Institute of Engineering & Management
Department of Computer Science & Engineering
Object Oriented Programming (IT) Lab for 3rd year 5th semester 2018
Code: CS594D

Date: 4/09/18

WEEK-9

Assignment-1

Problem Statement: Create an abstract class DataHolder containing following data members:

(i) One finite size array
(ii) Two data members front and rear
(iii) insert and delete method

Now Create two concrete classes Stack and Queue using DataHolder Class and also implement required functions.

Source code:

```
import java.util.Scanner;

abstract class DataHolder{
   final int MAX = 100;
   int arr[] = new int[MAX];
   int front=-1, rear=-1;
   abstract void insert(int newElm);
   abstract void del();
   abstract void display();
}

class Stack extends DataHolder{
   void insert(int newElm){
         if(rear >= MAX-1){
               System.out.println("Stack Overflow!");
         }
         else{
               arr[++rear] = newElm;
               System.out.println("\tInserted");
         }
   }
   void del(){
         if(rear < 0){
               System.out.println("Stack Underflow!");
         }
         else{
               rear--;
               System.out.println("\tDeleted");
         }
   }
   void display(){
         if(rear == -1)
               System.out.println("Nothing to display!");
         else{
               System.out.print("The Element(s) are: ");
               for(int i=0;i<=rear;i++)
                     System.out.print(arr[i]+" ");
               System.out.println();
         }
   }
```

```
}

class Queue extends DataHolder{
  void insert(int newElm){
        if(rear == -1){
              front++;
              arr[++rear] = newElm;
              System.out.println("\tInserted");
        }
        else if(rear < MAX-1){
              arr[++rear] = newElm;
              System.out.println("\tInserted");
        }
        else{
              System.out.println("No more space!");
        }
  }
  void del(){
        if(rear == -1){
              System.out.println("Nothing to delete!");
        }
        else{
              front++;
              if(front>rear){
                    rear = front = -1;
              }
              System.out.println("\tDeleted");
        }
  }
  void display(){
        if(rear == -1)
              System.out.println("Nothing to display!");
        else{
              System.out.print("The Element(s) are: ");
              for(int i=front;i<=rear;i++)
                    System.out.print(arr[i]+" ");
              System.out.println();
        }
  }
}

class Main{
  public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter 1-> for Stack 2-> for Queue: ");
        int ch = sc.nextInt();
        if(ch == 1){
              Stack st = new Stack();
              int flag;
              System.out.println("Stack created.\nEnter\n 1:Insert\n
                                        2:Delete\n 3:Display\n 4:Exit");
              do{
                    System.out.print("Enter the command: ");
                    flag = sc.nextInt();
                    switch(flag){
                          case 1:System.out.print("Enter the element: ");
                                 st.insert(sc.nextInt()); break;
                          case 2:st.del(); break;
                          case 3:st.display(); break;
                          case 4:break;
                          default:System.out.println("invalid input!");
```

```
                        }
                }while(flag != 4);
                System.out.println("Bye!");
        }
        else if(ch == 2){
                Queue st = new Queue();
                int flag;
                System.out.println("Queue created.\nEnter\n 1:Insert\n
                                        2:Delete\n 3:Display\n 4:Exit");
                do{
                        System.out.print("Enter the command: ");
                        flag = sc.nextInt();
                        switch(flag){
                                case 1:System.out.print("Enter the element: ");
                                       st.insert(sc.nextInt()); break;
                                case 2:st.del(); break;
                                case 3:st.display(); break;
                                case 4:break;
                                default:System.out.println("invalid input!");
                        }
                }while(flag != 4);
                System.out.println("Bye!");
        }
    }
}
```

Screen-Shot:

```
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Stack_queue$ javac a1.java
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Stack_queue$ java Main
Enter 1-> for Stack 2-> for Queue: 1
Stack created.
Enter
 1:Insert
 2:Delete
 3:Display
 4:Exit
Enter the command: 1
Enter the element: 2
        Inserted
Enter the command: 1
Enter the element: 5
        Inserted
Enter the command: 3
The Element(s) are: 2 5
Enter the command: 2
        Deleted
Enter the command: 3
The Element(s) are: 2
Enter the command: 4
Bye!
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Stack_queue$
```

Fig: Stack

```
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Stack_queue$ javac a1.java
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Stack_queue$ java Main
Enter 1-> for Stack 2-> for Queue: 2
Queue created.
Enter
 1:Insert
 2:Delete
 3:Display
 4:Exit
Enter the command: 1
Enter the element: 4
        Inserted
Enter the command: 1
Enter the element: 5
        Inserted
Enter the command: 3
The Element(s) are: 4 5
Enter the command: 2
        Deleted
Enter the command: 3
The Element(s) are: 5
Enter the command: 4
Bye!
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Stack_queue$
```

Fig: Queue

Assignment-2

Problem Statement: Create a Dynamic Stack using Linked List

Source code:

```
import java.util.Scanner;
class Stack{
  Node top;
  Stack(){
        Node top = null;
  }
  void push(int elm){
        top = new Node(elm, top);
        System.out.println("\tPushed");
  }
  void pop(){
        if(top == null)
              System.out.println("Nothing to delete!");
        else{
              top = top.next;
              System.out.println("\tPopped");
        }
  }
  void display(){
        if(top == null)
              System.out.println("Nothing to display!");
        else{
              Node temp = top;
              System.out.print("The elements in the Stack: ");
              while(temp!=null){
                    System.out.print(temp.elm+" ");
                    temp = temp.next;
              }
```

```
                System.out.println();
            }
        }
    }
    class Node{
        int elm;
        Node next;
        Node(int newElm){
            elm = newElm;
            next = null;
        }
        Node(int newElm, Node nxt){
            elm = newElm;
            next = nxt;
        }
    }
    class Main{
        public static void main(String args[]){
            Scanner sc = new Scanner(System.in);
            Stack st = new Stack();
            int flag;
            System.out.println("Enter\n 1:Push\n 2:Pop\n 3:Display\n
                                                                4:Exit");
            do{
                System.out.print("Enter the command: ");
                flag = sc.nextInt();
                switch(flag){
                    case 1:System.out.print("Enter the element: ");
                           st.push(sc.nextInt()); break;
                    case 2:st.pop(); break;
                    case 3:st.display(); break;
                    case 4: break;
                    default:System.out.println("Invalid input!");
                }
            }while(flag!=4);
            System.out.println("Bye!");
        }
    }
```

Screen-Shot:

```
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Linked list$ javac a2.java
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Linked list$ java Main
Enter
 1:Push
 2:Pop
 3:Display
 4:Exit
Enter the command: 1
Enter the element: 3
        Pushed
Enter the command: 1
Enter the element: 4
        Pushed
Enter the command: 1
Enter the element: 6
        Pushed
Enter the command: 3
The elements in the Stack: 6 4 3
Enter the command: 2
        Popped
Enter the command: 3
The elements in the Stack: 4 3
Enter the command: 4
Bye!
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Linked list$
```

Assignment-3

Problem Statement: Write a java program to sort N values using Stack Data Structure.

Source code:

```
import java.util.Scanner;

class Stack{
  Node top;
  Stack(){
        top = null;
  }
  Stack(Stack copy){
        if(copy.top == null)
              top = null;
        else{
              top = new Node(copy.top.elm);
              Node tempc = copy.top, temp = top;
              while(tempc.next!=null){
                    tempc = tempc.next;
                    temp.next = new Node(tempc.elm);
                    temp = temp.next;
              }
        }
  }
  void push(int elm){
        top = new Node(elm, top);
  }
  void pop(){
        if(top == null)
              return;
        else
              top = top.next;
  }
  void display(){
        if(top == null)
              System.out.println("Nothing to display!");
        else{
              Node temp = top;
              System.out.print("The elements in the Stack: ");
              while(temp!=null){
                    System.out.print(temp.elm+" ");
                    temp = temp.next;
              }
              System.out.println();
        }
  }
  void sort(){
        if(top == null){
              System.out.println("Nothing to sort!");
              return;
        }
        Stack temp = new Stack(), copy = new Stack(this);
        top = new Node(copy.top.elm);
        Node temp_n = copy.top;
        while(temp_n.next != null){
              temp_n = temp_n.next;
              while(top.elm < temp_n.elm){
                    temp.push(top.elm);
                    this.pop();
```

```
                        if(top == null)
                              break;
                  }
                  this.push(temp_n.elm);
                  while(temp.top != null){
                        this.push(temp.top.elm);
                        temp.pop();
                  }
            }
      }
}

class Node{
      int elm;
      Node next;
      Node(int newElm){
            elm = newElm;
            next = null;
      }
      Node(int newElm, Node nxt){
            elm = newElm;
            next = nxt;
      }
}

class Main{
      public static void main(String args[]){
            Scanner sc = new Scanner(System.in);
            Stack st = new Stack();
            System.out.print("Enter the size of stack: ");
            int n = sc.nextInt();
            System.out.print("Enter the elements in the stack: ");
            for(int i=0;i<n;i++){
                  st.push(sc.nextInt());
            }
            System.out.print("\nBefore sorting:\n\t");
            st.display();
            st.sort();
            System.out.print("\nAfter sorting:\n\t");
            st.display();
            sc.close();
      }
}
```

Screen-Shot:

```
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Stack_sort$ javac a3.java
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Stack_sort$ java Main
Enter the size of stack: 9
Enter the elements in the stack: 5 6 2 4 8 9 3 1 7

Before sorting:
        The elements in the Stack: 7 1 3 9 8 4 2 6 5

After sorting:
        The elements in the Stack: 1 2 3 4 5 6 7 8 9
rana@ranajit:~/Git/College_programs/5th SEM/Java/Assignment 9/Stack_sort$
```