Object Oriented Programming, Spring 2022 Midterm Exam Solution

1)

```
class Marks {
        private String id;
        private double quizMark, midMark, finalMark;
        public Marks(String id, double quizMark, double midMark, double finalMark) {
                this.id = id;
                this.quizMark = quizMark;
                this.midMark = midMark;
                this.finalMark = finalMark;
        }
        public void passedOrNot() {
                if(quizMark + midMark + finalMark >= 55) {
                        System.out.println("passed");
                }
                else {
                        System.out.println("failed");
                }
        }
}

public class Run {
        public static void main(String[] args) {
                Marks marks = new Marks("011250", 15, 20.25, 35.75);
                marks.passedOrNot();
        }
}
```

2)

```
class Base {
        public int a;
        private double b;
        Base(int a, double b) {
                this.a = a;
                this.b = b;
        }
        Base (Base ob) {
                a = ob.a;
                b = ob.b;
        }
        void show() {
                System.out.println("Sum of variables "
                                + "in the base class " + (a+b));
        }
        public double getB() { return b; }
        public void setB(double b) { this.b = b; }
}

class Override extends Base {
        int c;
        Override(int a, double b, int c) {
                super(a, b);
        this.c = c;
    }
    void show() {
        System.out.println("Sum of variables "
                + "in the base class " + (a + c + super.getB()));
    }
}

class Override2 extends Base {
    Override2(int a, double b) {
        super(a, b);
    }
    Override2(Override2 obj) {
        super(obj.a, obj.getB());
    }
    void show() {
        System.out.println("Sum of variables "
                + "in the base class " + (a + super.getB()));
    }
}

public class MethodOverride {
    public static void main(String[] args) {
        Override obj = new Override(10,  20.5, 12);
        obj.show();
        Override2 obj2 = new Override2(6, 10);
        obj2.show();
        Override2 obj3 = new Override2(obj2);
        obj3.show();
    }

}
```

3)

```
Output:
> $ Hello from P Class!
> $ Hello from C Class!
> $ Double value: 11.22
> $ Double value: 2.99
> $ Integer value: 10
> $ Double value: 3.145
```

4)

```
class Book {
    void description() {
        System.out.println("Books are awesome!");
    }
}

class PaperBackBook extends Book {
    void putPageMarker(int pageNo) {
        System.out.println("Page marker was "
                + "put on page no: " + pageNo);
    }
    // Dynamic method overriding
```

```java
        void description() {
                System.out.println("Paperback books are good for your eyes!");
        }
}

public class Test {
        public static void main(String[] args) {
                Book book1, book2;
                book1 = new Book();
                book2 = new PaperBackBook();

                book1.description();
                book2.description();
                // Casting and parameter
                ((PaperBackBook)book2).putPageMarker(50);
        }
}
```

4)

```java
abstract class GeometricShape {
        double PI = 3.14;
        abstract double volume();
}

class Sphere extends GeometricShape {
        private double radius;
        public Sphere(double radius) {
                this.radius = radius;
        }
        public double volume() {
                return (4.0/3 * PI * Math.pow(radius, 3));
        }
}

class Cylinder extends GeometricShape {
        private double radius, height;
        public Cylinder(double radius, double height) {
                this.radius = radius;
                this.height = height;
        }
        public double volume() {
                return (PI * Math.pow(radius, 2) * height);
        }
}
```

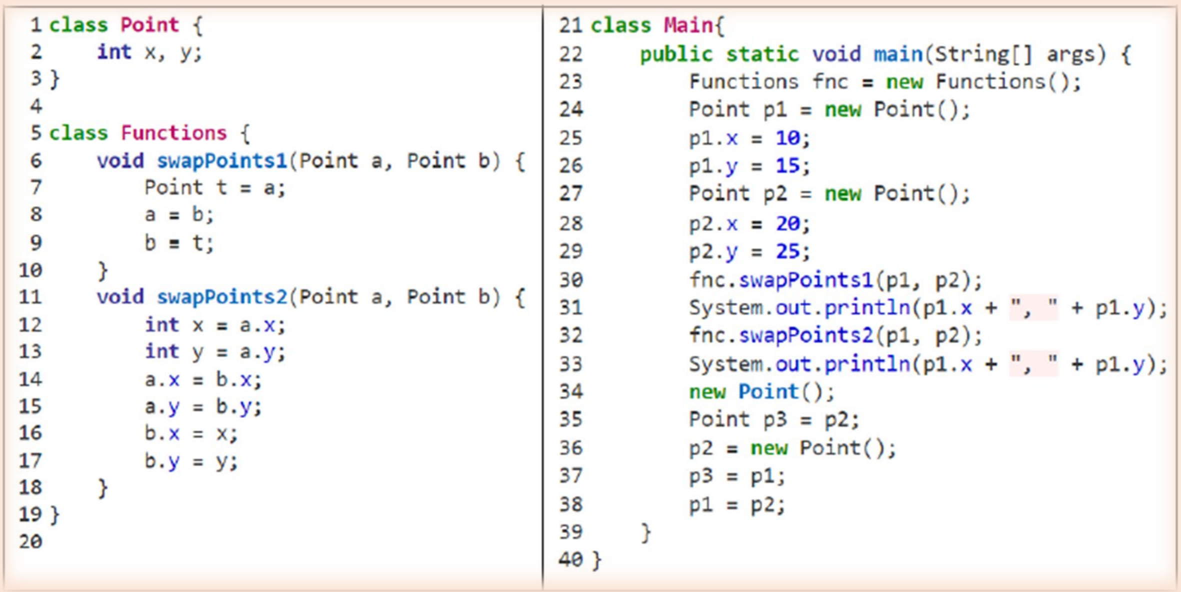
5a)

```java
package pack1;

public class A {
        public int x;
}

package pack2;
import pack1.*;
```

```
public class B {
	public static void main(String[] args) {
		A ob = new A();
		ob.x = 100;
	}
}
```

5b)

```
Output:
> $ 10, 15
> $ 20, 25
```



```
1 class Point {
2     int x, y;
3 }
4
5 class Functions {
6     void swapPoints1(Point a, Point b) {
7         Point t = a;
8         a = b;
9         b = t;
10    }
11    void swapPoints2(Point a, Point b) {
12        int x = a.x;
13        int y = a.y;
14        a.x = b.x;
15        a.y = b.y;
16        b.x = x;
17        b.y = y;
18    }
19 }
20
```

```
21 class Main{
22     public static void main(String[] args) {
23         Functions fnc = new Functions();
24         Point p1 = new Point();
25         p1.x = 10;
26         p1.y = 15;
27         Point p2 = new Point();
28         p2.x = 20;
29         p2.y = 25;
30         fnc.swapPoints1(p1, p2);
31         System.out.println(p1.x + ", " + p1.y);
32         fnc.swapPoints2(p1, p2);
33         System.out.println(p1.x + ", " + p1.y);
34         new Point();
35         Point p3 = p2;
36         p2 = new Point();
37         p3 = p1;
38         p1 = p2;
39     }
40 }
```

- **Line 34, no reference variable**
- **Line 37, p3 containing the reference of the original p2 is reassigned**