OOPS LAB-4

```
import java.io.*;
import java.util.Arrays;
import java.util.Scanner;

interface MyStack{
   void push(Object item) throws StackOverflowException;
   Object pop() throws StackUnderflowException;
   void display(PrintWriter pw);
   boolean isEmpty();
   boolean isFull();
}

class StackOverflowException extends Exception{
   public StackOverflowException(String message){
      super(message);
   }
}

class StackUnderflowException extends Exception{
   public StackUnderflowException(String message){
      super(message);
   }
}
```

```
class StackArray implements MyStack{
    Object[] stack;
    int top;
    private int maxCapacity;

    public StackArray(int initialCapacity){
        stack=new Object[initialCapacity];
        top=-1;
        maxCapacity=100;
    }

    @Override
    public void push(Object item) throws StackOverflowException{
        if (isFull()) throw new StackOverflowException("Cannot push. Stack reached maximum
capacity.");
        if (top + 1==stack.length && stack.length < maxCapacity)
            stack=Arrays.copyOf(stack, Math.min(stack.length * 2,maxCapacity));
        stack[++top]=item;
    }

    @Override
    public Object pop() throws StackUnderflowException{
        if (isEmpty()) throw new StackUnderflowException("Cannot pop from empty stack.");
        Object item=stack[top];
        stack[top--]=null;
        return item;
```

```
    }

    @Override
    public void display(PrintWriter pw){
        if (isEmpty()){
            pw.println("Stack elements (top to bottom): [ ]");
            return;
        }
        pw.print("Stack elements (top to bottom): [ ");
        for (int i=top;i>=0;i--){
            pw.print(stack[i]);
            if (i!=0) pw.print(", ");
        }
        pw.println(" ]");
    }

    @Override
    public boolean isEmpty(){ return top==-1; }
    @Override
    public boolean isFull(){ return top + 1==maxCapacity; }
}

public class StackAdt{
    public static void main(String[] args){
        try (Scanner sc = new Scanner(System.in);
             PrintWriter log = new PrintWriter(new File("stack_log.txt"))){
```

```
class Logger{
   void print(String s){
      System.out.print(s);
      log.print(s);
   }
   void println(String s){
      System.out.println(s);
      log.println(s);
   }
}
Logger print = new Logger();

print.print("Enter initial stack size: ");
int initialSize = sc.nextInt();
sc.nextLine();
print.println(String.valueOf(initialSize));

StackArray stack = new StackArray(initialSize);

int choice;
do {
   print.println("\n--- Stack Menu ---");
   print.println("1. Push  2. Pop  3. Display  4. Exit");
   print.print("Choice: ");
   choice = sc.nextInt();
   sc.nextLine();
```

```
print.println(String.valueOf(choice));


switch (choice){
   case 1:
      print.print("Enter value to push: ");
      String value = sc.nextLine();
      print.println(value);
      try {
         stack.push(value);
         print.println(value + " pushed to stack.");
      } catch (StackOverflowException e) {
         print.println("Exception: " + e.getMessage());
      }
      break;

   case 2:
      try{
         Object popped = stack.pop();
         print.println(popped + " popped from stack.");
      }
      catch (StackUnderflowException e){
         print.println("Exception: " + e.getMessage());
      }
      break;

   case 3:
```

```
                print.print("Stack elements (top to bottom): [ ");
                for (int i = stack.top; i >= 0; i--){
                    print.print(String.valueOf(stack.stack[i]));
                    if (i != 0) print.print(", ");
                }
                print.println(" ]");
                break;

            case 4:
                print.println("Program exiting...");
                break;

            default:
                print.println("Invalid choice. Try again.");
        }

    } while (choice != 4);

    } catch (IOException e){
        System.out.println("File error: " + e.getMessage());
    }
  }
}
```

SCREENSHOTS OF OUTPUT

```
PS D:\OOPS in Java\LAB-4> javac StackAdt.java
PS D:\OOPS in Java\LAB-4> java StackAdt
Enter initial stack size: 2
2

--- Stack Menu ---
1. Push  2. Pop  3. Display  4. Exit
Choice: 1
1
Enter value to push: pandi
pandi
pandi pushed to stack.

--- Stack Menu ---
1. Push  2. Pop  3. Display  4. Exit
Choice: 1
1
Enter value to push: kabilesh
kabilesh
kabilesh pushed to stack.

--- Stack Menu ---
1. Push  2. Pop  3. Display  4. Exit
Choice: 1
1
Enter value to push: harshini
harshini
harshini pushed to stack.

--- Stack Menu ---
1. Push  2. Pop  3. Display  4. Exit
Choice: 3
3
Stack elements (top to bottom): [ harshini, kabilesh, pandi ]

--- Stack Menu ---
1. Push  2. Pop  3. Display  4. Exit
```

```
--- Stack Menu ---
1. Push  2. Pop  3. Display  4. Exit
Choice: 2
2
harshini popped from stack.

--- Stack Menu ---
1. Push  2. Pop  3. Display  4. Exit
Choice: 2
2
kabilesh popped from stack.

--- Stack Menu ---
1. Push  2. Pop  3. Display  4. Exit
Choice: 2
2
pandi popped from stack.

--- Stack Menu ---
1. Push  2. Pop  3. Display  4. Exit
Choice: 2
2
Exception: Cannot pop from empty stack.

--- Stack Menu ---
1. Push  2. Pop  3. Display  4. Exit
Choice: Exception in thread "main" java.util.NoSuchElementException
        at java.base/java.util.Scanner.throwFor(Scanner.java:962)
        at java.base/java.util.Scanner.next(Scanner.java:1619)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2284)
        at java.base/java.util.Scanner.nextInt(Scanner.java:2238)
        at StackAdt.main(StackAdt.java:106)
PS D:\OOPS in Java\LAB-4>
```

stack_log.txt

Enter initial stack size: 2

--- Stack Menu ---

1. Push 2. Pop 3. Display 4. Exit

Choice: 1

Enter value to push: pandi

pandi pushed to stack.

--- Stack Menu ---

1. Push 2. Pop 3. Display 4. Exit

Choice: 1

Enter value to push: kabilesh

kabilesh pushed to stack.

--- Stack Menu ---

1. Push 2. Pop 3. Display 4. Exit

Choice: 1

Enter value to push: harshini

harshini pushed to stack.

--- Stack Menu ---

1. Push 2. Pop 3. Display 4. Exit

Choice: 3

Stack elements (top to bottom): [harshini, kabilesh, pandi]

--- Stack Menu ---

1. Push 2. Pop 3. Display 4. Exit

Choice: 2

harshini popped from stack.

--- Stack Menu ---

1. Push 2. Pop 3. Display 4. Exit

Choice: 2

kabilesh popped from stack.

--- Stack Menu ---

1. Push 2. Pop 3. Display 4. Exit

Choice: 2

pandi popped from stack.

--- Stack Menu ---

1. Push 2. Pop 3. Display 4. Exit

Choice: 2

Exception: Cannot pop from empty stack.

--- Stack Menu ---

1. Push 2. Pop 3. Display 4. Exit

Choice: