Solution

Problem 1: The Knight's tour problem (122090007_A3_p1.cpp)

The program use `vector< vector<int> >` to store the chessboard. In the `find_solution()` function we first declare two array to store the steps that the knight can move. In this recursive function, when the chessboard is full it returns. The function first tries the steps in the array, if available it continues the next step, if not it returns to the last step until there are no more alliable step it ends. It uses the back tracking method.

Problem 2: Program for merge sort (122090007_A3_p2.cpp)

The program has two functions, `merge()` and `merge_sort()`. The function `merge()` is used to merge two arrays into one array from smallest to largest. The function `merge_sort()` is a recursive function. It first check whether the previous array has at least two elements and then derive their middle index, divide then into two parts and then sort then again using itself, then use the `merge()` functions to merge then together.

Problem 3: Text editor (122090007_A3_p3.cpp)

The program defines a class TextEditor, it has two elements `current_text` and `cursor_position`, and four operate functions add (using insert function), delete (using erase function), cursor right and cursor left (changing the cursor position).

The program use `get_command()` to get all the operations that need to be done and use the class functions and print then out each time.

Problem 4: Priority Queue (122090007_A3_p4.cpp)

The program defines a class PriorityQueue and a structure Node. The Node contains the string data, the data's priority and the next Node. The class has a front and tail to store the first and last node in the queue. The function of enqueue first compare the node data's priority and then add to the queue to a right position. The dequeue is the normal operations of queue, using first in first out method.

The program use `get_data_command()` to get all the operations and data that need to be done and use the class functions and print then out finally.