Q1.

```
class item {
    String title;
    String author;
    int year;

    public String toString(){
        return title + " " +author + " "+year;
    }

    public item() {
    }

    public item(String title, String author, int year) {
        this.title = title;
        this.author = author;
        this.year = year;
    }
}

class book extends item{
    String publisher;
    int isbn;

    public String toString(){
        return title + " " +author + " "+year + " " +publisher + " "
+isbn;
    }

    public book(String title, String author, int year, String publisher,
int isbn) {
        super(title, author, year);
        this.publisher = publisher;
        this.isbn = isbn;
    }
}


class magazine extends item{
    String publisher;
    int issuenum;

    public String toString(){
        return title + " " +author + " "+year + " " +publisher + " "
+issuenum;
    }


    public magazine(String title, String author, int year, String
publisher, int issuenum) {
        super(title, author, year);
        this.publisher = publisher;
        this.issuenum = issuenum;
    }
}

class dvd extends item{
    String director;
    int length;

    public String toString(){
        return title + " " +author + " "+year + " " +director + " "
+length;
    }


    public dvd(String title, String author, int year, String director, int
length) {
        super(title, author, year);
        this.director = director;
        this.length = length;
    }
}

public class q1{
    public static void main(String[] args) {
        item i = new item("Alchemist","Paulo", 2003);
        book b = new book("Alchemist","Paulo", 2003, "Penguin", 12345 );
        magazine m = new magazine("Alchemist","Paulo", 2003, "Vogue", 12);
        dvd d = new dvd("Alchemist","Paulo", 2003, "Dave", 250);

        //by printing sout(object), all details are dispayed as string
too.
        System.out.println(i.toString());
        System.out.println(b.toString());
        System.out.println(m.toString());
        System.out.println(d.toString());
    }
}
```

Q2.

```
class person{
    String name;
    String email;
    int phone;

    public String toString(){
        return name + " " +email + " "+phone;
    }

    public person() {
    }

    public person(String name, String email, int phone) {
        this.name = name;
        this.email = email;
        this.phone = phone;
    }
}
class student extends person{
    int sid;
    String major;

    public int getGPA(int g){
        return g;
    }


    public student(String name, String email, int phone, int id, String
major) {
        super(name, email, phone);
        this.sid = id;
        this.major = major;
    }
}
class faculty extends person{
    int fid;
    String department;

    public String getRank(String g){
        return g;
    }


    public faculty(String name, String email, int phone, int fid, String
department) {
        super(name, email, phone);
        this.fid = fid;
        this.department = department;
    }
}

class staff extends person{
    int staffid;
    String job;

    public float getSalary(float g){
        return g;
    }

}

public class q2 {
}
```

Q3.

```
class Robot{
    int x, y;
    String direction;

    public Robot(int x, int y, String direction) {
        this.x = x;
        this.y = y;
        this.direction = direction;
    }
    public void display() {
        System.out.println("Current position: (" + x + ", " + y + "),
facing " + direction);
    }
}
class moving extends Robot{

    public moving(int x, int y, String direction) {
        super(x, y, direction);
    }

    public void move(int steps){
        switch (direction){
            case "N":
                y += steps;
                break;
            case "S":
                y -= steps;
                break;
            case "E":
                x += steps;
                break;
            case "W":
                x -= steps;
                break;
            default:
                System.out.println("Invalid direction");
        }
    }
    public void display() {
        System.out.println("Updated position: (" + x + ", " + y + "),
facing " + direction);
    }
}
public class q3 {

    public static void main(String[] args) {
        Robot r = new Robot(2,4,"N");
        r.display();
        moving robot  = new moving(2,4,"N");
        robot.move(5);
        robot.display();
    }


}
```

Q4.

```
class account{
    double balance;
    public account() {
    }

    public account(double balance) {
        this.balance = balance;
    }

    public void debit(double value){
        if(value<=balance){
            balance=balance-value;
        }
        else{
            System.out.println("Insufficient balance");
        }

    }
    public void credit(float value){
        balance=balance+value;
    }
    public double getBalance() {
        return balance;
    }
}
class saving extends account{
    int time;
    double interest;


    public saving(double balance, int time) {
        super(balance);
        this.time = time;
    }
    public void calculateInterest(double rate) {
        interest = getBalance() * rate * time;
    }
    public void credit(){
        balance = getBalance() +interest;
    }
}
class checking extends account{
    @Override
    public double getBalance() {
        return super.getBalance();
    }
}
public class q4{
    public static void main(String[] args) {
        saving s = new saving(1000,4);
        System.out.println("Before transactions");
        System.out.println(s.getBalance());
        s.calculateInterest(0.05);
        s.credit();
        s.debit(120);
        System.out.println("After transactions:");
        System.out.println(s.getBalance() );
    }
}
```

Q5.

```
import java.util.Scanner;

class Sorting {
    // we can also use Arrays.sort(array) method to avoid the long working
    public void sort(int[] a, int n) {
        int i = 0, j = 0, temp = 0;
        for (i = 0; i<n; i++) {
            for (j = i+ 1; j<n; j++) {
                if (a[i] > a[j]) {
                    temp = a[i];
                    a[i] = a[j];
                    a[j] = temp;
                }
            }
        }
        System.out.println("In ascending order:");
        for (i = 0; i < n; i++) {
            System.out.println(a[i]);
        }
    }

    public void sort(String[] a, int n) {
        int i = 0, j = 0;
        String temp;

        for (i = 0; i < n; i++) {
            for (j = 0; j < n; j++) {
                if ((a[i].compareToIgnoreCase(a[j])) < 0) { // It compares
the unicode value, ignoring upper case, it converts to lower case
                    temp = a[i];                                //if equal
returns 0, first string less returns <0,  first string greater returns >0
                    a[i] = a[j];
                    a[j] = temp;
                }
            }
        }

        System.out.println("In ascending order:");
        for (i = 0; i < n; i++) {
            System.out.println(a[i]);
        }
    }

    public void sort(int[] a, boolean descending, int n) {
        int i = 0, j = 0, temp = 0;
        if (descending == Boolean.TRUE) {
            for (i = 0; i < n; i++) {
                for (j = i + 1; j < n; j++) {
                    if (a[i] < a[j]) {
                        temp = a[i];
                        a[i] = a[j];
                        a[j] = temp;
                    }
                }
            }
            System.out.println("In descending order:");
            for (i = 0; i < n; i++) {
                System.out.println(a[i]);
            }
        }
        if (descending == Boolean.FALSE) {
            for (i = 0; i < n; i++) {
                for (j = i + 1; j < n; j++) {
                    if (a[i] > a[j]) {
                        temp = a[i];
                        a[i] = a[j];
                        a[j] = temp;
                    }
                }
            }
            System.out.println("In ascending order:");
            for (i = 0; i < n; i++) {
                System.out.println(a[i]);
            }
        }
    }

    public void sort(String[] a, boolean descending, int n){
        int i = 0, j = 0;
        String temp;
        if(descending==Boolean.FALSE){
            for (i = 0; i < n; i++) {
                for (j = 0; j < n; j++) {
                    if ((a[i].compareToIgnoreCase(a[j])) < 0) {
                        temp = a[i];
                        a[i] = a[j];
                        a[j] = temp;
                    }
                }
            }

            System.out.println("In ascending order:");
            for (i = 0; i < n; i++) {
                System.out.println(a[i]);
            }
        }

        if(descending==Boolean.TRUE){
            for (i = 0; i < n; i++) {
                for (j = 0; j < n; j++) {
                    if ((a[i].compareToIgnoreCase(a[j])) > 0) {
                        temp = a[i];
                        a[i] = a[j];
                        a[j] = temp;
                    }
                }
            }

            System.out.println("In descending order:");
            for (i = 0; i < n; i++) {
                System.out.println(a[i]);
            }
        }
    }
}
public class q5 {
    public static void main(String[] args) {
        Scanner A =new Scanner(System.in);
        Sorting s = new Sorting();
        System.out.println("Enter array size:");
        int n= A.nextInt();
        int arr1[] = new int[n];

        int i=0, j=0;
        System.out.println("Enter Integer array elements:");
        for(i=0; i<n; i++){
            arr1[i]= A.nextInt();
        }
        String arr2[] = new String[n];
        System.out.println("Enter String array elements:");
        for(i=0; i<n; i++){
            arr2[i]= A.next();
        }
        boolean d1, d2;
        System.out.println("Enter boolean value for integer array:");
        d1= A.nextBoolean();

        System.out.println("Enter boolean value for String array:");
        d2= A.nextBoolean();

        s.sort(arr1, n);
        s.sort(arr2, n);
        s.sort(arr1, d1, n);
        s.sort(arr2, d2, n);
    }
}
```