**Problem.** Write a program that uses structures and pointers.

You will have to write two functions: **get\_stats**, and **get\_median**.You will have to write a **makefile**.

**(1)** First in **lab7.h,** you need to declare a structure type **driver\_t**.

I named my structure **driver\_t** and its 4 parts are:

a character array **d\_name** that is 21 in length, (comes from the data file)

a double array of **d\_tries** that has a length of TRIES, (comes from the data file)

a double named **d\_best\_time**, (value computed by program)

a double named **deviation**, (value computed by program).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| d\_name | **d\_tries 0** | **d\_ tries 1** | **d\_ tries 2** | d\_Best Time | Deviation |

Array

**(2)** Next in **lab7.h**, you 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:

**average\_of\_best**, **winning\_time**, **slowest\_time**, and **median**.

**(3)** Add your name to the comment block in **lab7.h**. Eventually you will need to shift the comment marks ( // ) on the four #define statements: two for the data file, two for the NRACERS

**(4)** Write the function **get\_stats**. 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. The prototype is:

void get\_stats(driver\_t driver\_list[NRACERS], /\* in & out \*/

stats\_t \*race\_stats ); /\* in & out \*/

**(5)** Write the function **get\_median**. It will find the mid best time from the sorted list of driver best times. Examples of computing median are on the top of page 4. The prototype for **get\_median** is:

void get\_median(driver\_t driver\_list[NRACERS], stats\_t \*race\_stats );

**(6)** Write the makefile with a comment block at the top that includes your name.

**Input/Output Description**:

The program input is a set of driver's names and their three tries on the racetrack 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 4.100 5.300 6.700

The output is printed to **lab7.txt** as shown in the sample output.

**Algorithm Development - Pseudo code**:

/\*-------------------------------------------------------------\*/

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** \*/

// *Remember to include lab7.h and put this code in its own file*

void get\_stats( driver\_t driver\_list[NRACERS], /\* in & out \*/

stats\_t \*race\_stats ) /\* in & out \*/

Zero out the average\_of\_best (HINT: use the -> notation)

Set the slowest\_time to the first driver’s first try.

Set the winning\_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 , **d\_best\_time**.

find the winning and slowest track times. **winning\_time, slowest\_time**.

}

add the driver's best time (**d\_best\_time**) into the running total of driver’s best times

}

compute the average of the best times. **average\_of\_best**.

loop from d=zero to < NRACERS increment by one

{

figure the driver's deviation from the **winning\_time**

(deviation is fastest time minus driver's best time)

}

return

/\*-------------------------------------------------------------\*/

/\* **THIS IS A SUB-FUNCTION THAT YOU HAVE TO WRITE** \*/

// *Remember to include lab7.h and put this code in its own file*

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.

EXAMPLES:

1. The median of values {1, 6, **18**, 39, 86} is the middle value, or 18.
2. 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.

**Sample Data**:

This is the sample data example. It does not match the lab7.dat file in length **or** in value!

SAMPLE DATA:

Jay Johnson 4.100 5.300 6.700

Missy Monroe 1.000 2.000 3.500

Ned Niner 3.800 7.000 5.500

Lenny Loop 2.200 3.400 4.600

**Sample Output:**

Your Name. Lab 7 output.

Track Results

Drivers Try 1 Try 2 Try 3 Best Time Deviation

-------------------- --------- --------- --------- ---------- ---------

Missy Monroe 1.000 2.000 3.500 1.000 0.000

Lenny Loop 2.200 3.400 4.600 2.200 -1.200

Ned Niner 3.800 7.000 5.500 3.800 -2.800

Jay Johnson 4.100 5.300 6.700 4.100 -3.100

The average of best times = 2.775

The track fast time = 1.000

The track slow time = 7.000

The median of best times = 3.000

**Using the Sample Data:**

To use the sample data, the lines in lab7.h should be:

**//#define IN\_FILENAME "lab7.dat"**

**#define IN\_FILENAME "lab7sample.dat"**

**//#define NRACERS 10**

**#define NRACERS 4**

**Using the Final Data:**

To use the final data, the lines in lab7.h should be:

**#define IN\_FILENAME "lab7.dat"**

**//#define IN\_FILENAME "lab7sample.dat"**

**#define NRACERS 10**

**//#define NRACERS 4**

**Files To Copy for *aethna***:

Type: **cp -R /gaia/home/faculty/bielr/files\_csc60/lab7 .**

Spaces needed: (1) After the **cp *↑*** *Don’t miss the space & dot.*

(2) After the **-R**

(3) After the directory name at the end & before the dot.

After the files are in your account and you are still in **csc60**,

you need to type: **chmod 755 lab7**

This will give permissions to the directory.

Next move into lab7 directory by typing: **cd lab7**

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

This will give permissions to the files.

Your new lab7 directory should now contain:

*lab7.c, lab7.h, lab7.dat, lab7sample.dat,*

*get\_data.c, open\_out\_file.c, do\_sort.c, print\_all.c*

**Hint about for loops on our compiler**:

*Our compiler does not like:* **for (int i = 0; …..)**

You will receive the following errors:

test\_loopcounter.c:6: error: ‘for’ loop initial declarations are only allowed in C99 mode

test\_loopcounter.c:6: note: use option -std=c99 or -std=gnu99 to compile your code

These errors imply that on every “gcc” line, you must add: **-std=c99** OR -std=gnu99.

*It does like it on two lines:*

**int i;**

**for (i = 0; ……)**

**PREPARE YOUR FILE FOR GRADING:**

Make sure your program has been:

* corrected to use **lab7.dat**
* corrected to use the proper valuefor **NRACERS**
* has been re-complied by doing a **make**

When all is well and correct,

Type: **script StudentName\_lab7.txt** [Script will keep a log of your session.]

Type: **touch lab7.h [**This is a late add, no points off if you don’t see it.**]**

Type: **make** which will run your makefile

Type: **lab7** to run the program to show the output of the program

(or whatever name you used for the executable)

Type: **cat lab7.txt** to see the output of your program

Type: **exit** to leave and save the script session

**Turn in your completed session:**

Go to Canvas and turn in (no zip files):

1. lab7.h
2. get\_stats.c
3. get\_median.c
4. print\_all.c
5. makefile
6. your script session (StudentName\_lab7.txt)