

ALL B's BELOW!

1a.), 2a.)

```
using namespace std;
```

```
#include <iostream>
```

```
#include <fstream>
```

```
void valuepasser(int x, int y)
```

```
{
    int z;
    z = x;
    x = y;
    y = z;

    cout << "swapped " << x << " " << y << endl;
}
```

```
void referencepasser(int &x, int &y)
```

```
{
    int z;
    z = x;
    x = y;
    y = z;

    cout << "swapped " << x << " " << y << endl;
}
```

```
int main()
```

```
{
    int a = 5;
    int b = 1;

    valuepasser(a, b);
    cout << a << " " << b << endl;

    int w = 1;
    int i = 2;

    referencepasser(w, i);
    cout << w << " " << i << endl;
}
```

```

gfors@thomas:~$ g++ passing.cpp
gfors@thomas:~$ a.out
swapped 1 5
5 1
swapped 2 1
2 1
gfors@thomas:~$

```

1st uses valuepasser function, second uses referencepasser function.

3a.)

```
import java.awt.Point;
```

```

public class passing{
    public static void main(String[] args) {
        // System.out.println("Hello, World!");
        int x = 1;
        int y = 2;
        int a = 3;
        int b = 4;
        Point p1 = new Point(x, y);
        Point p2 = new Point(a, b);
        System.out.println("before swap");
        System.out.println("(" + p1.x + ", " + p1.y + ")");
        System.out.println("(" + p2.x + ", " + p2.y + ")");
        System.out.println("swapping!");
        swapper(p1, p2);
        System.out.println("after swap");
        System.out.println("(" + p1.x + ", " + p1.y + ")");
        System.out.println("(" + p2.x + ", " + p2.y + ")");
    }
}

```

```

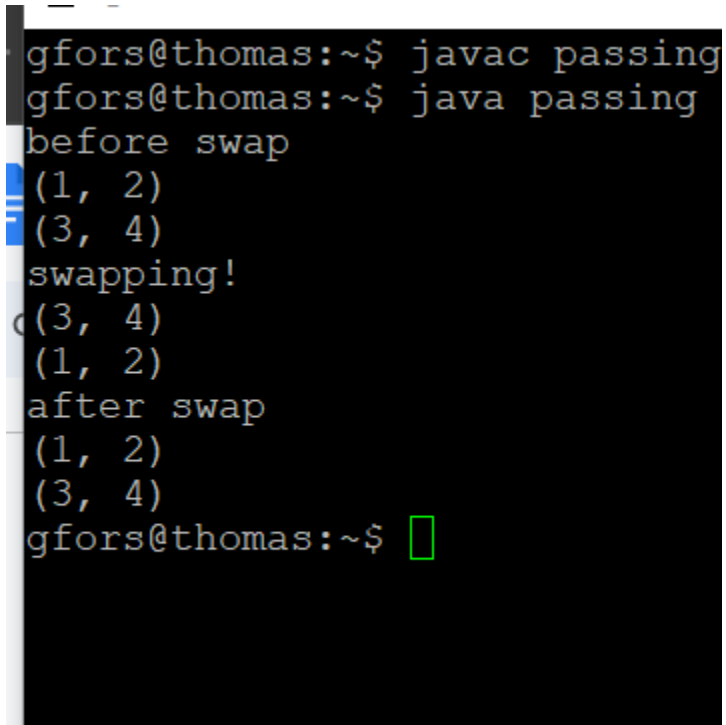
public static void swapper(Point p1, Point p2)
{
    Point temp = new Point(0, 0);
    temp = p1;
    p1 = p2;
    p2 = temp;
    System.out.println("(" + p1.x + ", " + p1.y + ")");
}

```

```

        System.out.println("(" + p2.x + ", " + p2.y + ")");
    }
}

```



```

gfors@thomas:~$ javac passing
gfors@thomas:~$ java passing
before swap
(1, 2)
(3, 4)
swapping!
(3, 4)
(1, 2)
after swap
(1, 2)
(3, 4)
gfors@thomas:~$ 

```

4a.)

```
import java.awt.Point;
```

```

public class passing{
    public static void main(String[] args) {
        // System.out.println("Hello, World!");
        int x = 1;
        int y = 2;
        int a = 3;
        int b = 4;
        Point p1 = new Point(x, y);
        Point p2 = new Point(a, b);
        System.out.println("swapping function");
        swapper(p1, p2);
        System.out.println("after swapping function in main");
        System.out.println("(" + p1.x + ", " + p1.y + ")");
    }
}

```

```

        System.out.println("(" + p2.x + ", " + p2.y + ")");
    }

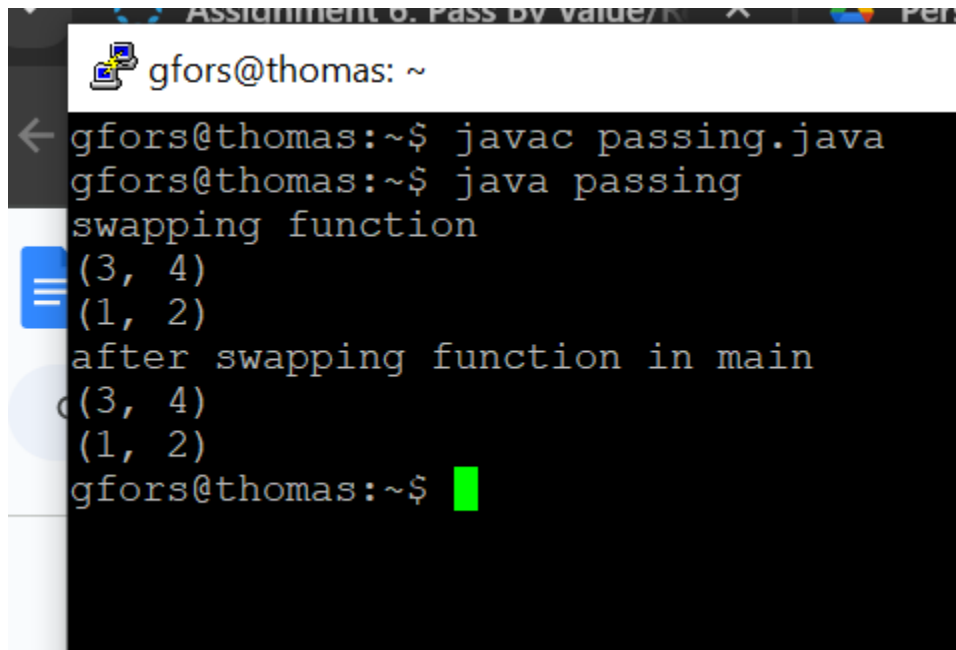
```

```

public static void swapper(Point p1, Point p2)
{
    Point temp = new Point(0, 0);
    temp.x = p1.x;
    temp.y = p1.y;
    p1.x = p2.x;
    p1.y = p2.y;
    p2.x = temp.x;
    p2.y = temp.y;

    System.out.println("(" + p1.x + ", " + p1.y + ")");
    System.out.println("(" + p2.x + ", " + p2.y + ")");
}
}

```



```

gfors@thomas: ~
gfors@thomas:~$ javac passing.java
gfors@thomas:~$ java passing
swapping function
(3, 4)
(1, 2)
after swapping function in main
(3, 4)
(1, 2)
gfors@thomas:~$

```

1b.), 2b.) 3b.), 4b.)

16, 20

in 1a, we are just swapping the values

120 5, 1 go to method  
in method, they are swapped  
but x & y remain the same in main

Passing value  
x = 5  
y = 1  
a = 5  
b = 1  
copies of x & y  
a = 1  
b = 5  
No change  
swapping values

in 2a, we pass the ~~object~~ variables themselves w/ a pointer

Passing variable  
by reference  
x = 5  
y = 1  
x = 5  
y = 1  
x = 1  
y = 5  
Swapping values of variables

7b  
Point P1  
Point P2  
Point a  
Point b  
Point a = b  
Point b = a  
in main, P1 & P2 are the same.  
we are just moving letters in the net

Point P1  
Point P2  
Point P1, x, y = P2, x, y  
Point P2, x, y = P1, x, y  
in main, P1 & P2 change, because we are swapping the actual values

5a.)

Java is almost always pass by value. It works similar to how pass by value works in c++; the caller of a method passes a copy of the variables; you aren't working with the variables themselves in the method. Non-primitive types, such as arrays, work differently. You can edit them in the method in java by passing them by value, and the changes reflect in main. This is not the case in c++.

5b.)

There are no pointers in java, so pass by reference works differently than c++. One way to do it, and the way it was done in this program, is to pass the object reference as a value, and then change that value through the object's methods (in this case, p1.x and p1.y, which are both in the point 'library'). C++ also allows you to dereference a pointer, which is not present in java.