## Question 4: Functions and Pointers

Complete the code in function get\_sum to calculate the sum of all elements of an array. Complete the line of code to call the function in main(). Your solution is restricted to appropriately coding the parameters.

#include “stdio.h”

// sum the elements of an array

void get\_sum (int \*toSum, arraySum[], int numberAmount)

{

int i = 0;

\*toSum = 0;

for(i = 0; i < numberAmount; i++){

\*toSum += arraySum[i];

}

}

// example of calling get\_sum()

main()

{

int array\_sum;

int my\_array[] = {25, 18, 6, 47, 2, 73, 100};

get\_sum (&array\_sum, my\_array, 7);

printf(“sum = %d, array\_sum);

// Desired output of printf: sum = 271

}

## MyQuestion : Functions and Pointers

void get\_Median(int \*middleNum, arraySum[], int size)

{

//uses a selection sort to make finding the median easier

int i = 0, j = 0, temp = 0,smallest = 0;

\*middleNum = 0;

for(i = 0; i < size; i++) {

for(j = 1; j < size; j++) {

if(arraySum[i] > arraySum[j]) {

smallest = arraySum[j];

}

}

temp = arraySum[i];

arraySum[i] = smallest;

arraySum[j] = temp;

}

if(size % 2 == 0) {

\*middleNum = arraySum[size/2]+arraySum[(size/2)-1];

\*middleNum /= 2;

}

else {

\*middleNum = arraySum[size/2];

}

}

main()

{

int median;

int my\_array[] = {99, 10, 45, 23, 105, 15, 1};

get\_sum (&median, my\_array, 7);

printf(“sum = %d, median);

// Desired output of printf: median = 23

}

Part 2)

* Some key concepts that can be seen inside this question is memory usage, how functions work in C, and how pointers work as well too. Memory usage comes up because if we were to scan everything in when using the iRobot using a pointer would save more memory then assigning each pin to a primitive type. As you can see the entire function is written above the main method which is why it can be called inside the main method. Another way to write it would be to give the definition above the main method and then write the function below the main method. The function is also void which means it doesn’t return anything which gives insight on what parameters are needed for the function and how it will work. Pointers are used in this function because the address of the variable is equal to the pointer in the function line it can be seen inside both functions that \* is used to calculate the value of the sum and median value. By passing the address of the variable and using the pointer it sets the value of that variable to the sum and or median value. (pointers point to the address; from the address you can get the value)

Part 3)

* I used the “C book” and then went into the pointers chapter to help me understand the tools I needed to write these functions while still using pointers. (I could have just passed the value I wanted to be summed or get the median value from without using pointers, but it needed to be for this question) it gives a good example on how pointers work. It shows int x, \*p; then goes on to set p = &x; \*p = 0; and then in comments shows that this now sets x equal to zero. From this I then understood that I could pass the variable into the function with an & and then function parameters have it be a \*pointer and then inside the function just use \*pointer = whatever I needed to complete the function. This then would update variable through the function to the requirements allowing me to then print showing I have the right values even with a void function.