1. **Title:** Dynamic Math Quiz

2. **Data Structured used:**
    - Linked List
    - Priority Queue

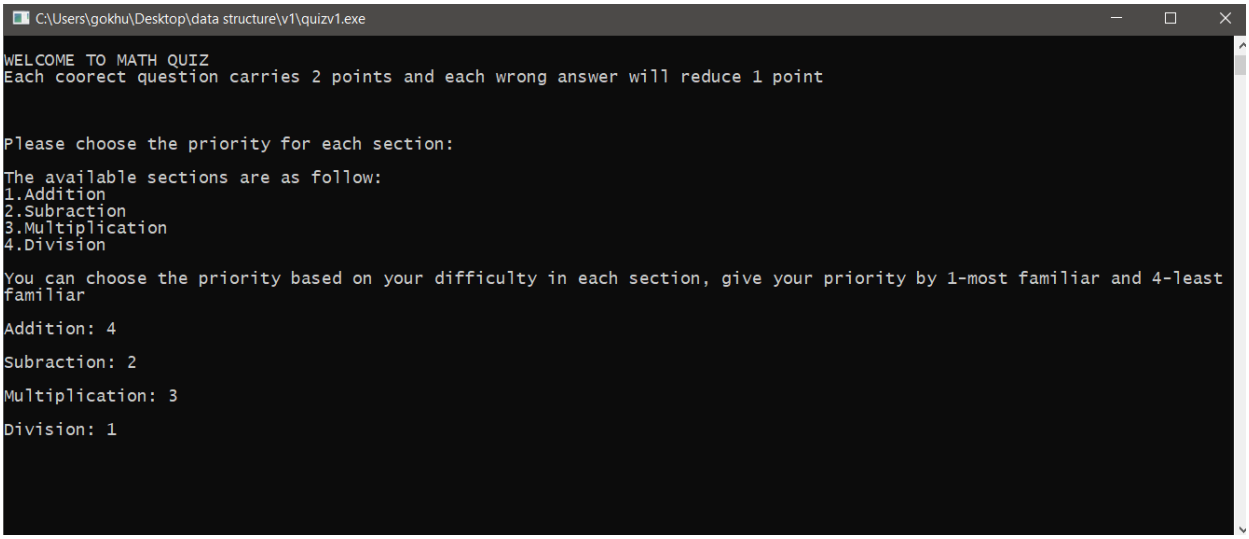
3. **Other Functionality used:**
    - Files
    - System Commands

4. **Description:**

As the name states this Math Quiz is a Dynamic one which prioritises the question based on the user's knowledge on the specific section of question. In brief the user is asked to enter the well-known section as 1 and the least known section as 4, so the other section must be marked in the given range. The program automatically selects a higher number of questions from the well-known section and the least from the least known section. Now the program starts the quiz in a new fresh screen clearing all the previous details. The rules are pretty simple, the user gets 2 points for correct answer and a negative point for wrong answer. Finally, the program displays the score of the particular user.

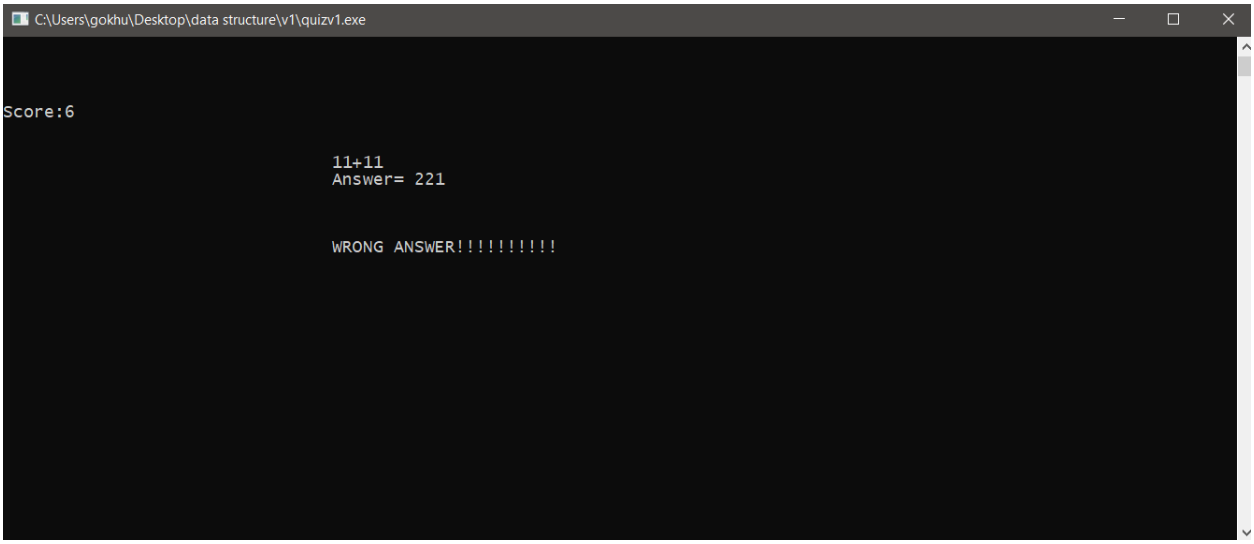
5. **Code Explanation:**

The initial screen,

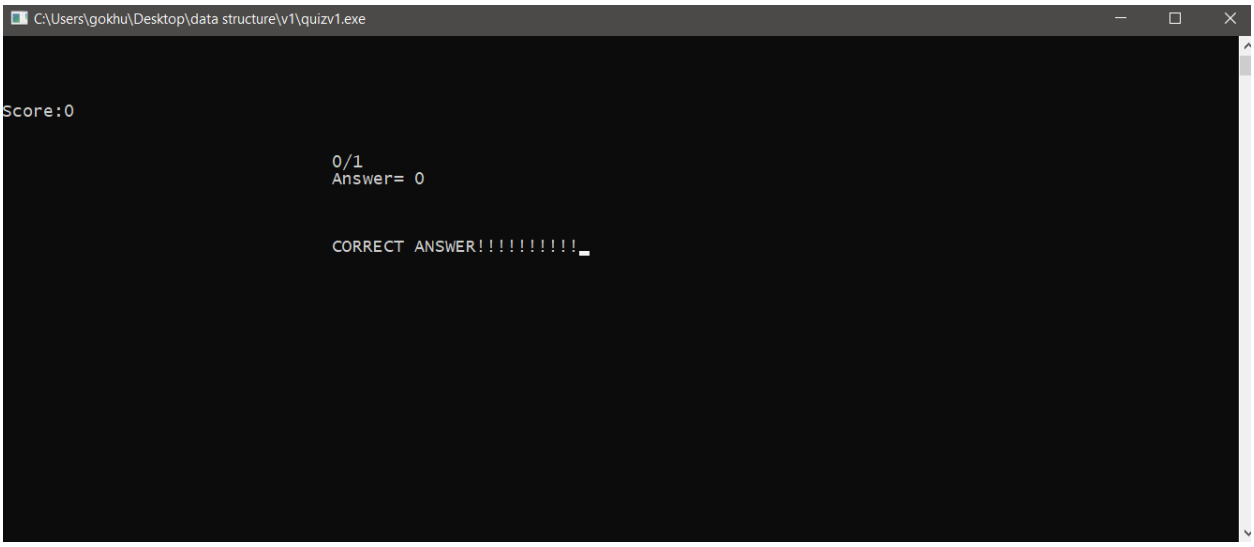



Here the rules and regulations are displayed. And it prompts to enter the difficulty level of each section. The concept of **Priority Queue** is used in here, as the user enters the priority the corresponding section and its priority is enqueued as data and key which then gets priorities. The top priority one gets 4 question added to the **Linked List** then the

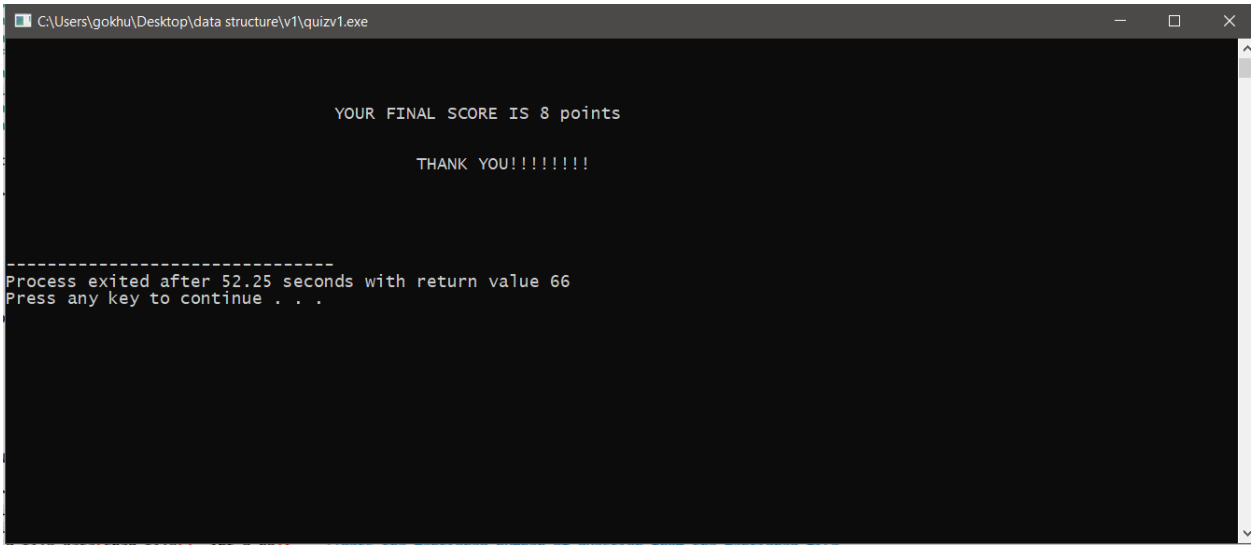
remaining as 3, 2, 1 question. So, this addition in to linked list happens immediately after the dequeue from the priority queue. Then the list is traversed and the questions are displayed.




The score is updated simultaneously as the questions are updated. If the user gives a wrong answer then the score will be deducted by 1 point.




If the user answer correctly 2 points will be added. So, after completion of 10 question the last page will display the final score.




6. **Function Prototype:**
    - struct node_LL *start = NULL; **//Start of the linked list node**
    - void insert(char q[],int a); **//inserts question and its answer to linked list**
    - void traverse_question(); **//displays all the question and answer**
    - void file_data(char file[], int q_no); **//gets the mentioned number of question from the mentioned file**
    - Node_PQ* newNode_PQ(int d, int p); **//start node of the priority queue**
    - int peek(Node_PQ** head); **//peek the top element**
    - void pop(Node_PQ** head); **//pops the top element**
    - void push(Node_PQ** head, int d, int p); **//push the element to the queue**
    - int isEmpty(Node_PQ** head); **//checks whether the queue is empty**
    - void checkAnswer(char q[],int a); **//checks the answer and updates the score;**

7. **Additional Features:**
    - Each section has a separate text document that stores the list of questions. So we can append as much questions we can.
    - We can change the total number of questions to display.
    - We can add as many sections we can.
    - Since the text file is read in a way that the question is considered as String and answer is considered as Integer which makes it easy to add word problem too.