stack (top) as poped element.
3.2 top = top - 1
4 15 Stop

→ Peek() function
1. Start
2 If top = -1
20 2.1. Display underflow !! Stack empty
3. else
3.1 Display stack (top) as element at top
4. Stop

→ 25 display() function
1 Start

Teacher's Signature: _____

Experiment Name / No.:

Camlin Page No. 11
Date / /

2 if top = -1

2.1 print underflow !! stack empty

3 ~~if~~ else

3.1 initialise i to top

3.2 Repeat while i > -1

3.2.1 print stack[i]

3.2.2 decrement i by 1

4 STOP

CONCLUSION

The program has been executed correctly and output has been verified.

Teacher's Signature:

Experiment Name / No.: 4

Camlin / Page No. 12

Date / /

POSTFIX CONVERSION & EVALUATION

AIM

write a program to convert infix expression into postfix and evaluate the postfix expression.

ALGORITHM

1 Start

2 make the user select between conversion & evaluation of postfix expression.

3 If conversion:

3.1 input the infix expression from the user.

3.2 Call the InfixToPostfix() function and store the postfix expression into postfix

3.3 Display postfix as postfix expression

4 If evaluation:

4.1 call the evaluate() function.

5 Stop

→ InfixToPostfix() function

1. Start
2. push '(' into the stack using push()
3. append ')' at end of the infix expression
4. initialise i to 0, j to 0 and item to infix exp[i]
5. while item != '\0'

Teacher's Signature:

5.1 If item = '('

5.1.1 push(item)

5.2 If item is digit or a letter

5.2.1 postfix_exp[j] = item

5.2.2 j = j + 1

5.3 If item is_operator()

5.3.1 x = POP()

5.3.2 while (is_operator(x) == 1 && precedence(x) >= Precedence(item))

5.3.2.1 postfix_exp[j] = x

5.3.2.2 j = j + 1

5.3.2.3 x = pop()

5.3.3 push(x)

5.3.4 push(item)

5.4 If item = ')'

5.4.1 x = POP()

5.4.2 while x != '('

5.4.2.1 postfix_exp[j] = x

5.4.2.2 j = j + 1

5.4.2.3 x = POP()

5.5 else

5.5.1 print invalid expression

6 i = i + 1

7 item = infix_exp[i];

8 postfix_expression[j] = '\0'

Teacher's Signature: _____

9. STOP

→ PUSH() function

1 START

2 accept the item to be inserted into a variable item

3 if top >= SIZE-1

3.1 print stack overflow

4 else

4.1 top = top + 1

4.2 stack(top) = item

5 Stop

→ POP() function

1 START

2 if top < 0

2.1 print underflow

2.2 exit

3 else

3.1 item = stack[top]

3.2 top = top - 1

3.3 print item as poped element.

4 STOP

Teacher's Signature: _____

→ is_operator() function

1. Start
2. accept the character to be checked to a variable symbol.
3. if symbol = ^ or * or / or + or -
   3.1 return 1
4. else
   4.1 return 0
5. stop

→ precedence() function

1. Accept the character to be checked into the variable symbol.
2. if symbol = ^
   2.1 return 3
3. if symbol = * or /
   3.1 return 2
4. if symbol = + or -
   4.1 return 1
5. else
   5.1 return 0

→ evaluate() function

1. Start
2. ask user to enter the postfix expression

Teacher's Signature: _____

Experiment Name / No.: _____ Camlin Page No. 16 Date / /

and input into the variable postfix_exp

3 Repeat while postfix_exp[i] != 0

3.1 if postfix_exp[i] is alpha

5 3.1.1 ask the user to enter the value of the variable and input it into num

3.1.2 push(num)

3.2 else

3.2.1 n1 = pop()

10 3.2.2 n2 = pop()

3.2.3 if postfix_eval[i] = '+'

3.2.3.1 n3 = n1 + n2

3.2.4 if postfix_eval[i] = '-'

3.2.4.1 n3 = n2 - n1

15 3.2.5 if postfix_eval[i] = '*'

3.2.5.1 n3 = n2 * n1

3.2.6 if postfix_eval[i] = '/'

3.2.6.1 n3 = n2 / n1

3.2.7 push(n3)

20 3.3 increment i by 1

4 print pop() as the result of the expression

5. STOP

CONCLUSION

25 The program has been executed correctly and output has been verified.

Teacher's Signature: _____