## 19-KelvinZimmerman

| Item | Description | Points Possible | Points | Percent |
|---|---|---|---|---|
| 1 | Temp Converter Input | 2 | 2 | 100% |
| 2 | Temp Converter Output | 2 | 2 | 100% |
| 3 | Temp Converter Correctness | 2 | 2 | 100% |
| 4 | Bitcoin Converter Conversion Rate Message | 2 | 1.5 | 75% |
| 5 | Bitcoin Converter Input | 1 | 1 | 100% |
| 6 | Bitcoin Converter Output | 2 | 2 | 100% |
| 7 | Bitcoin Converter Correctness | 3 | 3 | 100% |
| 8 | Bitcoin Converter Style | 2 | 2 | 100% |
| 9 | Fight Song: Output | 2 | 2 | 100% |
| 10 | Fight Song: Use of Functions | 3 | 2.5 | 83% |
| 11 | Fight Song: Style | 1 | 1 | 100% |
| 12 | Phrase Repeater: Phrase Input | 1 | 1 | 100% |
| 13 | Phrase Repeater: Num of Repetitions Input | 1 | 1 | 100% |
| 14 | Phrase Repeater: Output | 2 | 2 | 100% |
| 15 | Answers Questions in Canvas Submission | 1 | 1 | 100% |
| | **Total** | **27** | **26** | **96.30%** |

Item 1 – Temp Converter Input.....Points: 2/2

Item 2 – Temp Converter Output .....Points: 2/2

Item 3 – Temp Converter Correctness.....Points: 2/2

Item 4 – Bitcoin Converter Conversion Rate Message .....Points: 1.5/2

The rate message should show what the rate of exchange is for 1 bitcoin to dollars. This rate message should be separate from the line of output that tells the user what their bitcoin(s) are worth in U.S. dollars.

Item 5 – Bitcoin Converter Input.....Points: 1/1

Item 6 – Bitcoin Converter Output.....Points: 2/2

Item 7 – Bitcoin Converter Correctness.....Points: 3/3

Item 8 – Bitcoin Converter Style .....Points: 2/2

Code will read better and easier if your comments are indented to the same level as the code lines that they are about.

Item 9 – Fight Song: Output .....Points: 2/2

Item 10 – Fight Song: Use of Functions .....Points: 2.5/3

Generally, a good job of breaking down into functions.

However, the idea here was to use several functions to eliminate redundancy in your code. The fundamental motivation for such an approach is to make it easier to adapt your code to changes.

For example, let's assume that the requirements of your application were to change, and the song author/singer wanted "Go, team, go!" replaced with "Go, side, go!". Then how many places in your version of the code would need to be changed in order to adapt to that change in requirements? I see at least 3 places where "Go, team, go!" appears that would need to be changed.

This may seem trivial in such a simple application, but getting in the habit of eliminating redundancy will help significantly in bigger software projects.

Can you think of a way to eliminate redundant "Go, team, go!" phrasing using functions?

Item 11 – Fight Song: Style ..... Points: 1/1

Item 12 – Phrase Repeater: Phrase Input ..... Points: 1/1

Item 13 – Phrase Repeater: Num of Repetitions Input ..... Points: 1/1

Item 14 – Phrase Repeater: Output ..... Points: 2/2

Item 15 – Answers Questions in Canvas Submission ..... Points: 1/1

## General Comments

Well done!