```java
public abstract class Role { /17
	private String rolename;
	protected int nbOccur; ..............................0.5
	protected int totalTime; ..............................0.5

	private Award arrAwards[];..............................1
	private int nbAwards; ..............................1

	public Role(String name, int nb, int t) {
		rolename = name;
		nbOccur = nb;
		totalTime = t;

		arrAwards = new Award[10]; ..............................1
		nbAwards = 0; ..............................1
	}

	public Role(Role r) {
		rolename = r.rolename; ..............................0.5
		nbOccur = r.nbAwards; ..............................0.5
		totalTime = r.totalTime; ..............................0.5

		arrAwards = new Award[r.arrAwards.length]; ..............................1
		nbAwards = r.nbAwards; ..............................1

		for (int i=0; i < r.nbAwards; i ++) ..............................1
			arrAwards[i] = r.arrAwards[i]; ..............................1
	}

	public int gteNbOccur() {
		return nbOccur;
	}

	public abstract double calculateCachet();..............................1

	public double calculateTotalRevenue() throws Exception{..............................1

		double total = 0.0; ..............................0.5
		if (nbAwards == 0) throw new Exception();..............................1

		for (int i = 0; i < nbAwards; i++) ..............................1
			total += arrAwards[i].getCash();..............................1

		return total; ..............................1
	}
}
```

```
public class PrincipalRole extends Role /5 {.............................0.5
	private double basicSalary;

	public PrincipalRole(String name, int nb, int t, double bs) {

			super(name, nb, t); ...............................1
			basicSalary = bs; ................................0.5

	}

	public PrincipalRole(PrincipalRole r) {
		super(r); ................................1
		basicSalary = r.basicSalary; ..............................0.5
	}

	public double calculateCachet() {
		double res = basicSalary + nbOccur * 2000; ..............................1

		return res; ..............................0.5
	}

}
```

```
public class Movie {  /38
	private String name;
	private int year;
	private Role arrRoles[];.............................1
	private int nbRo; ..............................1

	public Movie(String s, int y, int size) {
		name = s;
		year = y;
		arrRoles = new Role[size]; ..............................1
		nbRo = 0; ..............................1
	}

	public boolean addRole(Role r) { /5
		if (nbRo < arrRoles.length) {..............................0.5
			if (r instanceof PrincipalRole) ..............................1
			arrRoles [nbRo] = new PrincipalRole( (PrincipalRole) r);.........1

			else
				arrRoles [nbRo] = new SecondaryRole( (SecondaryRole) r);.......1

			nbRo ++;..............................0.5

			return true; ..........................0.5

		}
		else
			return false; ..........................0.5

	}

	public int countPrincipalRoles(int nb) {/5
		int n = 0; ..............................1

		for (int i=0; i < nbRo; i++) ..............................1
			if (  arrRoles[i] instanceof PrincipalRole .....................1
				&& arrRoles[i].gteNbOccur() > nb) .........................1
				    n++;..............................0.5
		return n; ..............................0.5
	}

	public double getTotalCachetOfSecondaryRoles(double hr) throws Exception/8 {......1

		double res = 0.0; ..............................1
		SecondaryRole s;

		if (hr < 0) throw new Exception("Negative Argument");.....................1


		for (int i = 0; i < nbRo; i++) {..............................1
				if (arrRoles[i] instanceof SecondaryRole ) {.................1

					s = (SecondaryRole) arrRoles[i]; ..........................1
					if (s.getHourlyRate() == hr) ..............................1
						res += arrRoles[i].calculateCachet();.......0.5
				}
		}
		return res; .................0.5
	}
```

```
public Role[] getPrincipalRoles() { /8
            Role res[] = new Role[nbRo]; .........................1
            int j=0; .........................1

            for (int i = 0; i < nbRo; i++) {.........................1
                  try {.........................1
                        if (arrRoles[i] instanceof PrincipalRole && .....................1
                              arrRoles[i].calculateTotalRevenue() >
                              arrRoles[i].calculateCachet() ) {

                              res[j] = arrRoles[i]; .........................1
                              j++;.........................0.5
                        }
                  } catch (Exception e) {.........................1
                        System.out.println(e.getMessage());
                  }
            }

            return res; .........................0.5
      }

      public void splitAndsave(String pActFileName, String sActFileName, double c)
                  throws IOException { /8

            File f1 = new File(pActFileName); .........................0.5
            FileOutputStream fo1 = new FileOutputStream(f1); .........................0.5
            ObjectOutputStream pf = new ObjectOutputStream(fo1); .....................0.5

            File f2 = new File(sActFileName); .........................0.5
            FileOutputStream fo2 = new FileOutputStream(f2); .........................0.5
            ObjectOutputStream sf = new ObjectOutputStream(fo2); .....................0.5

            for (int i=0; i < nbRo; i++) {.........................1
                  if (  arrRoles[i] instanceof PrincipalRole .........................1
                        && arrRoles[i].calculateCachet() > c)
                              pf.writeObject(arrRoles[i]); .........................1
                  else
                        if (  arrRoles[i] instanceof SecondaryRole .....................1
                              && arrRoles[i].calculateCachet() < c)
                                          pf.writeObject(arrRoles[i]); .................1
            }
      }
}
```