The CSE home VM on my Windows desktop

Model Name: Intel(R) Core(TM) i7-6700K CPU @ 4.00GHz

Array Size	Performing src assignment?	App	Time with I then j	Time with j then i
2048	No	Java	0.15	0.96
2048	No	JavaInteger	0.16	0.99
2048	No	C	0.09	0.54
2048	No	Optimized C	0.01	0.55
Array Size	Performing src assignment?	App	Time with I then j	Time with j then i
2048	Yes	Java	0.15	1.10
2048	Yes	JavaInteger	2.33	7.90
2048	Yes	C	0.10	0.91
2048	Yes	Optimized C	0.03	0.75
Array Size	Performing src assignment?	App	Time with I then j	Time with j then i
4096	No	Java	0.21	4.86
4096	No	JavaInteger	0.39	5.11
4096	No	C	0.44	2.44
4096	No	Optimized C	0.06	2.25
Array Size	Performing src assignment?	App	Time with I then j	Time with j then i
4096	Yes	Java	0.25	5.13
4096	Yes	JavaInteger	10.76	24.20
4096	Yes	C	0.41	3.50
4096	Yes	Optimized C	0.19	3.00

1. The difference between the two java implementations is that cacheExperiment.java uses ints, a four-byte datatype, whereas cacheExperimentInteger.java uses the Integer object, which is much larger than an int.
2. The most surprising difference in a pair of results for me was the change from using i then j versus j then i with the 2048 java code with Integers and src. The jump from just over 2 seconds to 7.5 seconds really shows the vast difference in performance something so simple can have.
3. If these programs truly don’t do anything, then yes, that would be a good optimization. One purpose of them, however, may be to fill the cache, clearing out previous data.