Class Employee

```java
public abstract class Employee {

	private String name;
	private int id;

	public Employee(String name, int id) {
		this.name = name;
		this.id = id;
	}
	public Employee(Employee e) {
		this.name = e.name;
		this.id = e.id;
	}
	public String getName() {
		return name;
	}
	public int getId() {
		return id;
	}
	public void display() {
		System.out.println("Employee name: " + name);
		System.out.println("Employee id: " + id);
	}
	public abstract double calculatePay();

}
```

Class PartTimeEmp

```java
public class PartTimeEmp extends Employee{

	private int nbHours;
	private int rate;

	public PartTimeEmp(String name, int id, int nbHours, int rate) {
		super(name, id);
		this.nbHours = nbHours;
		this.rate = rate;
	}
	public PartTimeEmp(PartTimeEmp pt) {
		super(pt);
		this.nbHours = pt.nbHours;
		this.rate = pt.rate;
	}
	public void display() {
		super.display();
		System.out.println("Number of work hours: " + nbHours);
		System.out.println("Hourly rate: " + rate);
	}
	public double calculatePay() {
		return nbHours * 4 * rate;
	}
	public int getNbHours() {
		return nbHours;
	}
	public int getRate() {
		return rate;
	}
}
```

Class FullTimeEmp

```
public class FullTimeEmp extends Employee{

	private double salary;

	public FullTimeEmp(String name, int id, double salary) {
		super(name, id);
		this.salary = salary;
	}
	public FullTimeEmp(FullTimeEmp ft) {
		super(ft);
		this.salary = ft.salary;
	}
	public void display() {
		super.display();
		System.out.println("Employee salary: " + salary);
	}
	public double calculatePay() {
		return salary - (salary * 0.09);
	}
	public double getSalary() {
		return salary;
	}
}
```

Class Company

```java
public class Company {

	private String name;
	private Employee arrEmp[];
	private int nbEmp;

public Company(String name, int size) throws NegativeArraySizeException{
		if(size < 0) throw new
		NegativeArraySizeException("Company size can't be negative");
		this.name = name;
		arrEmp = new Employee[size];
		nbEmp = 0;
	}

	public void displayAll() {
		for(int i = 0; i < nbEmp; i++) {
			arrEmp[i].display();
		}
	}

	public void addEmployee(Employee e) throws IllegalStateException{
		if(nbEmp == arrEmp.length)
			throw new IllegalStateException("Array is full!");
		if(e instanceof PartTimeEmp)
			arrEmp[nbEmp++] = new PartTimeEmp((PartTimeEmp)e);
		else
			arrEmp[nbEmp++] = new FullTimeEmp((FullTimeEmp) e);
	}

	public int searchEmployee(String name) {
		for(int i = 0; i < nbEmp; i++)
			if(arrEmp[i].getName().equalsIgnoreCase(name))
				return i;
		return -1;
	}

public void deleteEmployee(String name) throws IndexOutOfBoundsException{
		int index = searchEmployee(name);
		if(index == -1)
	throw new IndexOutOfBoundsException("Employee is not found to delete");
		arrEmp[index] = arrEmp[nbEmp-1];
		arrEmp[nbEmp-1] = null;
		nbEmp--;
	}
	public double getYearlyPay(String name) {
		int index = searchEmployee(name);
		if(index == -1)
			return -1;
		return arrEmp[index].calculatePay() * 12;
	}

	public double calAvgPayForPartTime() throws ArithmeticException{
		double sum = 0;
		int count = 0;
		for(int i = 0; i < nbEmp; i++)
			if(arrEmp[i] instanceof PartTimeEmp) {
				sum += arrEmp[i].calculatePay();
				count++;
			}
		if(count == 0) throw new ArithmeticException();
		return sum / count;
	}

}
```

Class test

```java
public class test {

	public static void main(String[] args) {
		PartTimeEmp e1 = new PartTimeEmp("Ahmad", 111, 6, 150);
		PartTimeEmp e2 = new PartTimeEmp("Omar", 222, 10, 200);
		PartTimeEmp e3 = new PartTimeEmp("Khalid", 333, 9, 150);
		FullTimeEmp e4 = new FullTimeEmp("Mohammed", 444, 5000);
		FullTimeEmp e5 = new FullTimeEmp("Ali", 555, 10000);

		try {
			Company c = new Company("KSU", 4);
			try {
				c.addEmployee(e1);
				System.out.println("Added 1 employee");
				c.addEmployee(e2);
				System.out.println("Added 2 employees");
				c.addEmployee(e3);
				System.out.println("Added 3 employees");
				c.addEmployee(e4);
				System.out.println("Added 4 employees");
				c.addEmployee(e5);
				System.out.println("Added 5 employees");
			} catch(IllegalStateException e) {
				System.out.println(e);
			}
			c.displayAll();
			try {
				c.deleteEmployee("Abdulrahman");
				System.out.println("Deleted successfully");
			} catch(IndexOutOfBoundsException e) {
				System.out.println(e);
			}

System.out.println("Yearly pay of mohammed: " + c.getYearlyPay("Mohammed"));
			try {
		System.out.println("Average pay for part time employees: " + 
c.calAvgPayForPartTime());
			}catch(ArithmeticException e) {
				System.out.println(e.getMessage());
			}

		} catch(NegativeArraySizeException e) {
			e.printStackTrace();
		}
		System.out.println("Bye");

	}
}
```