9. Design and implement C/C++ Program to sort a given set of n integer elements using Selection Sort method and compute its time complexity. Run the program for varied values of n> 5000 and record the time taken to sort. Plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

// Function to perform Selection Sort
void selectionSort(int arr[], int n) {
    int i, j, min_idx;

    for (i = 0; i < n-1; i++) {
        min_idx = i;
        for (j = i+1; j < n; j++) {
            if (arr[j] < arr[min_idx]) {
                min_idx = j;
            }
        }
        // Swap the found minimum element with the first element
        int temp = arr[min_idx];
        arr[min_idx] = arr[i];
        arr[i] = temp;
    }
}

// Function to generate random integers and write to file
void generateAndWriteToFile(const char *filename, int n) {
    FILE *fp;
    int i;
    fp = fopen(filename, "w");
    if (fp == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    srand(time(NULL));
    for (i = 0; i < n; i++) {
        fprintf(fp, "%d\n", rand() % 10000); // Write random integers between 0 and 9999
    }

    fclose(fp);
}

// Function to read integers from file
void readFromFile(const char *filename, int arr[], int n) {
    FILE *fp;
    int i;
    fp = fopen(filename, "r");
    if (fp == NULL) {
        printf("Error opening file.\n");
        exit(1);
    }

    for (i = 0; i < n; i++) {
        if (fscanf(fp, "%d", &arr[i]) != 1) {
            printf("Error reading from file.\n");
            fclose(fp);
            exit(1);
        }
    }

    fclose(fp);
}

int main() {
    const char *filename = "input.txt";
    const char *outputFilename = "sorting_time.csv";
    int n,i,arr[10000];
    // Measure the time taken for sorting
    clock_t start, end;
    double cpu_time_used;

    FILE *output_fp;
    output_fp = fopen(outputFilename, "w");
    if (output_fp == NULL) {
        printf("Error opening file.\n");
        return 1;
    }

    fprintf(output_fp, "n,Time taken (seconds)\n");

    for (n = 5000; n <= 10000; n += 1000) {
        arr[n]; // Array to store the elements

        // Generate random integers and write to file
        generateAndWriteToFile(filename, n);

        // Read integers from file
        readFromFile(filename, arr, n);

        start = clock(); // Start time

        // Perform Selection Sort
        selectionSort(arr, n);

        end = clock(); // End time
        cpu_time_used = ((double) (end - start)) / CLOCKS_PER_SEC; // Calculate time taken
        printf("Time taken to sort %d elements: %f seconds\n", n, cpu_time_used);
        fprintf(output_fp, "%d,%f\n", n, cpu_time_used);
    }
    for(i=0;i<5000;i++)
    printf("%d\t",arr[i]);

    fclose(output_fp);
    printf("Data written to %s\n", outputFilename);

    return 0;
}
```