Data Structures in C++

Array

An array is one of basic data structure which allows us to store specific number of elements which can be changed however, if it is a requirement then a different data structure should be chosen.

Arrays are quick and easy to use which offers fast element access and are easy to iterate over using a simple for loop. On the otherhand it does not have a lot of built in functionality.

Vector

Vectors are very similar to arrays which can also be said as dynamic arrays. Vectors should be used when an array is ideal but with frequent size manipulation required due to this, vectors offer a great deal more functionality than arrays. When it comes to vectors have both insert and remove. This will dynamically adjust the size of the vector as more or less space is required.

List

List are a data structure in which all elements in the list are connected to the previous and next element only. This makes the addition and deletion of objects very fast with the use of pointers. To add an object in a list, you can squeeze it in. Meanwhile with an array, all other values need to be moved. Though, this would make it really slow to retrieve elements as you need to iterate through the list until you reach the element you want. This is a lot slower than using the element access in a vector or an array. Lists offer even more functionality than arrays with the addition of reverse and sort built in.

MultiMap

Maps are a sorted data structure which seems similar to array but here data is stored as a key-value pair. The value is then accessed through use of the key we assign it. One advantage of using a Multimap over a regular map is that the same key can be used more than once while Maps do offer very fast retrieval of values through clever sorting techniques.

Conclusion

Based on this short analysis, I have chosen to implement a vector for my inventory. This is because it is easy to use and offers the amount of functionality I believe is required.