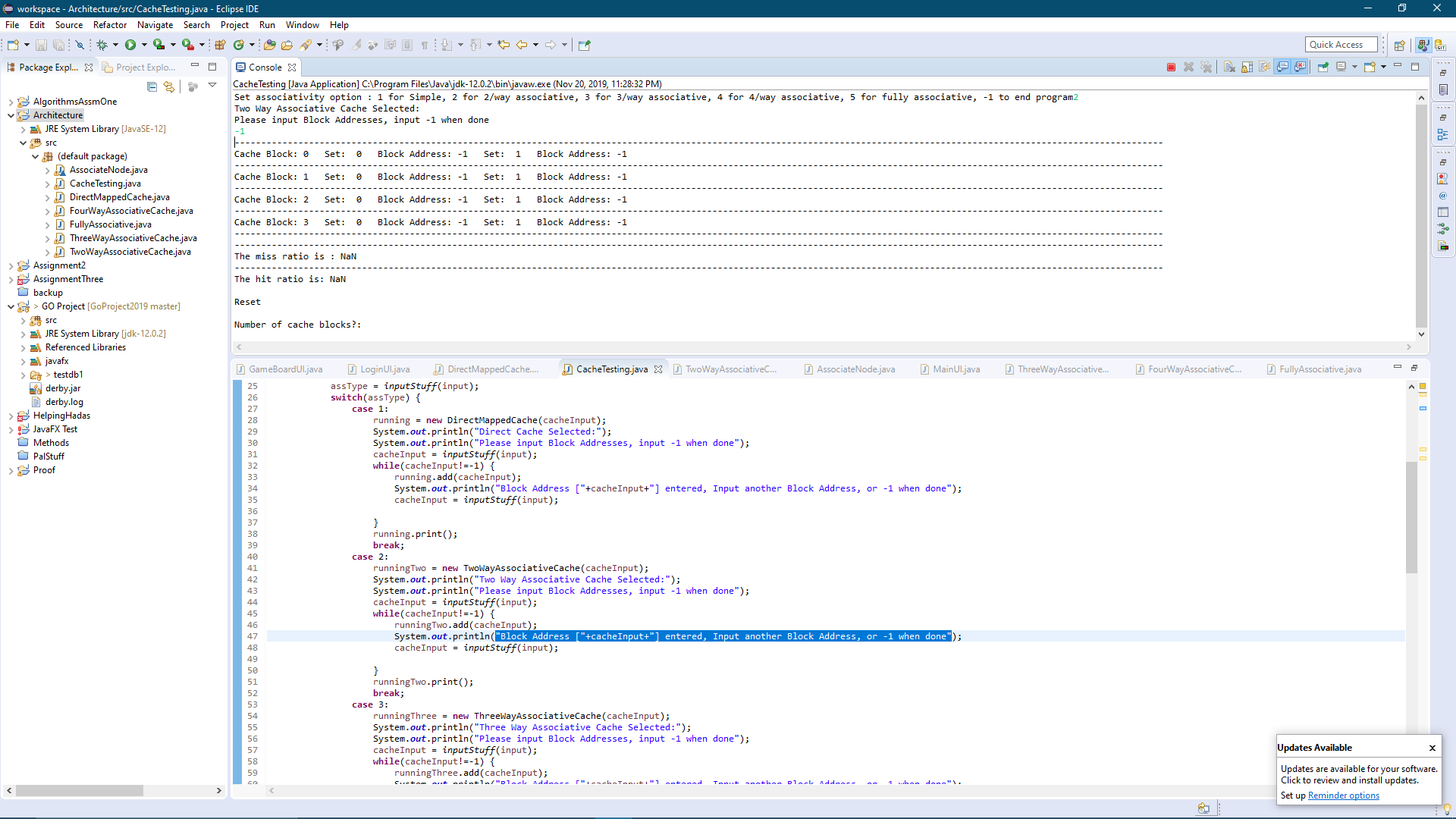
Read Me:

How to Run Program:

1. Install Eclipse and Java SDK 12
2. Uzip ProgramSrcFiles
3. Create New Project in Eclipse
4. Copy Unziped source files to Project and click run.

About Program:

Program Prints "Number of cache blocks?: " Asking the question: How Many Block Address locations does this cache have. The Block Address entered by user will be modulus divided by this number for direct, two-way, and four-way associative. Cache **memory blocks are initialized with -1 to symbolize null or “nothing” there**, this acts as a “dirty” bit. "Set associativity option : 1 for Simple, 2 for 2/way associative, 3 for 3/way associative, 4 for 4/way associative, 5 for fully associative, -1 to end program" is printed next. If the user enters a character by mistake, the program will perform exception handling and give the user another chance to enter. **If the user enters a number besides those list here, the program will default to fully associative.** Next program will display the selected cache method: eg "Two Way Associative Cache Selected:" . The program will then ask the user for a series of inputs to be terminated with -1. Eg "Block Address ["+cacheInput+"] entered, Input another Block Address, or -1 when done" . Valid inputs will be displayed so that the user may confirm them, invalid input will prompt the program to ask for a valid input. An input of “-1” will terminate the looping query of inputs. Once the Cache input query loop is terminated, program will print Cache table, and reset eg 

At this point program resets back to "Number of cache blocks?: " . And it is just as if you were back at that step. You may enter a new number for cache blocks and select a different technique for caching this time around.

Test Case Predictions:

My Cache Program will be fed the same set of inputs for the four different caching methods (Direct-Mapped, Two-Way Associative, Four-Way Associative, and Fully Associative). I will perform the various caching methods by hand and place the results in the tables below. The tables produced by my program and produced manually should map up perfectly.

I have selected Inputs that represent the widest possible failure scenarios.

Inputs in order { 1 , 11 , 1 , 21 , 21 , 31 , 41 , 09 , 31, 19 , 29 , 39 , 49 , 0 , 2 , 4 , 6 , 7 , 8 , 33 , 77 }

These Inputs were selected because they are

1. Compact (too many input would require laborious calculations by hand)
2. Will represent LRU properly (4 way-associative many unique numbers contend for the same spot)
3. Null in the 6th location or Block Address [05]

Predicted Cache Results for each Caching method:

|  |  |
| --- | --- |
| Direct-Mapped table | |
| Block Address: [0] | Mem[0] |
| Block Address: [1] | Mem[31] |
| Block Address: [2] | Mem[2] |
| Block Address: [3] | Mem[33] |
| Block Address: [4] | Mem[4] |
| Block Address: [5] | null |
| Block Address: [6] | Mem[6] |
| Block Address: [7] | Mem[77] |
| Block Address: [8] | Mem[8] |
| Block Address: [9] | Mem[49] |

Miss ratio: 95.2% Hit ratio: 4.8%

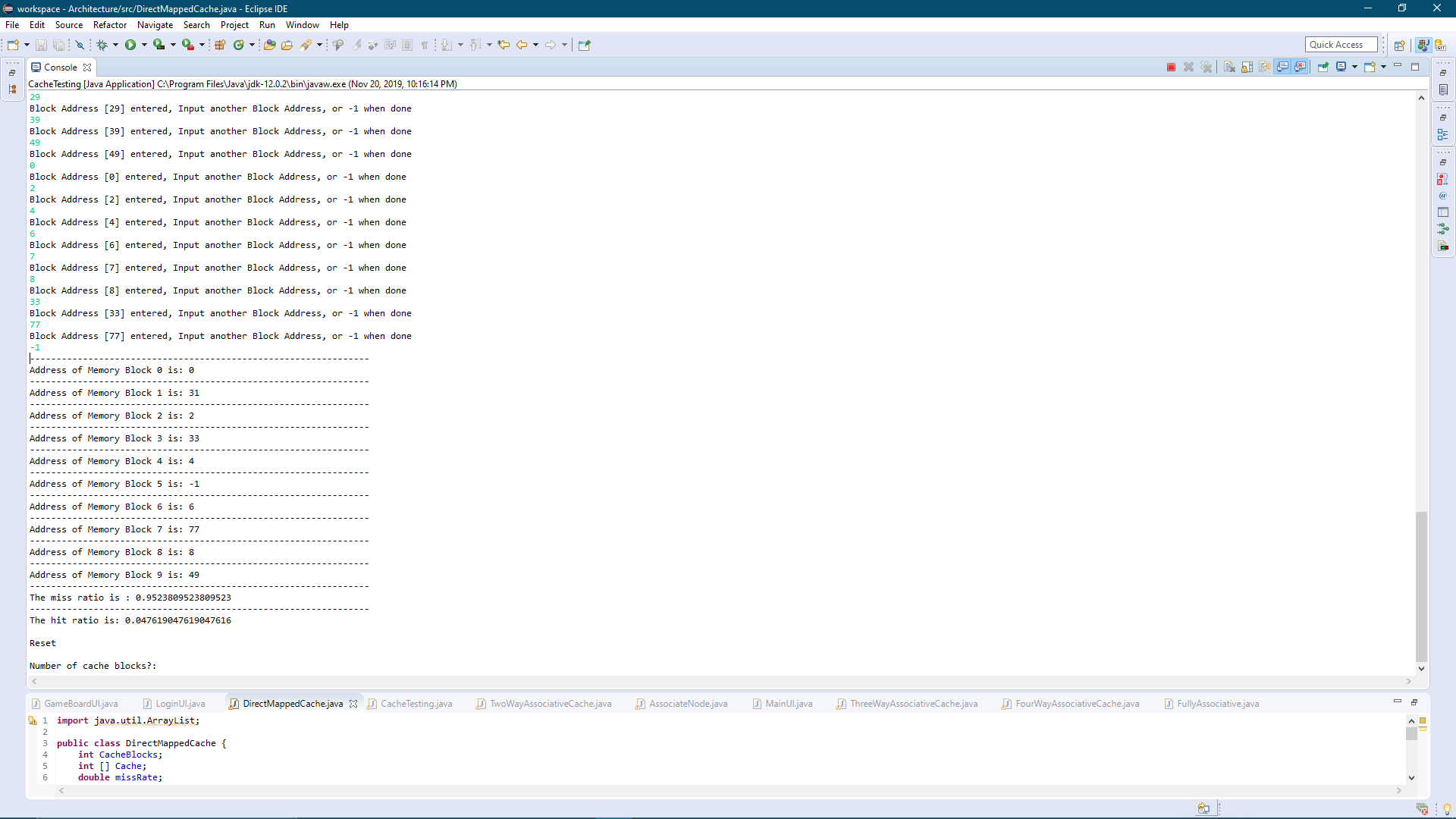
|  |  |  |
| --- | --- | --- |
| Two-Way Associative table | | |
| Block Address: [0] | Mem[0] | Null |
| Block Address: [1] | Mem[31] | Mem[41] |
| Block Address: [2] | Mem[2] | Null |
| Block Address: [3] | Mem[33] | Null |
| Block Address: [4] | Mem[4] | Null |
| Block Address: [5] | Null | Null |
| Block Address: [6] | Mem[6] | Null |
| Block Address: [7] | Mem[7] | Mem[77] |
| Block Address: [8] | Mem[8] | Null |
| Block Address: [9] | Mem[49] | Mem[39] |

Miss ratio: 85.7% Hit ratio: 14.28%

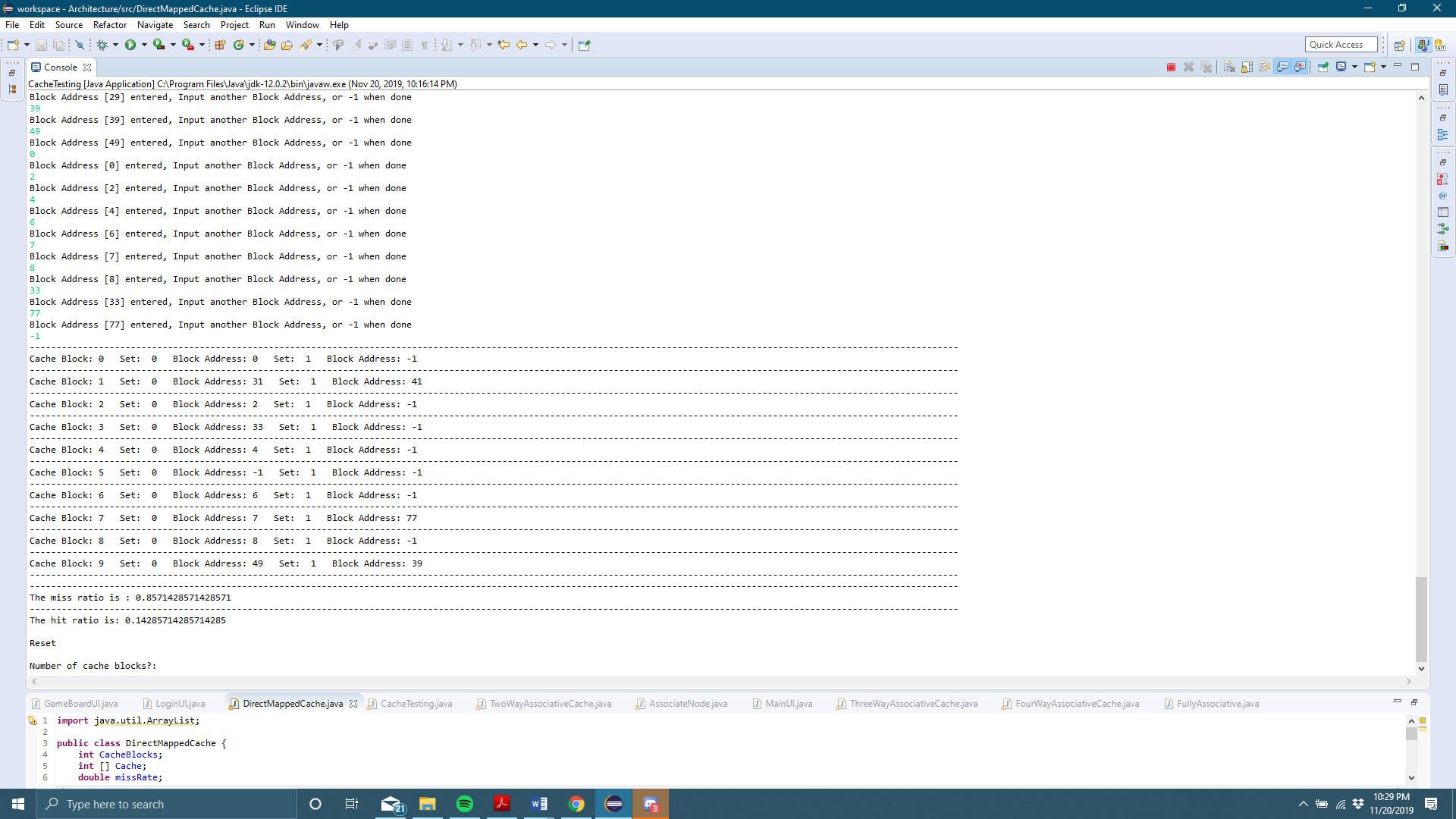
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Four-Way Associative table | | | | |
| Block Address: [0] | Mem[0] | Null | Null | Null |
| Block Address: [1] | Mem[1] | Mem[41] | Mem[21] | Mem[31] |
| Block Address: [2] | Mem[2] | Null | Null | Null |
| Block Address: [3] | Mem[33] | Null | Null | Null |
| Block Address: [4] | Mem[4] | Null | Null | Null |
| Block Address: [5] | Null | Null | Null | Null |
| Block Address: [6] | Mem[6] | Null | Null | Null |
| Block Address: [7] | Mem[7] | Mem[77] | Null | Null |
| Block Address: [8] | Mem[8] | Null | Null | Null |
| Block Address: [9] | Mem[49] | Mem[19] | Mem[29] | Mem[39] |

Miss ratio: 85.7% Hit ratio: 14.28

Results of Direct-Cache Method:



Results of Two-Way Association Method:



Results of Four-Way Association:

