Jewelia England

| Vector | Line cost | Number executed | Total cost |
| --- | --- | --- | --- |
| Open file | 1 | 1 | 1 |
| Read data one line at a time | 1 | n | n |
| Validate course | 1 | n | n |
| Check prerequisites | 1 | n | n |
| Create object and insert | 1 | n | n |
| Close file | 1 | 1 | 1 |

| Hash | Line cost | Number executed | Total cost |
| --- | --- | --- | --- |
| Initialize | 1 | 1 | 1 |
| calculate | 1 | n | n |
| insert | 1 | n | n |
| print | 1 | n | n |
| remove | 1 | n | n |
| sort | 1 | n | n |

| Tree | Line cost | Number executed | Total cost |
| --- | --- | --- | --- |
| Initialize (self, number, title, prereq) | 1 | n | n |
| Initialize (self, course) | 1 | n | n |
| Initialize (self) | 1 | n | n |
| Insert course | 1 | n | n |
| Search | 1 | n | n |
| Print | 1 | n | n |

Each data structure comes with it’s own advantages and disadvantages. When looking at the vector, it can be challenging because you must parse through the entire list until you find what you are looking for. On the other hand it is great because of the simplicity in adding additional courses and time.

Hash tables on the other hand can search a list much quicker. That does not make it better though because they cannot be sorted which is quite important for the advisors. Due to this reason, it is not a good option.

Binary trees are faster than Vectors when it comes to sorting which is a big advantage for them but still would not be the best option in this scenario.

All that being said, I would recommend using a vector data structure. Vectors are just able to sort the most efficiently and would be the best fit for this project.