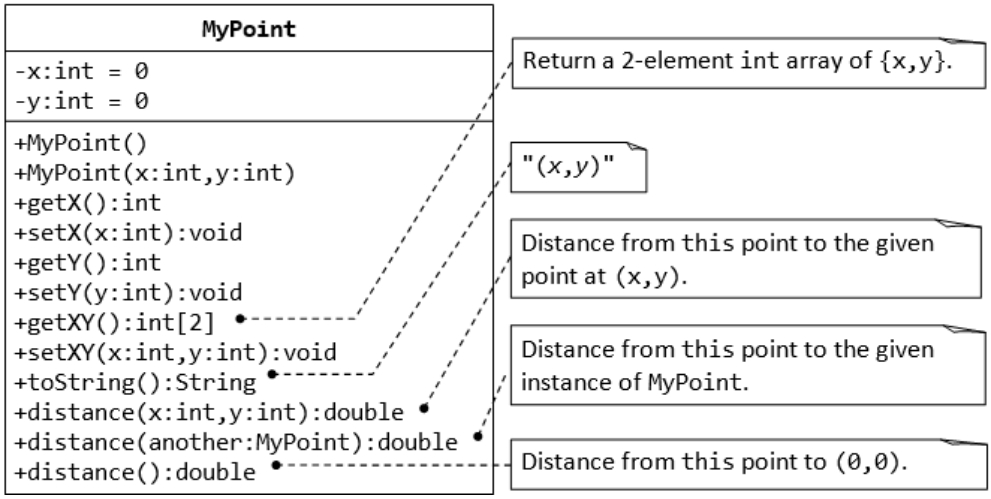
QUY ĐỊNH RIÊNG CHO THI THỰC HÀNH ONLINE:

** Sinh viên chỉ được phép sử dụng tài liệu giấy.
** **KHÔNG COPY** từ file có sẵn, mọi nội dung sinh viên phải tự gõ.
** ***Bài giống nhau sẽ bị 0 điểm (cả người chép và người cho chép).***
** Sinh viên phải nộp bài làm và file video quay màn hình, nếu thiếu 1 trong 2 thì coi như bài thi của sinh viên **KHÔNG** hợp lệ.

QUY ĐỊNH VỀ QUAY PHIM MÀN HÌNH CỦA SINH VIÊN:
1. Sinh viên phải quay phim lại toàn bộ thao tác trên màn hình từ lúc nhận đề đến khi được phép logout ra khỏi Zoom.
2. Thanh Taskbar phải để ở chế độ show 100% *(không được ẩn)* và phải hiển thị ngày giờ làm bài chính xác.
3. SV tự chọn sử dụng PM quay phim màn hình tùy ý và phải đảm bảo mục 1 và 2. SV cài đặt trước và luyện sử dụng nhiều lần cho quen. Chọn PM cho phép quay thời gian trên 60 phút.

CÁC DẠNG BÀI ÔN TẬP:

1. Cho mô hình lớp sau:



Mô tả lớp MyPoint:

Lớp MyPoint trừu tượng hóa cho một điểm 2D point với tọa độ x và y được thiết kế như trong sơ đồ lớp trên, bao gồm:

- Hai thuộc tính x (int) và y (int).
- **Constructor mặc định** khởi tạo một điểm có tọa độ mặc định là (0,0).
- **Constructor đầy đủ tham số** khởi tạo một điểm có tọa độ x, y.
- Các **getter** và **setter** cho x và y.
- Phương thức **setXY**() gán giá trị cho cả x và y.
- Phương thức **getXY**() trả về hai giá trị x, y trong mảng hai phần tử.
- Phương thức **toString**() trả về chuỗi chứa thông tin của một tọa độ có định dạng "(*x*, *y*)".
- Phương thức **distance**(int x, int y) trả về khoảng cách từ điểm hiện tại (*this*) đến một điểm khác có tọa độ *x, y*; ví dụ:

```
MyPoint p1 = new MyPoint(3, 4);
System.out.println(p1.distance(5, 6));
```

Công thức tính khoảng cách giữa 2 điểm: $d = \sqrt{(\Delta x)^2 + (\Delta y)^2}$, với $(\Delta x = x_A - x_B, \Delta y = y_A - y_B)$

- Phương thức **distance**(MyPoint another) trả về khoảng cách từ điểm hiện tại (*this*) đến một điểm khác (*another*); ví dụ:

```
MyPoint p1 = new MyPoint(3, 4);
MyPoint p2 = new MyPoint(5, 6);
System.out.println(p1.distance(p2));
```

- Phương thức **distance**() trả về khoảng cách từ điểm hiện tại (*this*) đến tọa độ *gốc* (0,0); ví dụ:

```
MyPoint p1 = new MyPoint(3, 4);
System.out.println(p1.distance());
```

Yêu cầu:

a. Cài đặt lớp MyPoint.
b. Cài đặt lớp TestMyPoint để thực thi và kiểm tra các phương thức của lớp MyPoint.

Gợi ý:

Lớp MyPoint:

```
1. // Overloading method distance()
2. // This version takes two ints as arguments
3. public double distance(int x, int y) {
4.    int xDiff = this.x – x;
5.    int yDiff = ......
6.    return Math.sqrt(xDiff*xDiff + yDiff*yDiff);
7. }
8.
9. // This version takes a MyPoint instance as argument
10.public double distance(MyPoint another) {
11.   int xDiff = this.x – another.x;
12.   .......
13.}
```

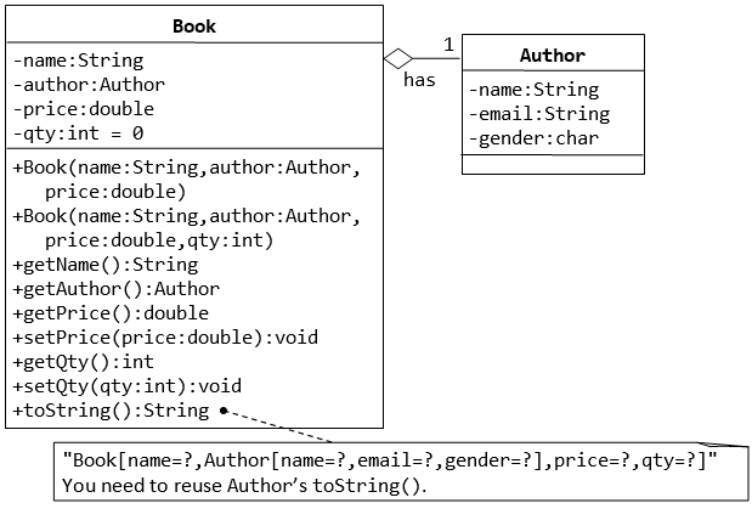
Lớp TestMyPoint:

```
// Test program to test all constructors and public methods
MyPoint p1 = new MyPoint();  // Test constructor
System.out.println(p1);      // Test toString()
p1.setX(8);   // Test setters
p1.setY(6);
System.out.println("x is: " + p1.getX());  // Test getters
System.out.println("y is: " + p1.getY());
p1.setXY(3, 0);   // Test setXY()
System.out.println(p1.getXY()[0]);  // Test getXY()
System.out.println(p1.getXY()[1]);
```

```
System.out.println(p1);

MyPoint p2 = new MyPoint(0, 4);  // Test another constructor
System.out.println(p2);
// Testing the overloaded methods distance()
System.out.println(p1.distance(p2));    // which version?
System.out.println(p2.distance(p1));    // which version?
System.out.println(p1.distance(5, 6));  // which version?
System.out.println(p1.distance());      // which version?
```

Bài tương tự: bài tập 6-Module 2

2. Cài đặt cho mô hình lớp sau và viết hàm main để test lớp Book:



Gợi ý viết hàm main:

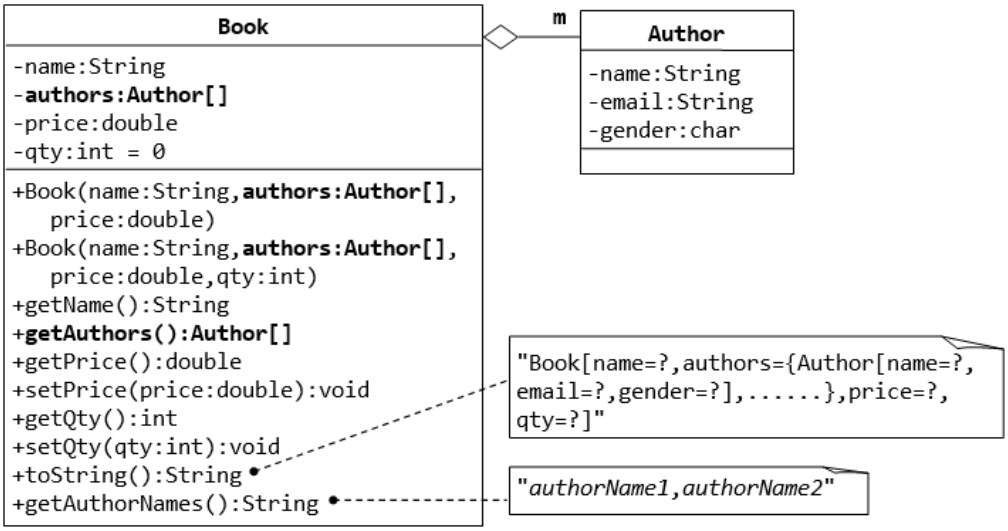
```
public class Test {
   public static void main(String[] args) {
      // Construct an author instance
      Author ahTeck = new Author("Tan Ah Teck", "ahteck@nowhere.com", 'm');
      System.out.println(ahTeck);
      // Author's toString()
      Book dummyBook = new Book("Java for dummy", ahTeck, 19.95, 99);
      // Test Book's Constructor
      System.out.println(dummyBook);
      // Test Book's toString()
      // Test Getters and Setters
      dummyBook.setPrice(29.95);
      dummyBook.setQty(28);
      System.out.println("New price is: " + dummyBook.getPrice());
      System.out.println("New qty is: " + dummyBook.getQty());
      System.out.println("Author is: " + dummyBook.getAuthor());
      // Author's toString()
      System.out.println("Author's name is: " + dummyBook.getAuthor().getName());
      System.out.println("Author's email is: " + dummyBook.getAuthor().getEmail());

      // Use an anonymous instance of Author to construct a Book instance
      Book anotherBook = new Book("more Java",
                  new Author("Paul Tan", "paul@somewhere.com", 'm'), 29.95);
      System.out.println(anotherBook); // toString()
   }
}
```

Kết quả thực thi:

```
Author[name=Tan Ah Teck, email=ahteck@nowhere.com, gender=m]
Book[name=Java for dummy, Author[name=Tan Ah Teck, email=ahteck@nowhere.com, gender=m], price=19.95, qty=99]
New price is: 29.95
New qty is: 28
Author is: Author[name=Tan Ah Teck, email=ahteck@nowhere.com, gender=m]
Author's name is: Tan Ah Teck
Author's email is: ahteck@nowhere.com
Book[name=more Java, Author[name=Paul Tan, email=paul@somewhere.com, gender=m], price=29.95, qty=0]
```

Bài tương tự: bài tập 7 - Module 2

3. Cài đặt cho mô hình lớp sau và viết hàm main để test lớp Book:

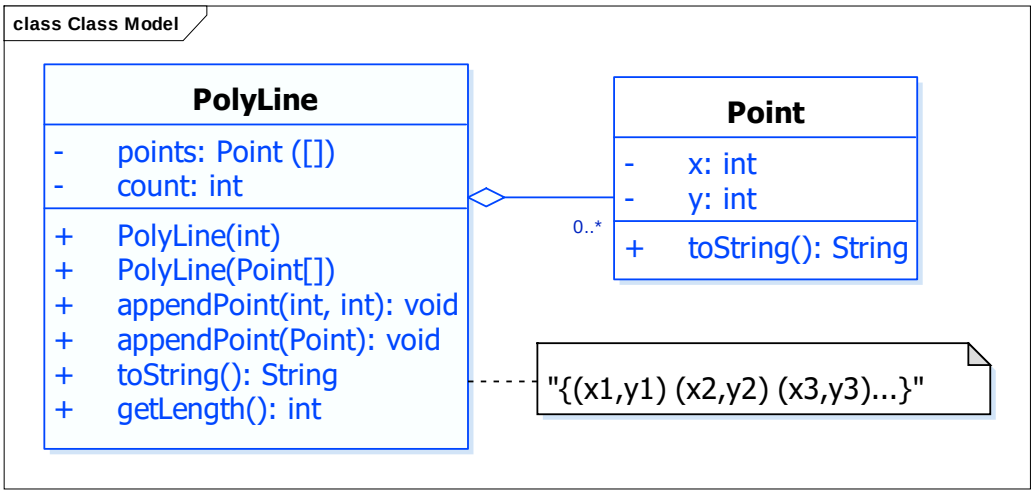


Gợi ý viết hàm main:

```
// Declare and allocate an array of Authors
Author[] authors = new Author[2];
authors[0] = new Author("AhTeck", "AhTeck@somewhere.com", 'm');
authors[1] = new Author("Paul Tan", "Paul@nowhere.com", 'm');
// Declare and allocate a Book instance
Book javaDummy = new Book("Java for Dummy", authors, 19.99, 99);
System.out.println(javaDummy); // toString()
```

Bài tương tự: bài tập 8, 13 - Module 2

4. Cài đặt cho mô hình lớp sau và viết hàm main để test lớp PolyLine:



Bài tương tự: bài tập 9, 10, 11, 12 - Module 2