**Exercise 1: Inventory Management System**

**Code:**

import java.io.\*;

import java.util.\*;

public class Main

{

public static void main(String[] args) {

Inventory manager=new Inventory();

Product p1=new Product(1,"PC",10,50000);

Product p2=new Product(2,"Laptop",5,1000000);

Product p3=new Product(3,"Phone",4,400000);

Product p4=new Product(4,"Keyboard",5,3000);

manager.add(p1);

manager.add(p2);

manager.add(p3);

manager.add(p4);

manager.add(p1);

System.out.println("\nInventory");

manager.display();

manager.update(2,"Mouse",3,1500);

manager.update(5,"Speaker",1,5000);

System.out.println("\nInventory");

manager.display();

manager.delete(4);

manager.delete(8);

System.out.println("\nInventory");

manager.display();

}

}

class Product{

int productId;

String productName;

int quantity;

int price;

public Product(int a,String b,int c,int d)

{

this.productId=a;

this.productName=b;

this.quantity=c;

this.price=d;

}

public void show(Product p)

{

System.out.println(p.productId+" "+p.productName+" "+p.quantity+" "+p.price);

}

}

class Inventory{

HashMap<Integer,Product> inventory;

public Inventory()

{

inventory=new HashMap<>();

}

public void add(Product p)

{

if(!inventory.containsKey(p.productId)){

inventory.put(p.productId,p);

System.out.println("Adding product with id "+p.productId);

}

else

System.out.println("Product with id "+p.productId+" Already exists");

}

public void delete(int id)

{

if(inventory.containsKey(id)){

System.out.println("Deleting product with id "+id);

inventory.remove(id);

}

else

System.out.println(id+" Product does not exist");

}

public void update(int id,String name,int qty,int cost)

{

if(inventory.containsKey(id))

{

System.out.println("Updating product with id "+id);

Product p=inventory.get(id);

p.productName=name;

p.quantity=qty;

p.price=cost;

}

}

public void display()

{

if(inventory.isEmpty())

System.out.println("The Inventory is Empty");

else{

for(Product p:inventory.values())

{

p.show(p);

}

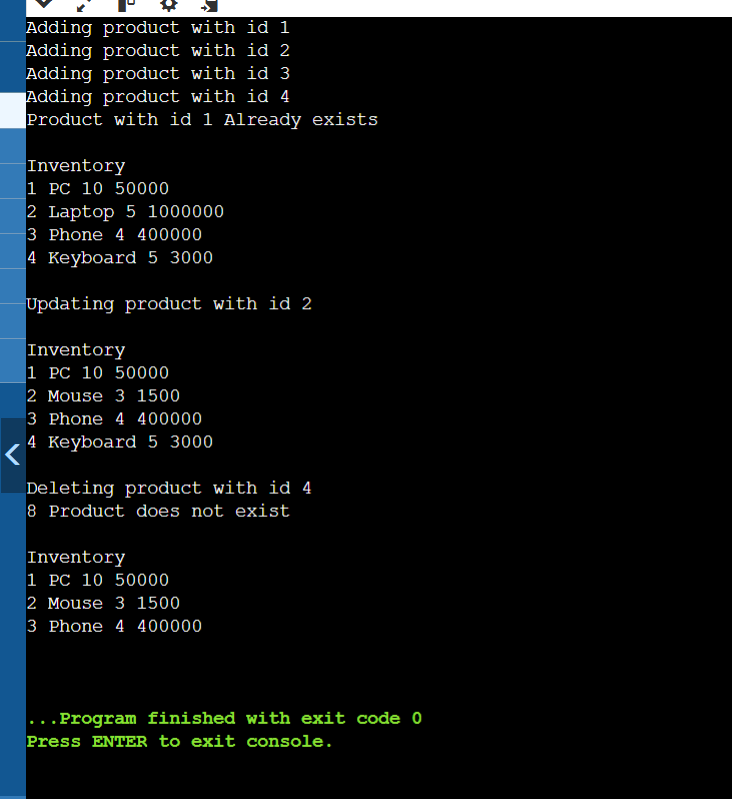
}

System.out.println();

}

}

**Output:**

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Exercise 3: Sorting Customer Orders**

**Code:**

class Order

{

int id;

String name;

double amt;

public Order(int id,String name,double amt)

{

this.id=id;

this.name=name;

this.amt=amt;

}

public void show()

{

System.out.println("ID: "+id+" Name: "+name+" Amount: "+amt);

}

}

public class Main

{

public static void BubbleSort(Order[] o,int n)

{

int i,j;

for(i=0;i<n;i++)

{

for(j=0;j<n-i-1;j++)

{

if(o[j].amt>o[j+1].amt)

{

Order temp=o[j];

o[j]=o[j+1];

o[j+1]=temp;

}

}

}

}

public static void QuickSort(Order[] o,int low,int high)

{

if(low<high)

{

int p=partition(o,low,high);

QuickSort(o,low,p-1);

QuickSort(o,p+1,high);

}

}

public static int partition(Order[] o,int low,int high)

{

double pivot= o[high].amt;

int i=low-1;

int j;

for(j=low;j<high;j++)

{

if(o[j].amt<pivot)

{

i++;

Order temp=o[i];

o[i]=o[j];

o[j]=temp;

}

}

Order temp=o[i+1];

o[i+1]=o[high];

o[high]=temp;

return i+1;

}

public static void main(String[] args) {

Order o[]={

new Order(1,"Alice",1200),

new Order(2,"Bob",500),

new Order(3,"Charlie",400),

new Order(4,"David",1750),

new Order(5,"Emma",700)

};

Order o2[]={

new Order(1,"Alice",1200),

new Order(2,"Bob",500),

new Order(3,"Charlie",400),

new Order(4,"David",1750),

new Order(5,"Emma",700)

};

System.out.println("\nOriginal: ");

for(Order k:o)

k.show();

BubbleSort(o,o.length);

System.out.println("\nBubble Sort");

for(Order k:o)

k.show();

QuickSort(o2,0,o2.length-1);

System.out.println("\nQuick Sort");

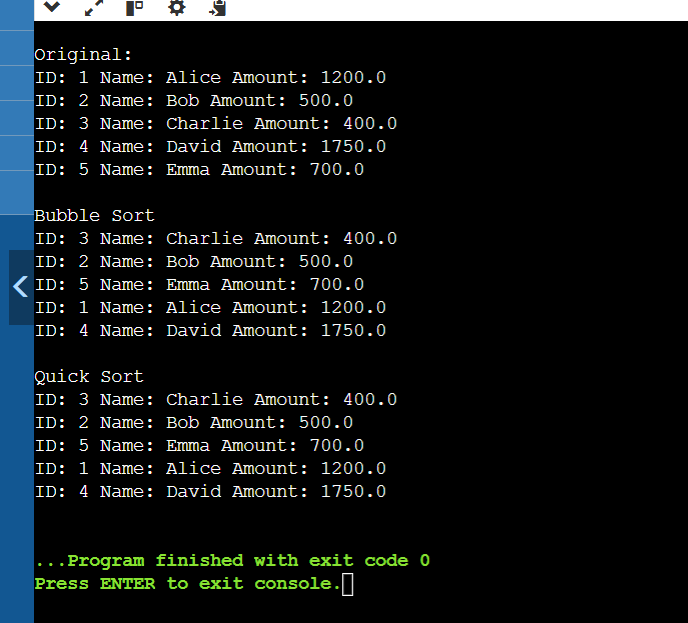
for(Order k:o2)

k.show();

}

}

**Output:**

****

**Exercise 4: Employee Management System**

**Code:**

class Employee{

private int id;

private String name;

private String position;

private double salary;

public Employee(int id,String name,String position,double salary)

{

this.id=id;

this.name=name;

this.position=position;

this.salary=salary;

}

public void display()

{

System.out.println("ID: "+id+" Name: "+name+" Position: "+position+" salary: "+salary);

}

public int getid(){

return id;

}

public String getname()

{

return name;

}

public String getposition()

{

return position;

}

public double getsalary()

{

return salary;

}

}

class EmployeeManagement{

private Employee[] emp;

private int cnt;

public EmployeeManagement(int cnt)

{

emp=new Employee[cnt];

cnt=0;

}

public void addemployee(Employee e)

{

if(cnt<emp.length)

{

emp[cnt++]=e;

System.out.println("Employee added");

}

else

{

System.out.println("Array is full,cannot add");

}

}

public Employee search(int id)

{

int i;

for(i=0;i<cnt;i++)

{

if(emp[i].getid()==id)

return emp[i];

}

return null;

}

public void showAll()

{

if(cnt==0)

System.out.println("Empty. no employees");

else

{

for(int i=0;i<cnt;i++)

{

emp[i].display();

}

}

}

public void delete(int id)

{

int j;

for(int i=0;i<cnt;i++)

{

if(emp[i].getid()==id)

{

for(j=i;j<cnt-1;j++)

{

emp[j]=emp[j+1];

}

cnt--;

emp[cnt]=null;

System.out.println("Employee deleted");

return;

}

}

System.out.println("Not found");

}

}

public class Main

{

public static void main(String[] args) {

EmployeeManagement em=new EmployeeManagement(5);

em.addemployee(new Employee(1,"Bob","Manager",60000));

em.addemployee(new Employee(2,"David","Doctor",75000));

em.addemployee(new Employee(3,"Charlie","Analyst",55000));

System.out.println("\nEmployees: ");

em.showAll();

System.out.println("\nAdd: ");

Employee k=new Employee(4,"Alice","HR",80000);

em.addemployee(k);

em.showAll();

System.out.println("\nEmployee Search: ");

Employee s=em.search(3);

if(s!=null)

s.display();

else

System.out.println("Not found ");

System.out.println("\nDeleting ");

em.delete(2);

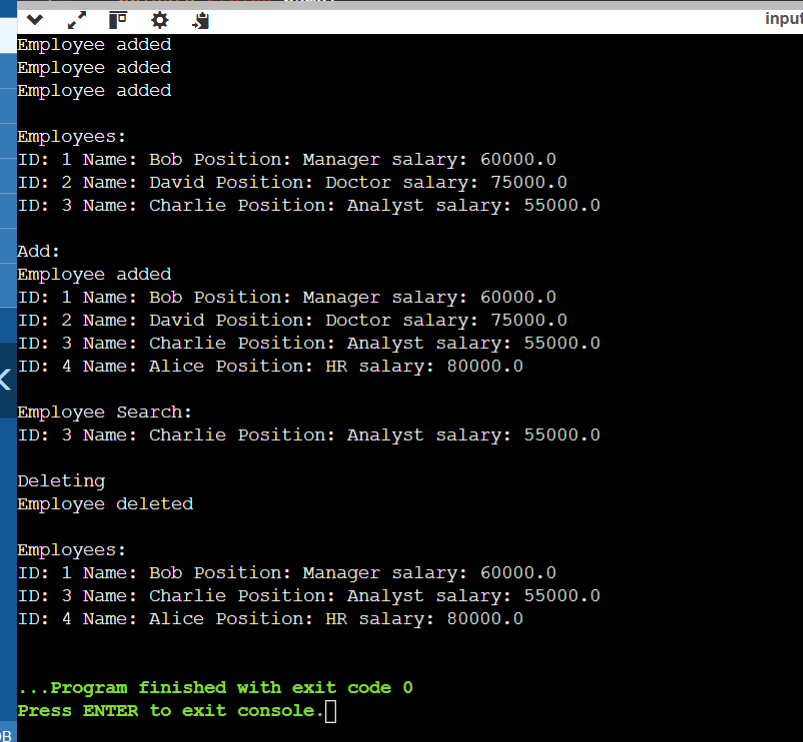
System.out.println("\nEmployees: ");

em.showAll();

}

}

**Output:**

****

**Exercise 5: Task Management System**

class Task{

int id;

String name;

String status;

public Task(int id,String name,String status)

{

this.id=id;

this.name=name;

this.status=status;

}

public void display()

{

System.out.println("ID: "+id+" Name: "+name+" Status: "+status);

}

}

class Node{

Task t;

Node next;

public Node(Task t){

this.t=t;

this.next=null;

}

}

class TaskManagement{

private Node head;

public void add(Task t){

Node curr=new Node(t);

if(head==null)

{

head=curr;

}

else{

Node temp=head;

while(temp.next!=null)

{

temp=temp.next;

}

temp.next=curr;

}

System.out.println("Task: "+t.name);

}

public Task search(int id)

{

Node temp=head;

while(temp!=null)

{

if(temp.t.id==id)

return temp.t;

temp=temp.next;

}

return null;

}

public void displayTasks()

{

if(head==null)

{

System.out.println("empty");

return;}

Node curr=head;

while(curr!=null)

{

curr.t.display();

curr=curr.next;

}

}

public void delete(int id)

{

if(head==null)

{

System.out.println("empty");

return;}

if(head.t.id==id)

{

head=head.next;

System.out.println("Deletd");

return;

}

Node prev=head;

Node curr=head.next;

while(curr!=null)

{

if(curr.t.id==id)

{

prev.next=curr.next;

System.out.println("Deletd");

return;

}

prev=curr;

curr=curr.next;

}

System.out.println("Task not found");

}

}

public class Main

{

public static void main(String[] args) {

TaskManagement tm=new TaskManagement();

tm.add(new Task(1,"Testing","Completed"));

tm.add(new Task(2,"Dev","Pending"));

tm.add(new Task(3,"Testing","In progress"));

System.out.println("\nTasks: ");

tm.displayTasks();

System.out.println("\nAdding task ");

Task k=new Task(4,"Dev","Completed");

tm.add(k);

System.out.println("\nSearching: ");

Task s=tm.search(3);

if(s!=null)

s.display();

else

System.out.println("\nNot found: ");

System.out.println("\nDeleting: ");

tm.delete(2);

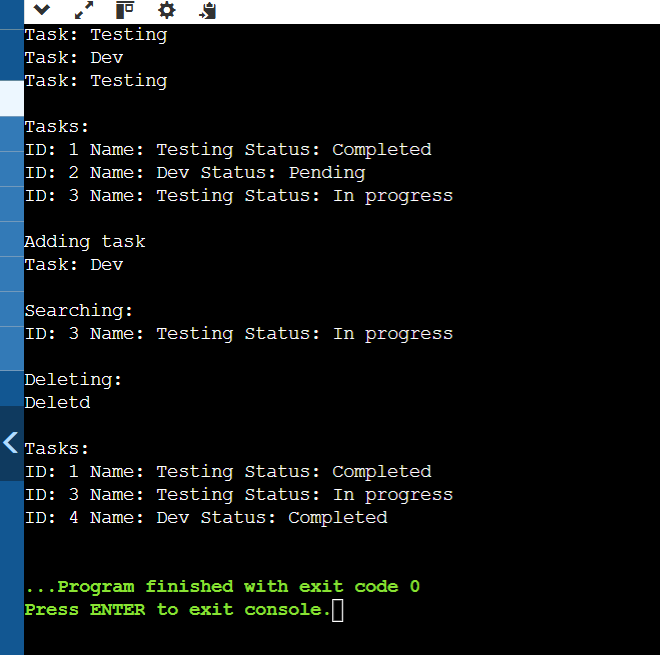
System.out.println("\nTasks: ");

tm.displayTasks();

}

}

**Output:**

****

**Exercise 6: Library Management System**

**Code:**

class Book{

int id;

String title;

String author;

public Book(int id,String title,String author)

{

this.id=id;

this.title=title;

this.author=author;

}

public void display()

{

System.out.println("ID: "+id+" Title: "+title+" Author: "+author);

}

public String gettitle()

{

return title;

}

}

class LibraryManagement{

Book[] b;

public LibraryManagement(Book[] b)

{

this.b=b;

}

public Book LinearSearch(String title)

{

for(Book k:b)

{

if(k.title.equalsIgnoreCase(title))

return k;

}

return null;

}

public Book BinarySearch(String title)

{

int left=0,right=b.length-1;

while(left<=right)

{

int mid=(left+right)/2;

int c=b[mid].title.compareToIgnoreCase(title);

if(c==0)

return b[mid];

else if(c<0)

left=mid+1;

else

right=mid-1;

}

return null;

}

public void show()

{

for(Book k:b)

{

k.display();

}

}

}

public class Main

{

public static void main(String[] args) {

Book[] b={

new Book(1," Harry Potter "," JK Rowling"),

new Book(2,"Aptitude","RS Agarwal"),

new Book(3,"Merchant of Venice","Shakespeare")

};

LibraryManagement lib=new LibraryManagement(b);

System.out.println("\nBooks:");

lib.show();

System.out.println("\nLinear Serach for Aptitude: ");

Book b1=lib.LinearSearch("Aptitude");

if(b1!=null)

{

System.out.println("\nBook found");

b1.display();

}

else

System.out.println("\nNot found");

Book b2=lib.BinarySearch("Merchant of Venice");

if(b2!=null)

{

System.out.println("\nBook found");

b2.display();

}

else

System.out.println("\nNot found");

}

}

**Output:**

