COMP07027 Introduction to Programming

Practice Programming Project – 2014/15

Banner ID **B00265222** **Name Mark Williams**

Your program should address the following points:

Point	**Description (the facilities you need to implement)**	**Achieved**
1	The program follows the specification in providing a description of the problem, the initial state (the three disks are all initially on pole A) and in prompting the user to choose which disk to move and to which pole.	Yes. You allow the user to play again. However, you should have achieved this by using a while loop whose condition was reset, rather than by recursively calling main(). You should never use recursion (have a method call itself) if all you want is simple repetition. Also, a Java application should never call the main() method – only the Java Virtual Machine should do this.
2	The program defines a suitable set of variables to keep track of the location of the three disks.	Yes, the 3 static `int` variables were fine for this. Presumably you made them static so that they were visible to all four of the methods in the program.
3	The program gets the user input and, if it is valid, updates the program state so that the disk is moved to its new pole. This will need you to identify some more variables to keep track of inputs and conditions. If the input is not valid, the program displays an appropriate message and the program state is not updated.	This was all fine – except that when moving the small disk the user is asked which pole they want to move the medium disk to (just an error in the output – it is the small disk that gets moved).
4	After each move, the program displays where the three disks are now located.	Yes, but see comment on point 6.
5	After each move, the program checks if the puzzle has been solved	Yes, but see comment on point 6.
6	If the puzzle is not solved, the program repeats step 3.	Yes, but this is the really bad part – the code in main() that calls one(), two() or three() should be in a loop that terminates when all three disks are on

		pole C. You do not use a loop but have the three methods each call main() to repeat the steps. The main() method then calls them – so you have mutual recursion. This is extremely inefficient (it uses a lot of memory) and the only reason your program exits correctly is that you call System.exit() to close down the whole Java Virtual Machine. Again, as noted in 1 above, a Java application should never call the main() method of the application.
7	The program counts the number of moves and displays a message at the end of the program if it has taken more than the seven moves required for an optimal solution.	Yes, and you maintain a high score, which is a nice touch.
8	Program structure. The solution to last year's practice project had all the code in one subroutine. Consider using subroutines to do such things as validate the input, to check the program state, and to update the program state. Document your program with comments in the code. At the very least, there should be comments against each variable declaration to explain what the variable is for and, if you have subroutines, above each one explaining what the subroutine does. Include a comment at the beginning of the program explaining what the program does.	The three static methods, one for moving each disk, are a reasonable way to separate the logic, but are more complex (i.e. have more code in them) than they need and the mutual recursion between them and main() is not good (even if it were not main() but another static method that were involved). It would have been good to have a comment on each method explaining what it does, and in fact the program contains very few comments.