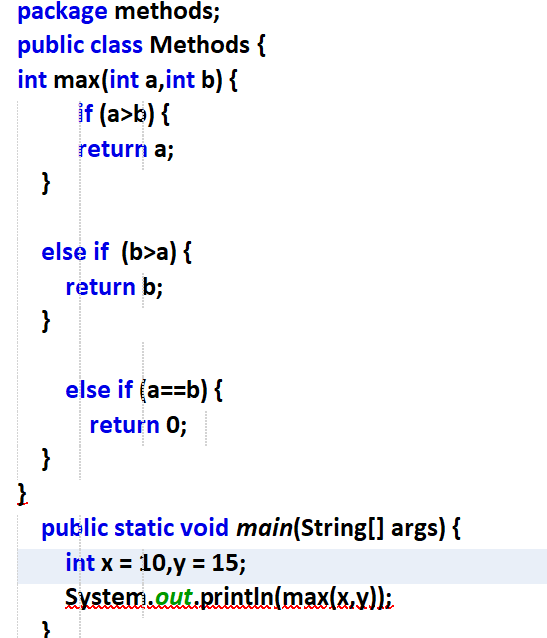
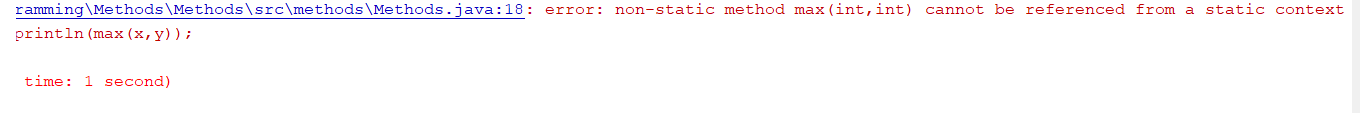
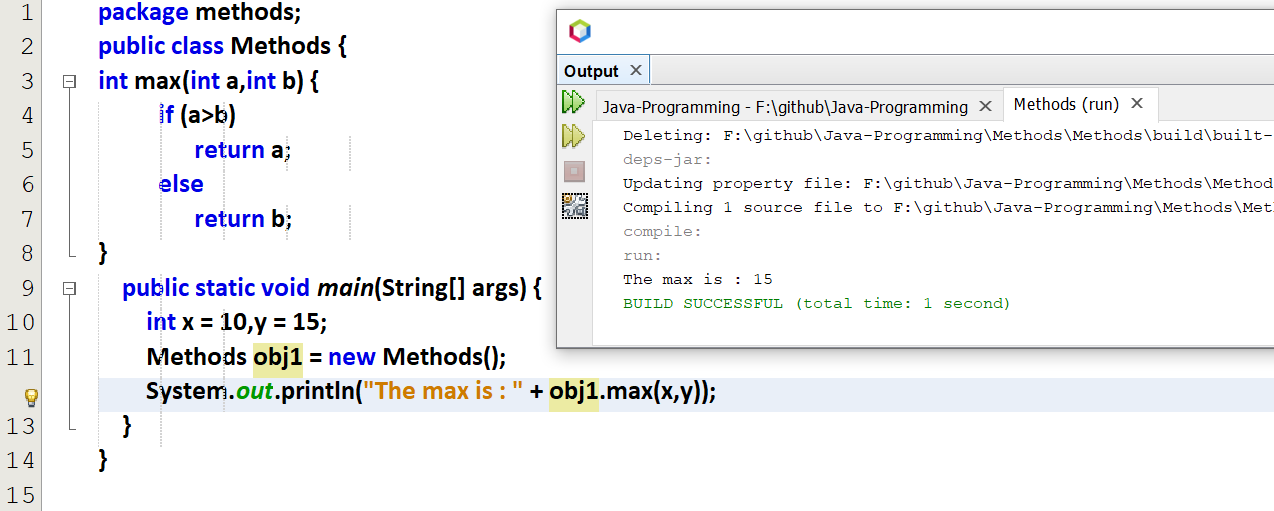
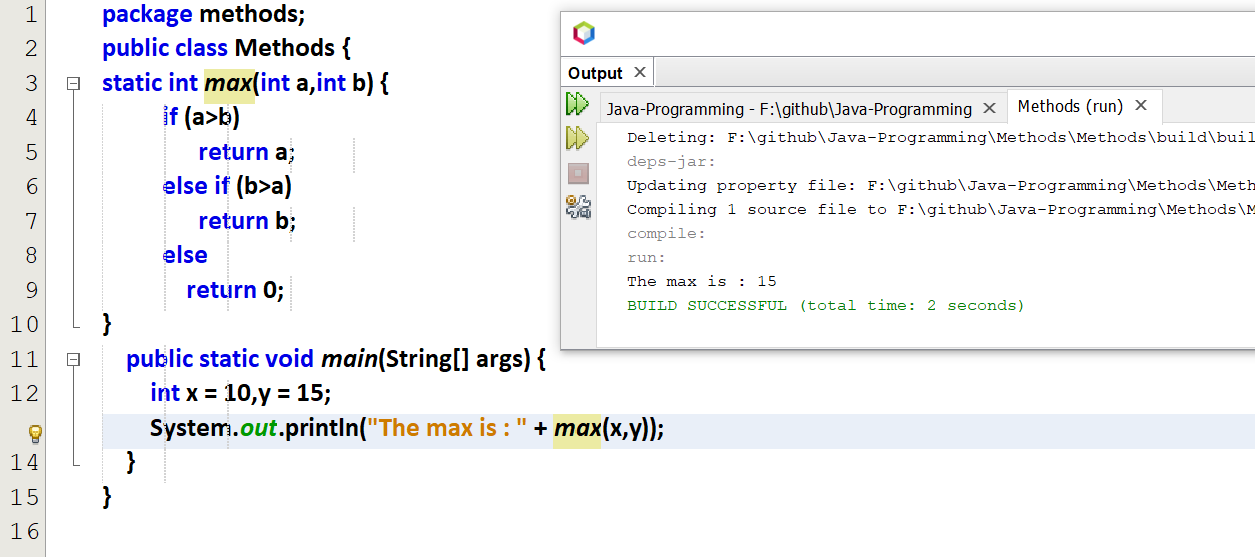
**return type**Java 🡪 Signature/header of a method  
C 🡪 prototype of a method



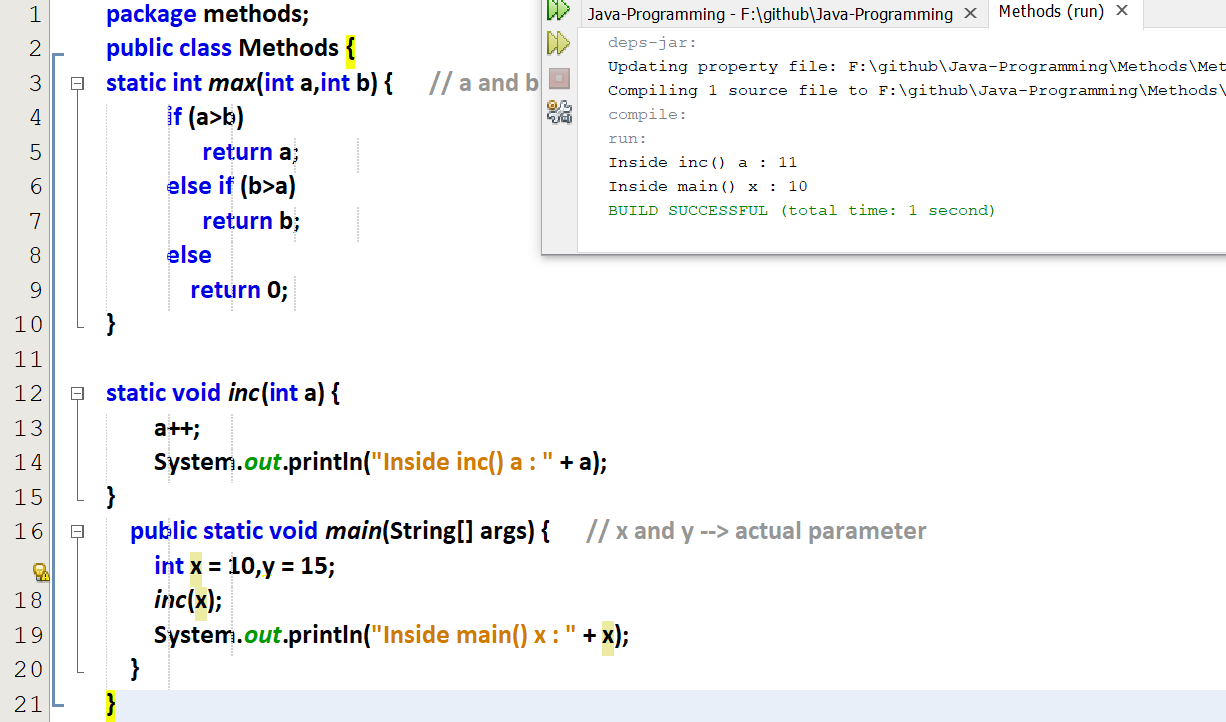


1. Create an object for method class and then call.
2. Main method is static, So make max also static.

Object function call  


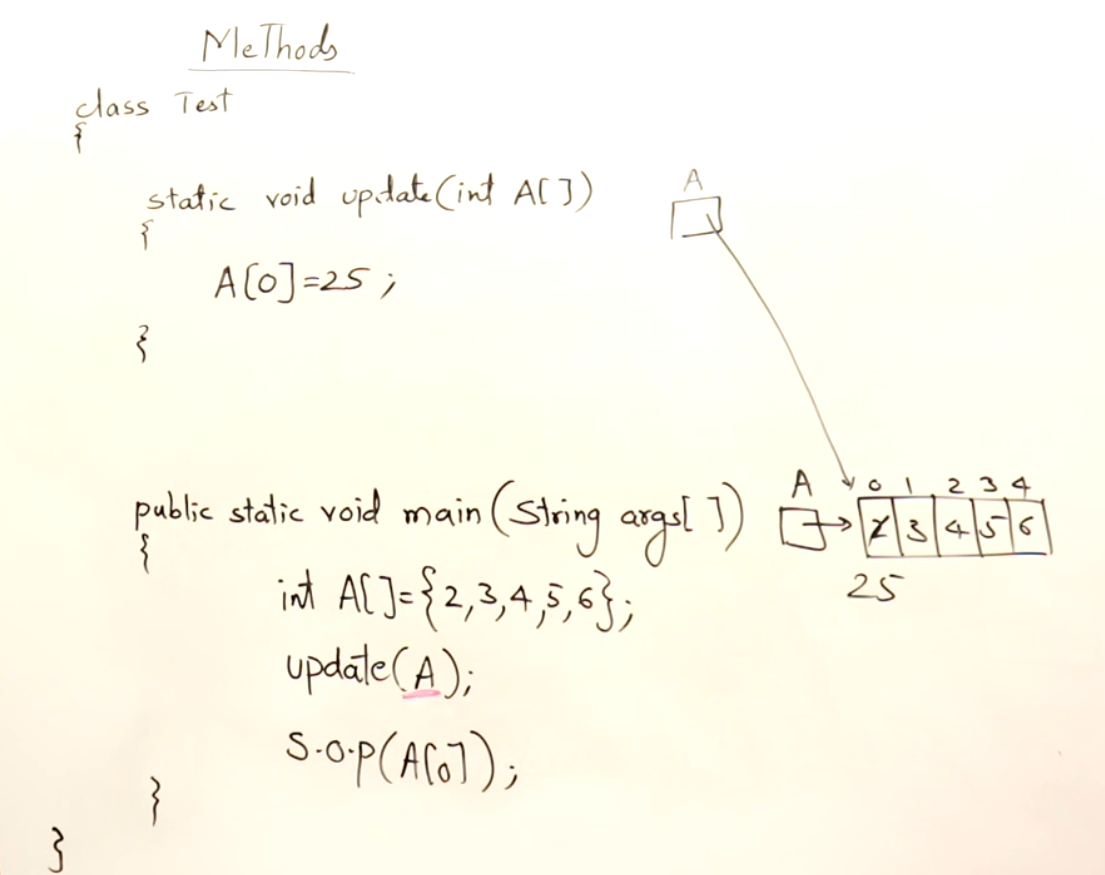
Using static functions  


Actual and formal parameters

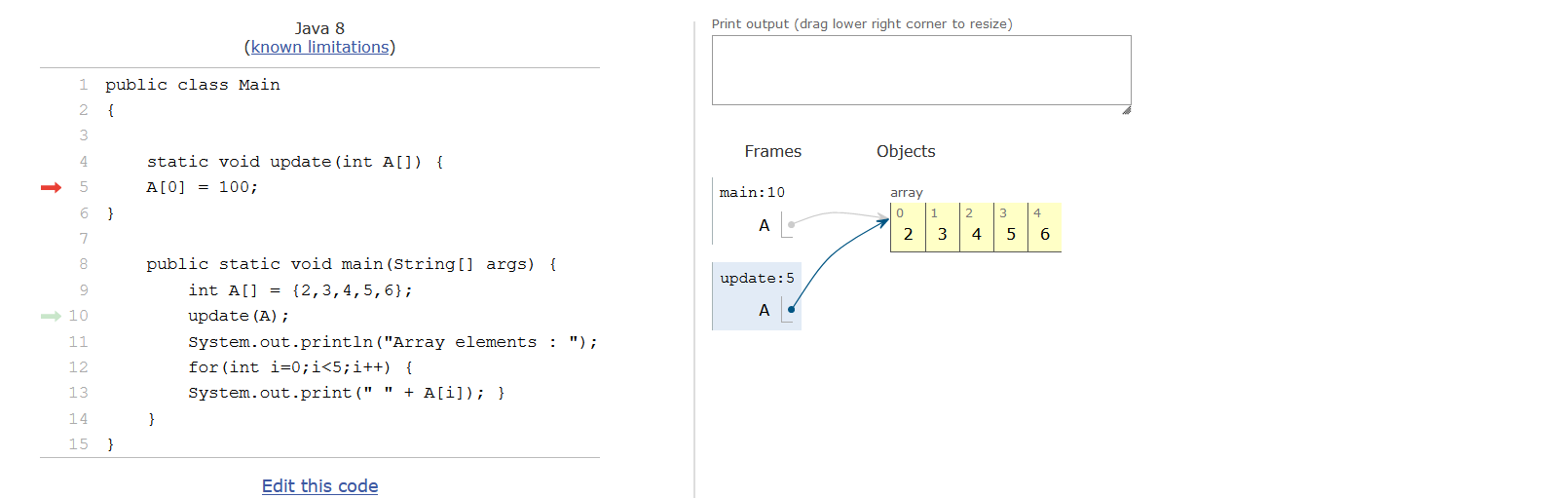
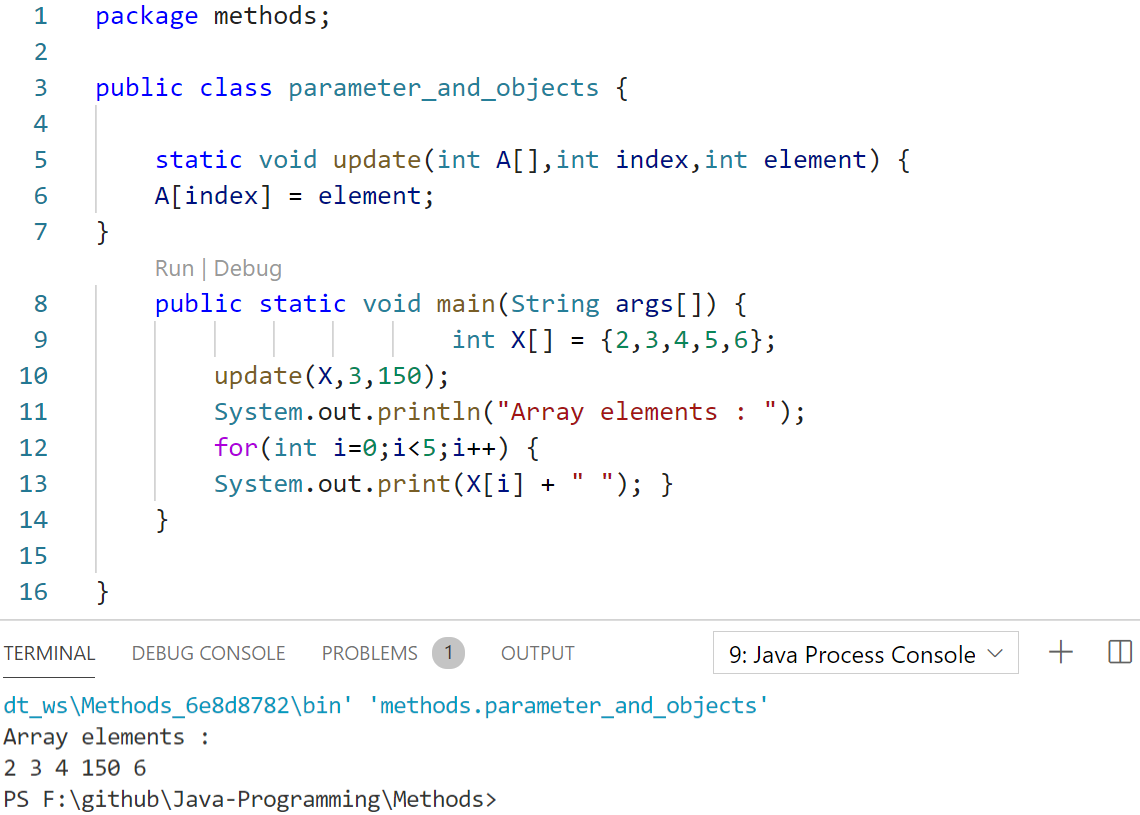
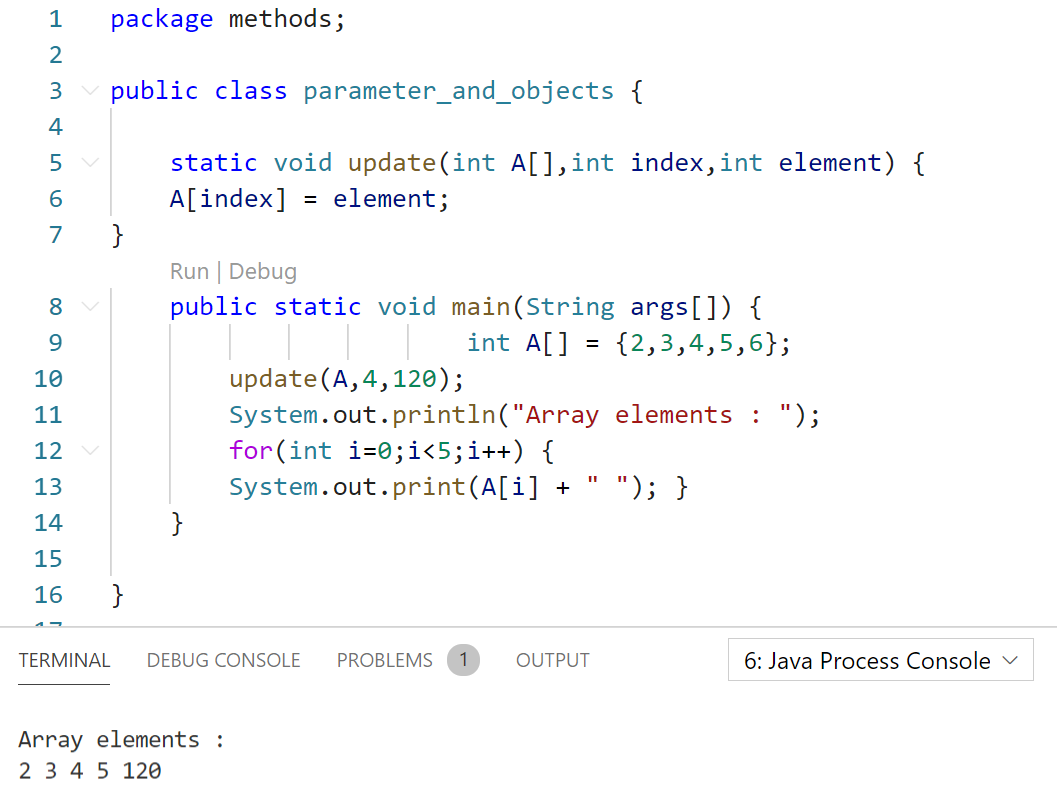


Any change in the formal parameter will not affect actual parameters, since it is a call by value and not call by reference.

Passing object and primitives as parameters



Passing objects (here passing arrays)

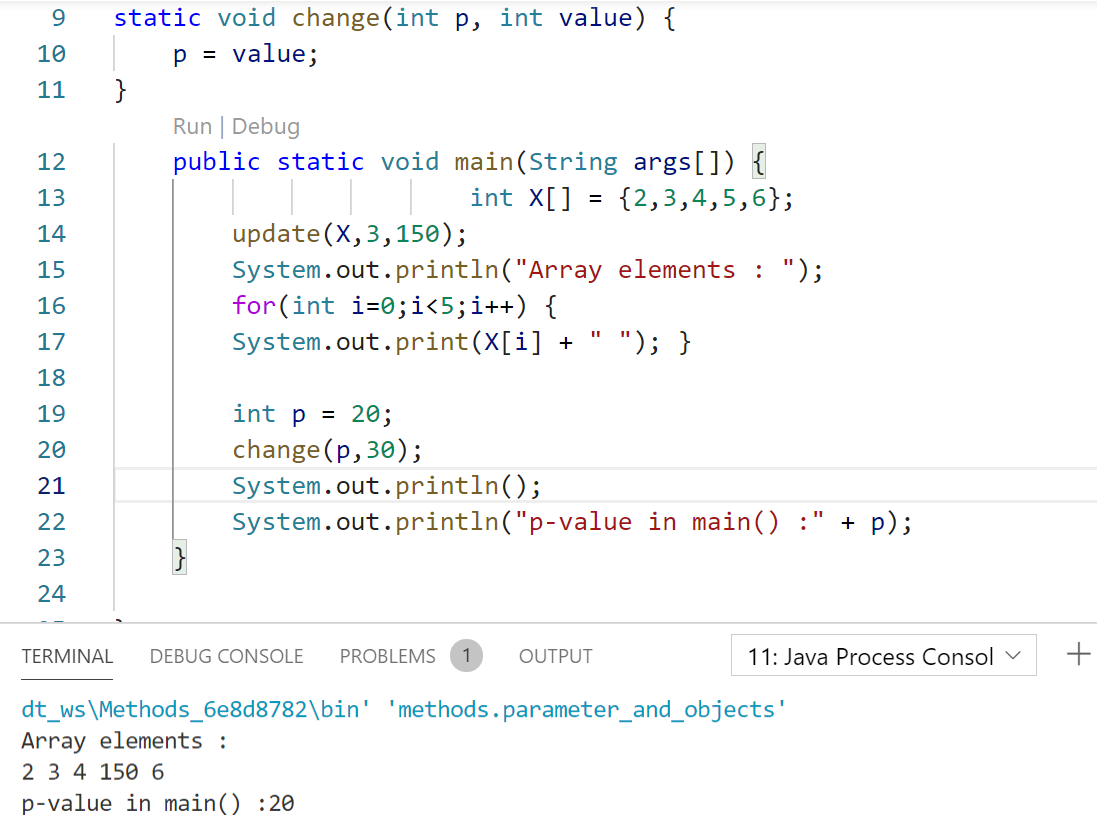
  




Array elements will be inside heap, and its reference will be in stack.  
When objects are passed as parameters, object itself is not passed, only the reference is passed.

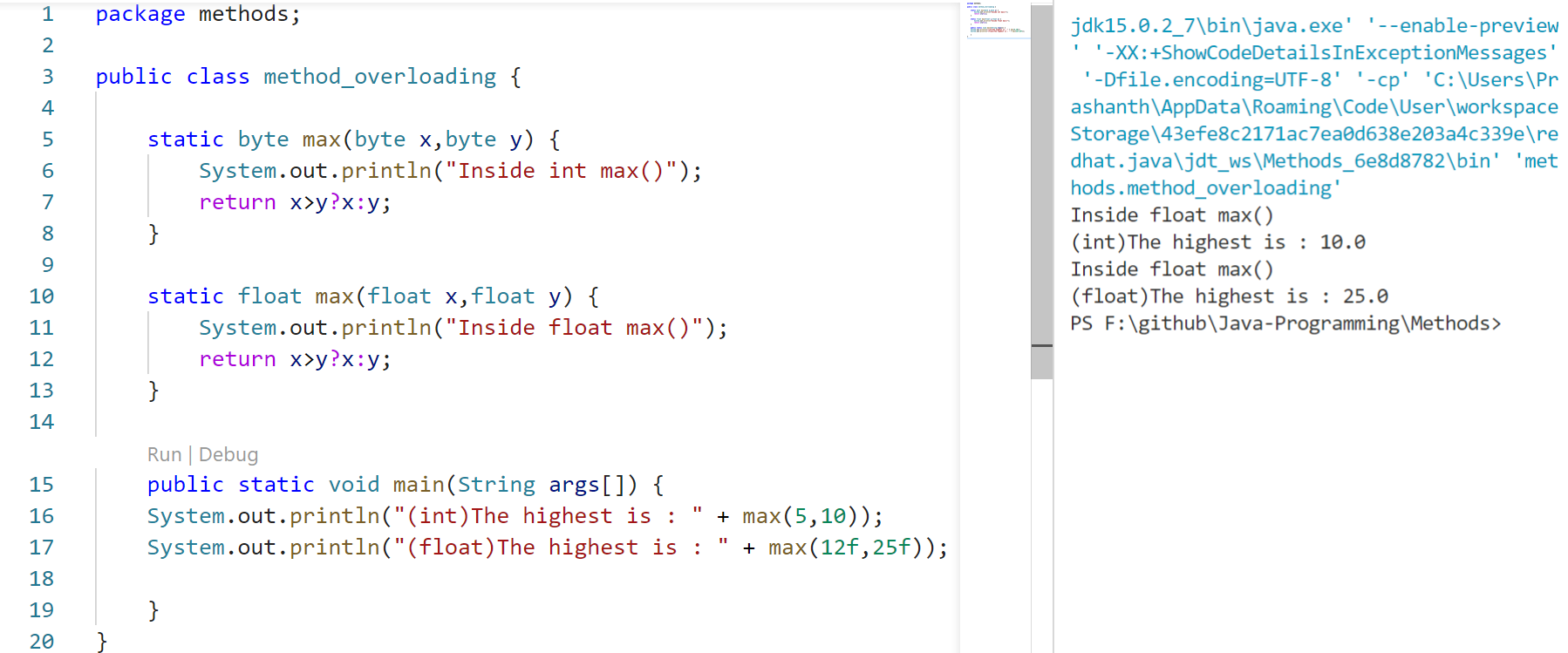
So array is an object.

Different Names in actual and formal parameters also, the array object is referenced.

Passing primitives in function call  


Parameter passing

Method Overloading



byte<int<float<double. So smallest one is byte.  
Even-though int is passed, float is accepting since byte will not support int but float will support int. 🡪 **Polymorphism**



byte will support int also without explicitly declaring

