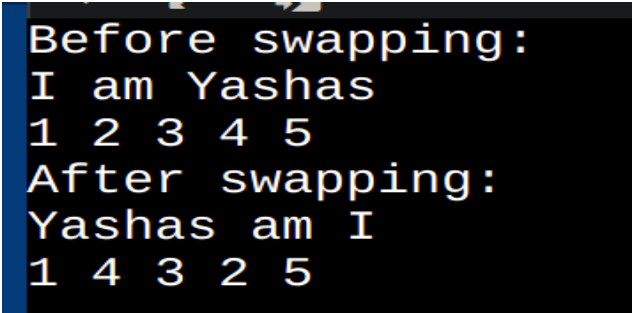


OOP LAB 7

Q1) Write a generic method to exchange the positions of two different elements in an array.

```
public class Main{
    static <T>
    void swap (T[] a, int i, int j) {
        T t = a[i];
        a[i] = a[j];
        a[j] = t;
    }
    public static void main(String[] args) {
        String a[] = {"I", "am", "Yashas"};
        System.out.println("Before swapping:");
        for(int i=0;i<a.length;i++){
            System.out.print(a[i]+" ");
        }
        System.out.println();
        swap(a, 0, 2);
        Integer b[] = {1, 2, 3, 4, 5};
        for(int i=0;i<b.length;i++){
            System.out.print(b[i]+" ");
        }
        swap(b, 1, 3);
        System.out.println("\nAfter swapping:");
        for(int i=0;i<a.length;i++){
            System.out.print(a[i]+" ");
        }
        System.out.println();
        for(int i=0;i<b.length;i++){
            System.out.print(b[i]+" ");
        }
        System.out.println();
    }
}
```



```
Before swapping:
I am Yashas
1 2 3 4 5
After swapping:
Yashas am I
1 4 3 2 5
```

Q2) Define a simple generic stack class and show the use of the generic class for two different class types Student and Employee class objects.

```
import java.util.*;
class stack<T> {
//T[] stk = T[] new Object[20];
T[] stk = (T[])new Object[20];
static int max = 20;
int top = -1;
void push(T ele) {
if(top == max) {
System.out.println("Stack Overflow");
return;
}
stk[++top] = ele;
}
T pop() {
if(top == -1) {
System.out.println("Stack Underflow");
}
return stk[top--];
}
void disp() {
for(int i=top;i>=0;i--) {
System.out.print(stk[i].toString());
}
System.out.println();
}
}
class Student {
String first;
String last;
String email;
int section;
public Student(String first, String last, String email, int section) {
this.first = first;
this.last = last;
this.email = email;
this.section = section;
}
public String toString() {
return section + " " + first + " " + last + " " + email + "\n";
}
```

```

}
}
class Employee {
String first;String last;
String email;
int empid;
public Employee(String first, String last, String email, int empid) {
this.first = first;
this.last = last;
this.email = email;
this.empid = empid;
}
public String toString() {
return empid + " " + first + " " + last + " " + email + "\n";
}
}
public class Main{
public static void main(String[] args) {
stack<Student> stu = new stack<Student> ();
stack<Employee> emp = new stack<Employee> ();
System.out.println("Student:");
stu.push(new Student("Ayush", "Goyal", "abc@gmail.com", 1));
stu.push(new Student("Anubhav", "Bagri", "xyz@gmail.com", 2));
stu.push(new Student("Dipesh", "Singh", "ghi@gmail.com", 3));
stu.disp();
System.out.println("Popping once : ");
stu.pop();
stu.disp();
System.out.println("Employee:");
emp.push(new Employee("Malaya", "Khandelwal", "abc@def.com", 100));
emp.push(new Employee("Kalpana", "Cahkro", "xyz@def.com", 200));
emp.push(new Employee("Satyendra", "Mishra", "ghi@def.com", 300));
emp.disp();
System.out.println("Popping twice : ");
emp.pop();
emp.pop();
emp.disp();
}
}

```

```

Student:
3 Dipesh Singh ghi@gmail.com
2 Anubhav Bagri xyz@gmail.com
1 Ayush Goyal abc@gmail.com

Popping once :
2 Anubhav Bagri xyz@gmail.com
1 Ayush Goyal abc@gmail.com

Employee:
300 Satyendra Mishra ghi@def.com
200 Kalpana Cahkro xyz@def.com
100 Malaya Khandelwal abc@def.com

Popping twice :
100 Malaya Khandelwal abc@def.com

```

Q3) Write a program to demonstrate the use of wildcard arguments.

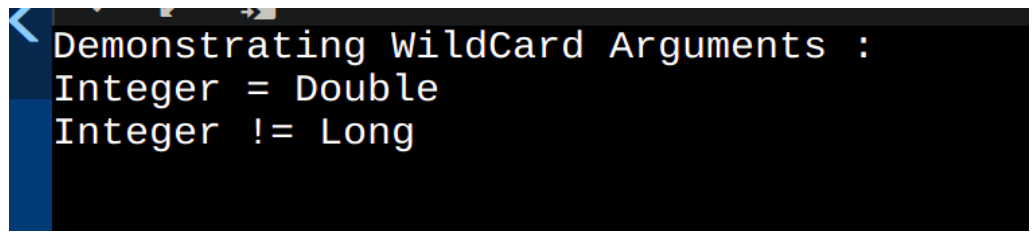
```

class NumFns<T extends Number> {
    T num;
    NumFns(T n) {
        num =n;
    }
    boolean absEqual (NumFns<?> ob) {
        if(Math.abs(num.doubleValue()) == Math.abs(ob.num.doubleValue()))
            return true;
        return false;
    }
}

public class Main{
    public static void main(String[] args) {
        NumFns<Integer> i = new NumFns<Integer> (8);
        NumFns<Double> d = new NumFns<Double> (-8.0);
        NumFns<Long> l = new NumFns<Long> (6L);
        System.out.println("Demonstrating WildCard Arguments : ");
        if(i.absEqual(d))
            System.out.println("Integer = Double");
        else
            System.out.println("Integer != Double");
        if(i.absEqual(l))
            System.out.println("Integer = Long");
        else
    }
}

```

```
System.out.println("Integer != Long");  
}  
}
```



```
< Demonstrating WildCard Arguments :  
Integer = Double  
Integer != Long
```