Exercise 06 - Select Algorithm

The `Select()` algorithm finds the ith smallest element in a collection in linear time. It consists of five steps:

1. Divide the collection into consecutive groups of five (last group may contain fewer than 5 elements).
2. Find the median of each group (sorting may be required).
3. Recursively apply `Select()` to the list of medians to find the pivot, the median of medians.
4. Partition over the pivot and find k, the count of elements less than the pivot.
5. Compare the i (desired position) and k. If i = k, return the pivot; otherwise, if i ¡ k, perform `Select()` on the lesser partition; otherwise, perform `Select()` on the upper partition with i = i - k.

However, if the collection contains only a single element, the `Select()` algorithm simply returns that element.

Create a header file named '`exercises06.h`' that defines the `Select()` algorithm with the following parts and uses a vector as the array.

1. Define an integer function `Mid()` that accepts an array and two integer indices low and high (with low $\leq$ high) and returns the index of the median of the subarray A[low. . . high]. If the subarray length exceeds 5, it returns -1.

2. Define a vector function `Medians()` that takes an array parameter and returns an array of the medians of consecutive groups of five elements of the array.

3. Define the integer function `Partition()` that takes an array, two integer indices low and high (with low $\leq$ high), and a pivot value, and partitions the array around the pivot, returning the count of the lower partition.

4. Define the `Select()` function using the previous functions and its description.