Rationale for Data structures:

1) **Templated Binary Search Tree:**

In a Binary search tree, every node has at most two children and each node stores some information. In this assignment, the BST stores weather related information that is encapsulated in a class called **WindLogType**.

A templated Binary search tree has been implemented in order to store WindLogType data in it, and perform traversals. The traversals direct the relevant data to a callback function, where it is stored in a static vector called **windLogVec**.

Pros:

- BST reflects structural relationship in the data and represent hierarchies
- BST provides an efficient insertion and searching of data
- In this assignment, if any weather data readings have the same time date and time, then they are considered redundant and gets dropped by the tree.

Cons:

- There are even faster search methods that are available, example hash lookups. But hash lookups involve a much more complicated data structure.

2) **STL Map:**

A map is a sequence container that is used to store a collection of ordered entries that consists of key-value pairs. The keys must be unique, but the values need not be unique. Map supports operations like insert, find, erase, etc.

In this assignment, the STL Map has been used in following ways:

- Storing the WindLogType vector inside a nested Map, where year and months are keys for the two maps.
- Checking whether data for particular year and months exist before displaying the menu outputs
- Looping through the data to get the size of the data for a particular month or a day before inserting that data into the Binary search tree for performing traversals.

Pros:

- It is easier to lookup large data using Maps, for example, in this assignment he Map is being used to lookup for the weather data of a particular year and month directly without looping through the whole data.
- Since, a Map stores data in key order, it can be used to iterate over all the items in a map from beginning to end in a sorted key order.

3) **STL Vector:**

Vectors are dynamic arrays with the ability to resize itself when an element is inserted or removed. In this assignment, a static vector called windLogVec has been used in a call back function whenever a traversal is performed on the data stored in the BST.

For menu options 1 to 4, the required months' worth of data is pushed into the vector and operations are performed on this data. Whereas, for option 5 only a particular day's worth of data of the given month is pushed to the vector. The static vector is cleared each time after performing the operations.

Pros:

- Resizes itself automatically as per the size of the data being loaded
- Vector performs a contiguous storage of elements

Cons:

- Insertion at the end of the vector is fast, but insertion anywhere but the end can be slow.