Summary 4 – Variables: Assignment, Primitive type vs. Object type

- What is this?

The ***primitive types*** in Java are: int byte short long double float char boolean

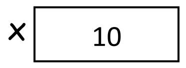
true or false

All other types are ***object type*** and are defined by a ***class***, e.g. Point, JFrame, Eye.

For character literals, surround them in single quotes, e.g. `'y'` or `'\n'`

Variables of primitive type contain their actual numeric value:

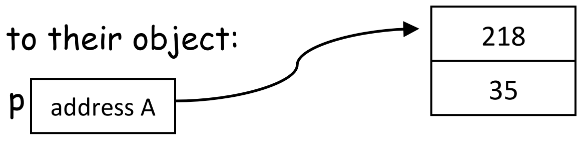
int x = 10;



In this example, x is 32 bits (for an *int*) that represent the decimal number 10.

Variables of object type contain a *reference* to their object:

Point p = new Point(218, 35);

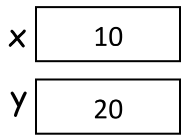


In this example, p is 32 bits that are the address in memory where the Point is stored.
The point itself is a pair of 32-bit chunks (for int's) that represent the numbers 218 and 35.
Thus p refers to that object, that is, to that pair of integers.

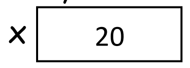
- Example

int x = 10;

int y = 20;



x = y; The bits in y are copied into x, so that x now looks like

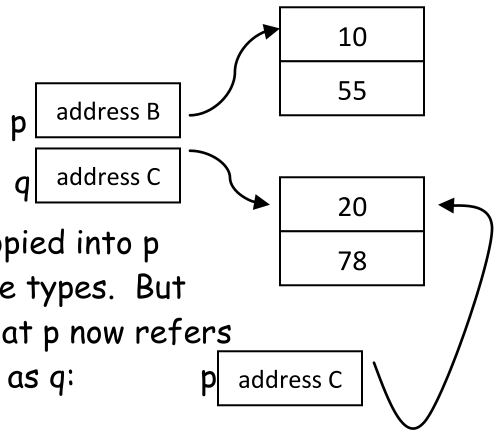


System.out.println(x); Prints 20.

Point p = new Point(10, 55);

Point q = new Point(20, 78);

p = q; The bits in q are copied into p just as for primitive types. But here that means that p now refers to the same object as q:



p.setX(33);

System.out.println(q.getX()); Prints 33.

- For further study:
  - *Big Java*, section 4.1 Number Types, describes the primitive types and their sizes.
  - *Big Java,* section 2.10 Object References, describes how variables of object type are references
  - *Authors* of this summary: David Mutchler.
  - See also the Summaries on *Variables: Fields vs. Parameters vs. Local Variables*

What do you think happens in this example to the bits at address B after the assignment p = q? Answer: they are *garbage-collected*.