Evaluation Summary

compile	pass
tester	pass
Total:	100%

Compiling Main Program

pass

```
Compiling 1 source file to /var/folders/9k/9kZk9EnHFr8kECweINgp2++++TI/-Tmp-/asnmt532270usr1252182-submit-1258012530654
```

Testing Main Program

pass

```
1600
Expected: 1600
1200
Expected: 1200
1600
Expected: 1600
2200
Expected: 2200
```

Student Files

```
 1: public class HourlyWorker extends Worker
 2: {
 3:      /**
 4:       Constructs a new worker with a name and salary
 5:       @param aName the name
 6:       @param aSalary the salary
 7:      */
 8:     public HourlyWorker(String aName, int aSalary)
 9:     {
10:         super(aName, aSalary);
11:     }
12:
13:     /**
14:       Computes the pay
15:       @return pay
16:      */
17:     public int computePay(int hours)
18:     {
19:         int pay = 0;
20:         if (hours <= 40)
21:         {
22:             pay = hours * getSalary();
23:         }
24:         else
25:         {
26:             int overtime = hours - 40;
27:             pay = (int) ( (40 * getSalary()) + (getSalary() * 1.5 * overtime) );
28:         }
29:
30:         return pay;
31:     }
32:
33: }
```

```
 1: public class SalariedWorker extends Worker
 2: {
 3:     /**
 4:       Constructs a new worker with a name and salary
 5:       @param aName the name
 6:       @param aSalary the salary
 7:      */
 8:     public SalariedWorker(String aName, int aSalary)
 9:     {
```

```
10:         super(aName, aSalary);
11:     }
12:
13:     /**
14:       Computes the pay
15:       @return pay
16:      */
17:     public int computePay(int hours)
18:     {
19:         return 40 * getSalary();
20:     }
21:
22: }
```

```
 1: public class Worker
 2: {
 3:     /**
 4:       Constructs a new worker with a name and salary
 5:       @param aName the name
 6:       @param aSalary the salary
 7:      */
 8:     public Worker(String aName, int aSalary)
 9:     {
10:         name = aName;
11:         salary = aSalary;
12:     }
13:
14:     /**
15:       Returns the name of the worker
16:       @return name
17:      */
18:     public String getName()
19:     {
20:         return name;
21:     }
22:
23:     /**
24:       Returns the salary of the worker
25:       @return salary
26:      */
27:     public int getSalary()
28:     {
29:         return salary;
30:     }
31:
32:     /**
33:       Computes the pay
34:       @return pay
35:      */
36:     public int computePay(int hours)
37:     {
38:         return 0;
39:     }
40:
41:
42:     private String name;
43:     private int salary;
44: }
```

```
 1: /**
 2:    This class tests class Worker and its subclasses.
 3: */
 4: public class WorkerTester
 5: {
 6:    public static void main(String[] args)
 7:    {
 8:       Worker s = new SalariedWorker("Sally", 40);
 9:       Worker h = new HourlyWorker("Harry", 40);
10:       System.out.println(s.computePay(30));
11:       System.out.println("Expected: 1600");
12:       System.out.println(h.computePay(30));
13:       System.out.println("Expected: 1200");
14:       System.out.println(s.computePay(50));
15:       System.out.println("Expected: 1600");
```

```
16:       System.out.println(h.computePay(50));
17:       System.out.println("Expected: 2200");
18:    }
19: }
20:
```