COMP 2511

Lesson 6 Lab

Task 1:
To define a function which takes in an int array and the length of it, finds and returns the mean of the array
Prototype: **double mean (const int data[], size_t length);**

```
double mean (const int data[], size_t length)
{
    int sum = 0;
    for (int x = 0; x < length; x++)
    {
        sum += data[x];
    }
    return (float)sum / length;
}
```

Task 2:
To define a function which takes in an int array and the length of it, finds and returns one number from the array which has the biggest absolute value
Prototype: **int max_abs (const int data[], size_t length);**

```
#include <math.h>

int max_abs(const int data[], size_t length)
{
    int max = data[0];
    for (int x = 1; x < length; x++)
    {
        if (abs(data[x]) > abs(max)) max = data[x];
    }
    return max;
}
```

Task 3:
To define a function which takes in an int array and the length of it, along with two other integers val_1 and val_2, finds and returns how many numbers in the array is between val_1 and val_2, inclusively.
[Hint: to determine how many numbers are val_1 ≤ number ≤ val_2]
Prototype:
size_t in_between (const int data[], size_t length, int val_1, int val_2);

```
size_t in_between(const int data[], size_t length, int val_1, int val_2)
{
   int count = 0;
   for (int x = 0; x < length; x++)
   {
      if (data[x] >= val_1 && data[x] <= val_2) count++;
   }
   return count;
}
```

Task 4:
To define a function which takes in a two-dimensional int array (with size of n-by-10) and an integer n for the row number, to find and return the minimum value in the array [Hint: column number is 10]
Prototype:
int minimum (const int data [][10], const size_t n);

```
int minimum(const int data[][10], const size_t n)
{
    int min = data[0][0];
    for (int x = 0; x < n; x++)
    {
        for (int y = 0; y < 10; y++)
        {
            if (data[x][y] < min) min = data[x][y];
        }
    }
    return min;
}
```

Task 5:
To define a function which takes in a two-dimensional int array (with size of n-by-7) and integer n for the row number, to find and return how many even numbers are in the array [Hint: column number is 7]
Prototype:
size_t count_even (const int data [][7], const size_t n);

```
size_t count_even(const int data [][7], const size_t n)
{
    int count = 0;
    for (int x = 0; x < n; x++)
    {
        for (int y = 0; y < 7; y++)
        {
            if (data[x][y] % 2 == 0) count++;
        }
    }
    return count;
}
```