Yash-Bhardwaj   Section - R   Roll.no : 74
U.rollno. : 2215002014

DSA Assignment - 01

Ans-1
```
import java.Scanner;
class Student {
String name;
int marks;
Student next;
Public Student(string name, int marks) {
this.name = name;
this.marks = marks;
this.next = null;
 }
}
 class stack {
   Private Student top;
   Public stack() {
     this.top = null;
   }
   Public void Push (string name, int marks) {
     Student newNode = new Student(name, marks);
       if(top == null) {
         top = newNode;
       }
        else {
          newNode.next = top;
          top = newNode;
        }
        System.out.println("student added to the stack");
     }
     Public void pop() {
     if (top == null) {
       System.out.println(" Stack is empty ");
     }
       else
         {
```

```
System.out.println ("student removed from the stack ; " + top.name);
        top = top.next ;
    }
}

 Public void display All () {
    if ( top == null) {
       System.out.println ("Stack is empty.");
    }  else
          {
         System.out.Println ("students in the stack ; ");
            Student current = top ;
            while (current != null) {
             system.out.println ("Name ; " + current.name +  ", Marks;" + }
                current.marks );

                    current = current.next;
                }
            }
        }
        Public void display Top3() {
         if (top == null) {
          System.out.Println ("stack is empty, ");
        } else
             { system.out.println ("Top 3 students; ");
              Student temp Top = top;
              Student [] top stuttents = new Student[3];
              for (int i = 0 ; i<3 ; i++) {
                 top Students [i] = null ;
              }
               while (temp Top != null) {
              for (int i = 0 ; i<3; i++) {
                if (top Students [i] == null || tempTop.mark > top students [i].marks)
              {
               for(int j = 2; j>i ; j--) {
                  topStudents[j] = topStudent [j-1];
               }
top Students [i] = tempTop;
  break;
 }
}
  temp Top = temp Top.next;
}
 for(int i=0; i<3; i++){
    if (topStudents [i] != null){
      System.out.println (". Position " + [i+1] + "; Name; " + topstudents[i].
                     name + ", marks; " + topstudents [i], marks);
    }
   }
  }
}
}
 Public class Main {
   Public static void main (string [] args){
      Scanner Sc = new Scanner (system.in);
        Stack stack = new Stack ();
         int ~~Cod~~ choice;
          do {
              System.out.println ("\n Choose an operation;" );
                System.out.Println ("1. Add a student to the stack");
                System.out.Println ("2. Remove a student from the stack");
              System.out.Println ("3. Display all students in the stack");
              System.out.Println ("4. Display the top 3 positions of students");
              System.out.Println ("5. Exit");
              System.out.println ("Enter your choice; ");
                  Choice = scanner.nextInt ();
                    ~~Switch case~~
```

```
switch (choice) {
 Case 1;
   System.out.print("Enter Student name: ");
     Scanner.nextLine();
     Consume newline character String name = Scanner.nextline();
      System.out.print("Enter marks: ");
        int marks = Scanner.nextInt();
         Stack.push(name, marks);
          break;
      Case 2;
           Stack.pop();
         break;
      Case 3;
            stack.displayAll();
           break;

      Case 4;
            Stack.displayTop3();
        break;

      Case 5;
            Sysbem.out.println("Exiting programs.");
        break;
         default:
           Sysbem.out.println("Invalid choice.");
        }
          while (choice != 5);
          Scanner.close();

    }
}
```

Que-2 Convert infix to Prefix and ~~Prefix~~ Postfix notation :-

A). Infix expression : A + B * (- (D/E)

Prefix :- (+ A - (* B C) (/ D E)

Symbol	Stack	Expression
A		A
+	+	A
B	+	AB
*	+ *	AB
C	+ *	ABC
-	-	ABC * +
(	- (	ABC * +
D	- (	ABC * + D
/	- (/	ABC * + D
E	- (	ABC * + DE/
)	- ( )	ABC * + DE/ -

Postfix notation :- ABC * + DE / -

B). (A*B) + (C-D)/E

Symbol	Stack	Operation
(	(	
A	(	A
*	(*	A
B	(*	AB
)	(*)	AB
+	+	AB*
(	+(	AB*
C	+(	AB*C
-	+(-	AB*C
D	+(-	AB*CD
)	+(-)	AB*CD-
/	+/	AB*CD-
E	+/	AB*CD-E/+

Prefix :→ (+(*AB)(/~~C~~-(D)E)

Postfix :→ (AB*CD-E/+)

C). A*(B+C)/D-E

Symbol	Stack	Operation.
A		A
*	*	A
(	*(	A
B	*(	AB
+	*(+	AB
C	*(+	ABC
)	*(+)	ABC
/	/	ABC+*
D	/	ABC+*D
-	-/	ABC+*D
E	-	ABC+*D/E
		ABC+*D/E-

Prefix :- (+A/(*B-(D/E)

Postfix :- (ABC+*D/E-).

D) A + B * (C - D)/E

Symbol	Stack	Operation
A		A
+	+	A
B	+	AB
*	+ *	AB
C	+ * (	AB
(	+ * (	ABC
-	+ * (-	ABC
D	+ * (-	ABCD
)	+ * (-)	ABCD - *
/	+ /	ABCD - *
E	+	ABCD - * E /
		ABCD - * E / +

Prefix :- (+ A / (* B - (D) E)
Postfix :- (ABCD - * E / +)

Que-3 solve all the following Expression :-

1) (5 + 3) x 2 - 8/4 (5 + 3) x 2 - 8/4

Symbol	Stack	Operation
(	(	
5	(	5
+	(+	5
3	(+	53
)	(+)	53
x	x	53 +
2	x	53 + 2
-	x -	53 + 2
8	-	53 + 2 x
/	- /	53 + 2 x 8
4	-	53 + 2 x 8 4 /
(	- (	53 + 2 x 8 4 /
5	- (	53 + 2 x 8 4 / 5
+	- (+	53 + 2 x 8 4 / 5
3	- (+	53 + 2 x 8 4 / 53
)	- (+)	53 + 2 x 8 4 / 53
x	x	53 + 2 x 8 4 / 53 + -

2	x	53 + 2x84 /53 + -2
-	x-	53 +2x84 /53 + -2
8	-	53 +2x84/53 + - 2 x8
/	-/	53 +2x84 /53 + - 2x8
4	-	53 +2 x84/53 + - 2x84/
		53 +2x84/53 -2 x84/-

Postfix :- 53 + 2x84 / 53 - 2 x84/ - .

2.) 4x(6+2)-34 x(6+2)-3.

Symbol	stack	Operation.
4		4
x	x	4
(	x(	4
6	x(	46
+	x(+	46
2	x(+	462
)	x(+)	462
-	-	462+x
34	-	462+x34
x	-x	462 +x34
(	-x(	462 +x34
6	-x(	462 + x346
+	-x(+	462+x346
2	-x(+	462 + x 3462
)	-x(+)	462 +x 3462
-	- -	462+ x 3462 +x
3	-	462 + x3462+x-3
		462 + x 34 62 + x -3-

Postfix :- 462 + x 34 62 + x - 3 -.

3) 10/2+3x5-210/2+3x5-2

Symbol	Stack	operation
10		10
/	/	10
2	/	102
+	/+	102
3	+	102/3
x	+x	102/3
5	+	102/3x5
-	+-	102/3x5
210	+-	102/3x5+210
/	-/	102/3x5+210
2	-[illegible]	102/3x5+210/2
+	-+	102/3x5+210/2
3	-	102/3x5+210/2+3
x	-x	102/3x5+210/2+3
5	-	102/3x5+210/2+3x5
-	--	102/3x5+210/2+3x5-
2	-	102/3x5+210/2+3x5-2
		102/3x5+210/2+3x5-2-

Postfix :- 102/3x5+210/2+3x5-2-

4). (7-2) x4 + 8/2 (7-2) x 4 + 8/2 .

Symbol	Stack	Operation
(	(	
7	(	7
-	(-	7
2	(-	72
)	(-)	72
x	x	72-
4	x	72-4
+	x+	72-4
8	+	72-4x8
/	+/	72-4x8
2	+	72-4x8/2
(	+(	72-4x8/2
7	+(	72-4x8/27
-	+(-	72-4x8/27
2	~~[illegible]~~ +(-	72-4x8/27•2
)	+(-)	72-4x8/27•2
x	x	72-4x8/272-+
4	x	72-4x8/272-+4
+	x+	72-4x8/272-+4
8	+	72-4x8/272-+4x8
/	+/	72-4x8/272-+4x8
2	+	72-4x8/272-+4x8/
		72-4x8/272-+4x8/+

Postfix :- ~~72~~ 72-4x8/272-+4x8/+