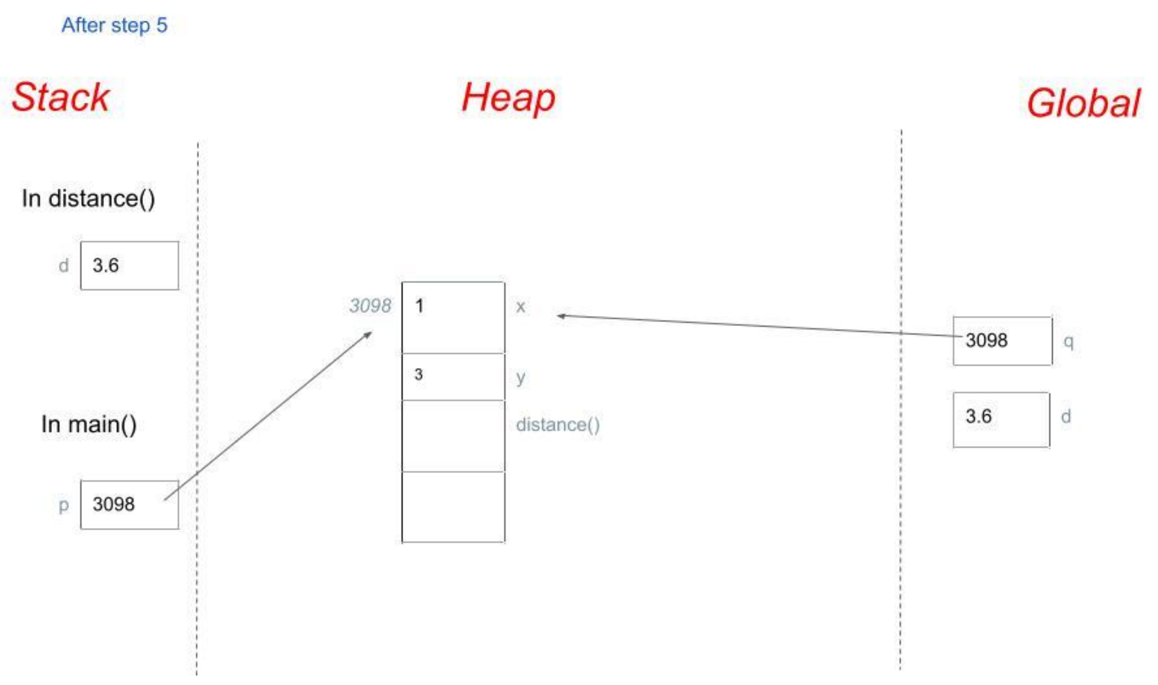
Dmitri Stanchevici
Unit 4
Module 5

Ex. 5.2

Printed is
3.605551275463989
XYPoint@7ad041f3

Ex. 5.3

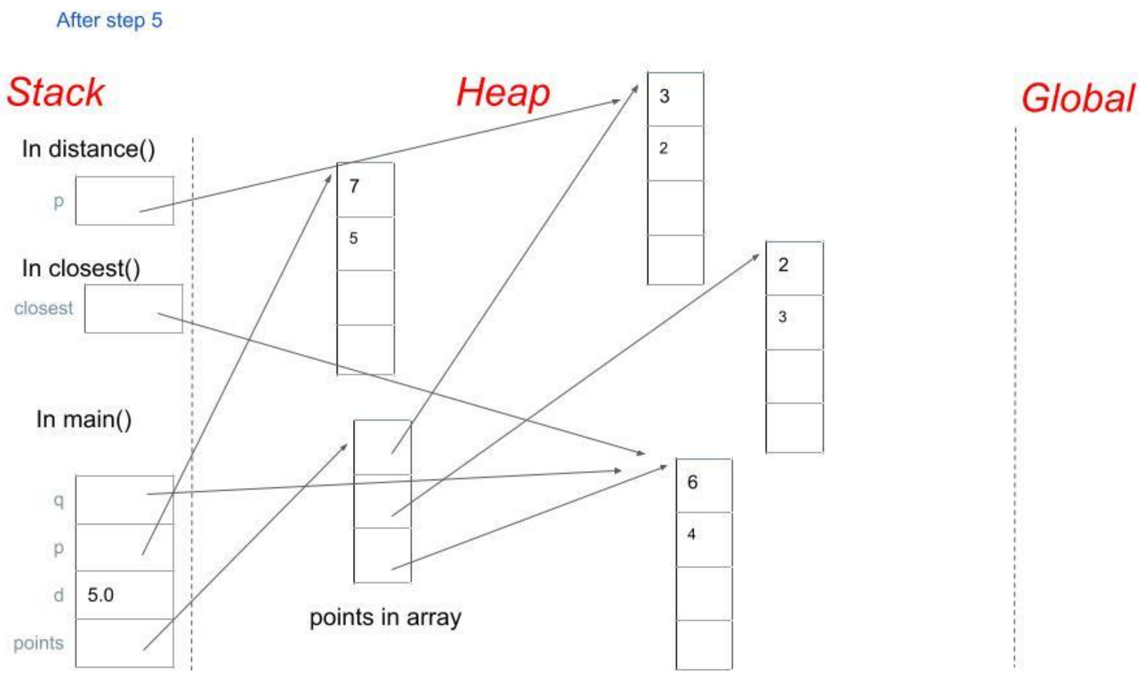


Printed is

```
3.605551275463989
1.0
1.0
```

When **q.x = 1** is executed, the variable **x** inside the instance pointed to by both **q** and **p** is changed to 1. Because **q** and **p** point to the same instance, 1 is accessible through both **p.x** and **q.x**.

Ex. 5.5



The output on my computers includes:

```
In main(): p=XYPoint@7ad041f3
In distance(): p=XYPoint@7344699f
```

The difference in the two addresses occurs because even though these two object variables have the same name, **p,** they appear inside two different classes and they are thus out of scope (out of reach) to each other. (If you allow me a pun: These two p's are from different pods.)

p in main() is a local variable pointing to an instance created by XYPoint p = new XYPoint (); in step 3.

p in distance() inside the XYPoint class is a parameter. The argument passed into this parameter from main() happens to be the address at **points[0].** So, here p == points[0] because they contain the same address (they point to the same instance).

Ex. 5.7

XYPoint r = new XYPoint ();

The above line throws a compile error:

```
ObjectExample4.java:44: error: constructor XYPoint in class XYPoint cannot be applied to given
types;
        XYPoint r = new XYPoint ();
                    ^
 required: double,double
 found:    no arguments
 reason: actual and formal argument lists differ in length
```

EXERCISES IN API SECTION

Ex. on String methods

Printed are

```
e
10
0
```