**Problem.** Write a program that uses structures and pointers. You will have to write two functions: **get_stats**, and **get_median**.

**(1)** You first need to declare a structure type **driver_t**.
I named my structure **driver_t** and its 4 parts are:
a character array **name** that is 21 in length,
a double array of **tries** that has a length of TRIES,
a double named **best_time**, and
a double named **deviation**.

| Name | **Try 0** | **Try 1** | **Try 2** | Best Time | Deviation |
|---|---|---|---|---|---|

**(2)** You next need to declare a structure type **stats_t**.
I named my structure **stats_t** and its 4 parts are:
four variables, all type double, named **best_average**, **fast_time**, **slow_time**, and **median**.

**(3)** Write the function **get_stats**. The prototype is:

```
void get_stats(driver_t driver_list[NRACERS],      /* in & out */
               stats_t *race_stats );              /* in & out */
```

This function will figure the driver's best time, the track slow time and fast time, the average of the driver's best times, and the driver's deviation from the fast time.

**(4)** Write the function **get_median**. The prototype for **get_median** is:

```
void get_median(driver_t driver_list[NRACERS],
                stats_t *race_stats );
```

It will find the mid best time from the sorted list of best times.

**(5)** You will be provided a test driver program that needs NO changing, only ADDing.
You will only need to add the two structures and the two functions as above.

**Input/Output Description**:
The program input is a set of driver's names and their three tries on the race track in one file. The race times are type double. Each record/line of the file has a student name and three times.

The first line from the sample data file is:

```
Jay Johnson      4.0  5.0  6.0
```

The output is printed to **lab5.txt** as shown in the sample output.

**<u>Algorithm Development - Pseudo code</u>**:

```
/*----------------------------------------------------------*/
main
  /* This function already exists. */
  out_file = open_out_file ();
  get_data(IN_FILENAME, driver_list);
  get_stats(driver_list, &race_stats);
  do_sort(driver_list);
  get_median(driver_list, &race_stats);
  print_all(out_file, driver_list, &race_stats);


/*----------------------------------------------------------*/
FILE * open_out_file(void)
    /* This function already exists. */
    /* Opens the output file */


/*----------------------------------------------------------*/
void get_data (char *filename,                 /* input  */
                   driver_t driver_list[NRACERS] );  /* output */

    /* This function already exists. */
    /*It opens the data file and reads it into the appropriate places. */


/*----------------------------------------------------------*/
void print_all(FILE * out_file,
                driver_t driver_list[NRACERS] ,
                stats_t *race_stats )

    /* This function already exists. */


/*----------------------------------------------------------*/
void do_sort(student_t student_list[NSTUDENTS])

    /* This function already exists. */


/*----------------------------------------------------------*/
```

➔ more on next page

```
/*----------------------------------------------------------*/
/*    THIS IS A SUB-FUNCTION THAT YOU HAVE TO WRITE        */
void get_stats( driver_t driver_list[NRACERS],    /* in & out */
                stats_t *race_stats )             /* in & out */

  Zero out the best_average  (HINT: use the -> notation)
  Set the slow_time to the first driver's first try.
  Set the fast_time to the first driver's first try.

  loop from d=zero to < NRACERS increment by one
 {
        zero out the driver_list[d].deviation
        set the driver's best time to the driver's first time
        loop from t=zero to t< TRIES increment by one
        {
           figure the driver's best time
            find the fastest and slowest track time
        }
        add the driver's best time into the running total of best times
 }
  compute the average of the best times

  loop from d=zero to < NRACERS increment by one
  {
        figure the driver's deviation from the class average
        (deviation is fast time minus driver's best time)
  }
  return

/*----------------------------------------------------------*/
/*    THIS IS A SUB-FUNCTION THAT YOU HAVE TO WRITE        */
void get_median(driver_t driver_list[NRACERS],
                  stats_t *race_stats )

   zero out the median.
   calculate the mid point (divide NRACERS by two)
   if  the number of racers is odd then
      set the median to the mid average
   else
      set the median to the average of the two numbers(averages) on
             each side of the median. [mid] & [mid-1]. NO integer division.
/*----------------------------------------------------------*/
```

➔ **Examples of Median on next page.**

**NOTES on the median**:
The median is the value in the middle of a group of values, assuming that the values are sorted. If there is an odd number of values, the median is the value in the middle. If there is an even number of values, the median is the average of the values in the two middle positions.
For example, the median of values {1, 6, 18, 39, 86} is the middle value, or 18.
The median of values {1, 6, 18, 39, 86, 91} is the average of the two middle values, or (18 + 39)/2 or 28.5.

**Hand Example**:
This is a sample data example. It does not match the lab5.dat file in length **or** in value!

```
      SAMPLE DATA:
      Jay Johnson   4.0  5.0  6.0
      Lenny Loop    2.0  3.0  4.0
      Missy Monroe  1.0  2.0  3.0
      Ned Niner     3.0  7.0  5.0
```

**Sample Output :**

```
Your Name.  Program 5 output.

Track Results

Driver          Try 1   Try 2   Try 3   Best Time    Deviation
-------------   -----   -----   -----   ----------   ---------
Missy Monroe     1.0     2.0     3.0       1.0           0.0
Lenny Loop       2.0     3.0     4.0       2.0          -1.0
Ned Niner        3.0     7.0     5.0       3.0          -2.0
Jay Johnson      4.0     5.0     6.0       4.0          -3.0

 The average of best times =  2.500

 The track fast time       =  1.000

 The track slow time       =  7.000

 The median of best times  =  2.500
```

**Using the Sample Data:**
To use the sample date, you need to make changes:

#define IN_FILENAME "lab5.dat" needs to change the file name to lab5sample.dat.
#define NRACERS 10 needs to change the 10 to 4 drivers.
You might also want to change the output file name.

Do remember to undo these changes as you prepare to turn your work in.

➔ **more on next page**

**Files To Copy**:
Copy the need files (lab5.c, lab5.dat, lab5sample.dat).
Move to your **csc60** directory.
Type: **`cp  /gaia/home/faculty/bielr/classfiles_csc60/lab5* .`**
^space space ^ & then **dot**

After the files are in your account, you need to type: **chmod 644 lab5***

**Prepare Your File For Grading:**
Make sure your program has been corrected to use **lab5.dat** and has been re-complied.

When all is well and correct, type: **script StudentName_lab5.txt**

| | | |
|---|---|---|
| At the prompt, type: | **cat lab5.c** | to display the code in your session. |
| At the prompt, type: | **a.out** | to run the program |
| Type: | **cat lab5.txt** | to show contents of the output file |
| After the program run is complete, | | |
| | type: **exit** | to leave the script session |

**Turn In Completed Session:** Go to SacCT and turn in your session (StudentName_lab5.txt).