## Danny Oxby

## B9017792

Functional Programming Report

Evaluate the functional paradigm of software development, comparing it with imperative, object-oriented development.

|  |  |  |  |
| --- | --- | --- | --- |
| Evaluation of the use of functional programming | 15 | * Outline some features of the paradigm. << paradigm is the programming strategy * Take all of your ideas from taught material. * Provide lists of “good” and “bad” points. * Give a supporting example. | * Discuss what is meant by functional programming and show how it differs from the imperative paradigm. * Go beyond the taught material to find your own sources. * Cite your sources throughout. * Establish your own evaluation criteria, and use them in making your argument. * Use appropriate supporting examples throughout. |
| Discussion of the Clojure language | 15 | * Describe Clojure. * Compare Clojure to another language that you know. | * Provide a detailed description of the language. * Show how the language is used to implement complex software. * Compare Clojure to another functional language and to an imperative one. * Show how the language integrates with Java libraries. |

## Functional Programming Evaluation

The functional paradigm is based around being a pure, side effectless and stateless paradigm, with a emphasis on functions. This means functions should not change properties outside of their scope, and that all variables should be immutable, generating new variables rather than editing existing ones. This is used in JavaScript and Clojure. In contrast to this the imperative paradigm focuses on how to complete a process, and uses state modification to help achieve this. This is used in C and Java. A simple way to show this difference is using an extendable list example. In this example both return a list of integers using a input of length, and loop to add the value to list. In the Imperative example the list is mutable, and is changed as a side effect of the create list method, while in the Functional example the list is internal to the function and doesn’t change any external variables.

|  |  |
| --- | --- |
| Functional using Clojure | Imperative using C++ |
|  |  |