**Data Structures**

The Data Structures we want to use to hold the functional and non-functional requirements need to first to be able to hold text but we also need to hold a secondary information on if the stored data is a functional requirement.

**List**

The List while allowing us to store requirement does not allow for the separation of functional and non-functional requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| requirements | requirements | requirements | requirements | requirements |

We can however expand the use of lists by combining with another data structure such as a tuple to hold the type along with the requirement in a list

**List of Tuples**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| (Type, requirements) | (Type, requirements) | (Type, requirements) | (Type, requirements) | (Type, requirements) |

This allows the type and the requirement to be stored along with the actual requirement.

This would work but would require us to process the entries to identify which are functional or non-functional devices a better solution is to use a dictionary

**Dictionary of Lists**

|  |  |
| --- | --- |
| Functional | [LIST] |
| Non Functional | [LIST] |

**Pure Dictionary**

|  |  |
| --- | --- |
| F1 | ITEM |
| F2 | ITEM |
| NF1 | ITEM |
| NF2 | ITEM |

A list can store a series of items in a certain order meaning you can index into the list, or iterate over the list. A dictionary on the other hand is an example of a hash table structure and is a key-value type of structure. it requires that the dictionary keys are hash table. The advantage of this structure is speed of lookups

Paper

G. van Rossum and F.L. Drake, eds., Python Reference Manual,

Python Software Foundation, 2006; http://docs.python.org/ref/

ref.html.

G. van Rossum and F.L. Drake, eds., Python Reference Manual,

Python Software Foundation, 2006; http://docs.python.org/ref/

ref.html.

<https://docs.python.org/3/reference/>