**Every Boilermaker Engineering Code – Entry Level Programming**

**Week 6 – Programming Exercises**

1. **(15 points, Rainfall Statistics)** Design a program that lets the user enter the total rainfall for each 12 months into a **list**. The program should calculate the display the total rainfall for the year, the average monthly rainfall, the months with the highest and lowest amounts. (**format the decimal precision of results as 2.**)

**Use the following numbers to test (Only type in the red numbers in interactive mode):**

|  |  |
| --- | --- |
| **Sample Input** | **Expected Output** |
| **Enter the rainfall for January: 1.925**  **Enter the rainfall for February: 1.856**  **Enter the rainfall for March: 2.6**  **Enter the rainfall for April: 3.584**  **Enter the rainfall for May: 4.761**  **Enter the rainfall for June: 4.092**  **Enter the rainfall for July: 4.215**  **Enter the rainfall for August: 3.628**  **Enter the rainfall for September: 2.863**  **Enter the rainfall for October: 3.031**  **Enter the rainfall for November: 3.237**  **Enter the rainfall for December: 2.449** | **Total rainfall: 38.24**  **Average rainfall: 3.19**  **Highest rainfall: May**  **Lowest rainfall: February** |

1. **(15 points, Larger than n)** In a program, write a function that accepts **two** arguments: a **list, and a number n**. Assume that the list contains numbers. The function should display all of the numbers in the list that are greater than the number n. Then call the function in the program.

**You can create the list and the number n in the main function of your code and run your code without inputs**.

**Use the following numbers to test:**

**Define the list as [19,2940,10,24,29,1,85,201,-15,-122,799] and the number n as 13 in the main function in your code.**

|  |  |
| --- | --- |
| **Sample Input** | **Expected Output** |
|  | **Number: 13**  **List of numbers:**  **[19, 2940, 10, 24, 29, 1, 85, 201, -15, -122, 799]**  **List of numbers that are larger than 13:**  **[19, 2940, 24, 29, 85, 201, 799]** |

1. **(20 points, Prime Number Generation)** A positive integer greater than 1 is said to be ***prime*** if it has no divisors other than 1 and itself. A positive integer greater than 1 is ***composite*** if it is not prime. Write a program that asks the user to enter an integer greater than 1, then displays all of the numbers that are larger than 1 and less than or equal to the number entered and determine whether they are prime or composite. The program should work as follows:
   * Once the user has entered a number, the program should populate a list with all of integer from 2 up through the value entered.
   * The program should then use a loop to step through the list. The loop should pass each element to a function that displays the element whether it is a prime number.

**Use the following numbers to test (Only type in the red numbers in interactive mode):**

|  |  |
| --- | --- |
| **Sample Input** | **Expected Output** |
| **Enter an integer greater than 1: 23** | **2 is prime.**  **3 is prime.**  **4 is composite.**  **5 is prime.**  **6 is composite.**  **7 is prime.**  **8 is composite.**  **9 is composite.**  **10 is composite.**  **11 is prime.**  **12 is composite.**  **13 is prime.**  **14 is composite.**  **15 is composite.**  **16 is composite.**  **17 is prime.**  **18 is composite.**  **19 is prime.**  **20 is composite.**  **21 is composite.**  **22 is composite.**  **23 is prime.** |
| **Enter an integer greater than 1: 0** | **Invalid input.** |

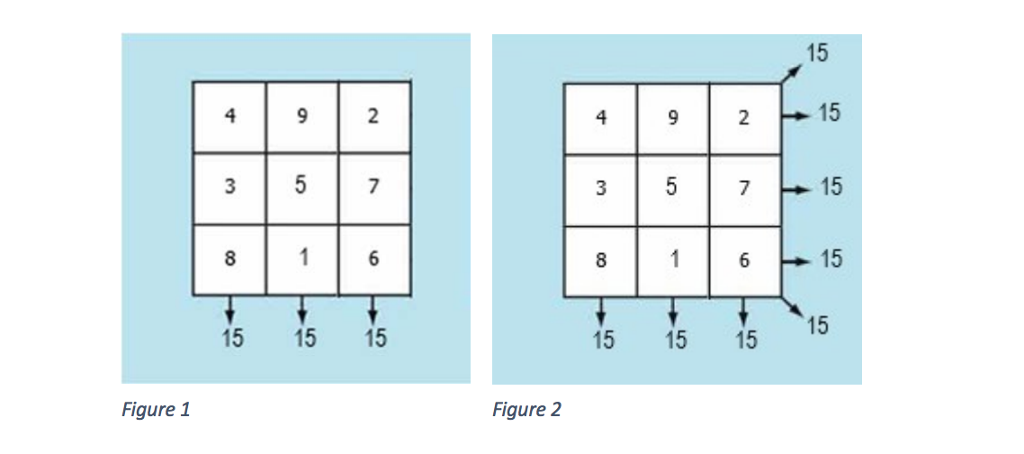
1. **(20 points, Number Analysis Program)** Design a program that asks the users to enter a series of 10 numbers. The program should store the numbers in a **list** then display the following data: (**format the decimal precision of total and average as 2.**)
   * The lowest number in the list
   * The highest number in the list
   * The total of the number in the list (by applying for loop)
   * The average of the numbers in the list

**Use the following numbers to test (Only type in the red numbers in interactive mode):**

|  |  |
| --- | --- |
| **Sample Input** | **Expected Output** |
| **Enter number 1 of 10: 1.5**  **Enter number 2 of 10: 6**  **Enter number 3 of 10: 9.2**  **Enter number 4 of 10: 1.35**  **Enter number 5 of 10: 3.14**  **Enter number 6 of 10: 3.1415**  **Enter number 7 of 10: 2.3**  **Enter number 8 of 10: 18**  **Enter number 9 of 10: 2.33**  **Enter number 10 of 10: 2.718** | **Lowest number: 1.35**  **Highest number: 18.0**  **Total: 49.68**  **Average: 4.97** |

1. **(20 points, Lo Shu Magic Square)** The Lo Shu Magic Square is a grid with 3 rows and 3 columns, shown in the down below figure. The Lo Shu Magic has the following properties:
   * The grid contains the numbers 1 through 9 exactly.
   * The sum of each row, each column, and each diagonal all add up to 15.

In a program you can simulate a magic square using a two-dimensional list. Write a function that accept a two-dimensional list as an argument and determine whether the list is a Lo Shu Magic Square.



**You can create the nested list of the square in the main function of your code and run your code without inputs**. **Test your function with 3 squares: [[1,2,3],[4,5,6],[7,8,9]], [[5,5,5],[5,5,5],[5,5,5]] and [[4,9,2],[3,5,7],[8,1,6]]**

|  |  |
| --- | --- |
| **Sample Input** | **Expected Output** |
|  | **Your matrix is:**  **1 2 3**  **4 5 6**  **7 8 9**  **It is not a Lo Shu Magic Square.** |
|  | **Your matrix is:**  **5 5 5**  **5 5 5**  **5 5 5**  **It is not a Lo Shu Magic Square.** |
|  | **Your matrix is:**  **4 9 2**  **3 5 7**  **8 1 6**  **It is a Lo Shu Magic Square!** |