

This example would score 74 points out of 100.

## P2 Upload a Code Review Document

### Introduction:

This is our first year to play Botball, so we are focusing on the programming. Our robot is a Create with a fixed plow on the back. We intend to push the game pieces around on the board and into the scoring areas. We are mostly using dead reckoning to drive our robot. The code being reviewed is our main program that moves the Create. Ted and Barney wrote all of the code and are the only people on the team who understand programming, so they are doing their own review.

### Best Practices Checklist:

- Code uses functions - Done - Ted 2/14/12
- Code includes comments documenting function's purpose - Done - Barney 3/6/12
- Code includes comments documenting function's arguments and return values - Done - Barney 3/6/12
- Variable names are descriptive and convey use in code - Done - Ted 3/5/12
- No unnamed numeric constants other than 0, 1, or 2 - Done - Ted 3/5/12
- Comments do not contain unused code - Done - Barney 3/6/12

### General Code Analysis

We do not do any error checking detection, or have any recovery logic in our code. Our code is mostly movement functions using the MAV() function in KISS.

We could improve our code by adding recovery logic, and logic at all for that matter. There are some cases where we could add loops, and logic, but up until now, we were just trying to get something to work. Between now and the tournament we hope to add some logic and the associated recovery logic, so we do not get stuck in infinite loops. The easiest way to do this is with loop counters, so all of our loops will have this recovery logic in it.

Since our code is mostly driving functions, it is pretty easy to understand and reuse. We have written functions for driving straight a set distance in inches, and for turning to given degree measurements. These functions are properly named and easy to use. Reuse is easy, since it has become a very high level program.

#### Comment [A1]:

1. Clearly labeled Introduction

#### Comment [A2]:

2. Brief overview of code function

#### Comment [A3]:

3. No Date of Review (-2 points)

#### Comment [A4]:

4. Who wrote the code and who performed review

#### Comment [A5]:

5. Clearly labeled Best Practices Checklist

#### Comment [A6]:

6. Checklist item 1

#### Comment [A7]:

7. Checklist Item 2

#### Comment [A8]:

8. Checklist Item 3

#### Comment [A9]:

9. Checklist item 4

#### Comment [A10]:

10. Checklist item 5

#### Comment [A11]:

11. Checklist item 6 (-2 points)

#### Comment [A12]:

12. Checklist item 7

#### Comment [A13]:

13. Discussion on what checklist requirements code does not meet(-6 points)

#### Comment [A14]:

14. Clearly labeled Reliability (-2 points)

#### Comment [A15]:

15. Discussion on error detection

#### Comment [A16]:

16. How Reliability can be improved

#### Comment [A17]:

17. Clearly labeled Maintainability(-2 points)

#### Comment [A18]:

18. Is code easy to understand, modify and reuse

This example would score 74 points out of 100.

This example would score 74 points out of 100.

We could improve maintainability by comments indicating when and why we added or removed parts of the code. We could also write bigger functions that do more and use logic to simplify our code. That will make it easier to update and maintain at the tournament.

**Comment [A19]:**

19. How maintainability can be improved

Currently our code does effectively perform the task assigned. We go out and bulldoze the poms back to our starting box. This is all done in a matter of seconds. We are so efficient at the moment, that we are looking to add some more functionality to our robot.

**Comment [A20]:**

20. Clearly labeled Effectiveness (-2 points)

**Comment [A21]:**

21. Does code effectively and correctly perform task

We have to be careful when adding on to our robot, that we do not compromise the high efficiency that we already have. If we cannot find a way to score more points quickly and easily then we will go back to our current code. It is better to score a few points well, then to maybe score a lot.

**Comment [A22]:**

22. Improving effectiveness of code

**Comment [A23]:**

23. Excerpt of code used in General Code Analysis section (-5 points)

**Comment [A24]:**

24. Source code or pseudocode used as example for suggestion for improvement in General Code Analysis section (-5 points)

This example would score 74 points out of 100.