Prénom :
Nom :

Soyez très précis dans vos réponses.

```
public class A1 {
  private double x;

  private double y;

  public A1(double x, double y) {
    this.x = x;
    this.y = y;
  }

  public A1 f(A1 that) {
    return new A1(x + that.y, y + that.x);
  }

  public A1 g() {
    return new A1(x + y, x - y);
  }

  public A1 h(A1 u, A1 v) {
    return new A1(x + u.x, y - v.y);
  }

  @Override
  public String toString() {
    return (x + y) + " " + (x - y);
  }
}

public class TestA1 {

  public static void main(String[] args) {
    A1 x = new A1(1,2);
    System.out.println(x);
    System.out.println(x.g());
    System.out.println(x.g().g());
    A1 y = new A1(3,-1);
    System.out.println(y);
    System.out.println(x.f(y));
    A1 z = new A1(0,0);
    System.out.println(z.h(new A1(1,2), new A1(3,4)));
    System.out.println(z);
  }

}
```

Indiquer l'affichage produit par le programme de gauche (méthode main)

Indiquer l'affichage produit par le programme de gauche (méthode main)

```
public class B1 {
  private int boo;

  private char X;

  public B1(int boo, char X) {
    this.boo = boo;
    this.X = X;
  }

  @Override
  public String toString() {
    if (boo > 0) {
      return X + " :-) " + boo;
    } else {
      return X + " :-( " + (-boo);
    }
  }

  public B1 flip() {
    return new B1(-boo, X);
  }

  public char poke(char Y) {
    char Z = X;
    X = Y;
    return Z;
  }

  public int zap(int gla) {
    boo = boo + gla;
    return boo / 2; // quotient de la division entière
  }
}

public class TestB1 {

  public static void main(String[] args) {
    B1 john = new B1(2, 'J');
    System.out.println(john);
    B1 bob = john.flip();
    System.out.println(john + " <-> " + bob);
    System.out.println(john.zap(1) + " <-> " + bob.zap(-2));
    System.out.println(john + " <-> " + bob);
    B1 robert = new B1(5, 'R');
    B1 alfred = robert;
    System.out.println(alfred.poke('A'));
    System.out.println(robert + " <-> " + alfred);
    B1 angie = new B1(-2,'A');
    System.out.println(angie.flip().zap(2));
    System.out.println(angie);
  }

}
```

```
public class C1 {
    private int x;

    public C1(int x) {
        this.x = x;
    }

    public void f() {
        à compléter (A)
    }

    à compléter (B)
}

public class TestC1 {

  public static void main(String[] args) {
    C1 foo = new C1(1);
    System.out.println(foo);
    for(int i=0;i<3;i++) {
      foo.f();
      System.out.println(foo);
    }
  }

}
```

Compléter la classe C1 afin que le programme (méthode main) affiche le texte suivant :

[1]
[2]
[3]
[4]

```
public class D1 {
  public int f() {
    return 2;
  }

  public int g() {
    return 3;
  }
}

public class E1 extends D1 {
  @Override
  public int f() {
    return 4;
  }
}

public class TestED1 {

  public static void main(String[] args) {
    E1 x = new E1();
    System.out.println(x.f());
    D1 y = new D1();
    System.out.println(y.f());
    y = x;
    System.out.println(x.g());
    Object o = x;
    System.out.println(o);
  }

}
```

Indiquer l'affichage produit par le programme de gauche (méthode main). ATTENTION le barème de l'exercice avant pondération est de 1 point par réponse juste et de -1 point par réponse fausse.

Un objet de la classe `Intervalle` représente un intervalle **fermé**. L'objectif de l'exercice est de compléter la classe de la façon suivante :

- la méthode `toString` représente l'intervalle sous la forme classique $[a, b]$;
- la méthode `contient` renvoie `true` si et seulement si `x` est contenu dans l'intervalle appelant ;
- la `longueur` de l'intervalle $[a, b]$ est $b - a$;
- la méthode `fixeMinimum` renvoie un nouvel intervalle avec la même borne supérieure que l'intervalle appelant et la borne inférieure passée en paramètre ;
- la méthode `étend` construit le plus petit intervalle contenant à la fois l'intervalle appelant et le paramètre `x`.

```
public class Intervalle {
    private double inf;

    private double sup;

    public Intervalle(double inf, double sup) {
        this.inf = inf;
        this.sup = sup;
    }

    @Override
    public String toString() {









    }

    public boolean contient(double x) {




    }

    public double longueur() {




    }

    public Intervalle fixeMinimum(double inf) {







    }

    public Intervalle étend(double x) {







    }
}
```