My Title

:-) ;-) :-}

1 Introduction

Hello this is my intro Attendance and active participation are expected in every class. Participation includes asking questions, contributing answers, proposing ideas, and providing constructive comments.

Please stay in touch with me, particularly if you feel stuck on a topic or assignment and can't figure out how to proceed. Often a quick e-mail, or face-to-face conference can reveal solutions to problems and generate renewed creative and scholarly energy. It is essential that you begin assignments early.

2 Background

introduce the problem, etc

3 Methodology

Hello this is my methodoloy. I plan to completely ignore all instructions and engage in wild flights of fancy.

Attendance and active participation are expected in every class. Participation includes asking questions, contributing answers, proposing ideas, and providing constructive comments.

Please stay in touch with me, particularly if you feel stuck on a topic or assignment and can't figure out how to proceed. Often a quick e-mail, or face-to-face conference can reveal solutions to problems and generate renewed creative and scholarly energy. It is essential that you begin assignments early.

4 Results and Analysis

So in the text I have a lot of analysis of Figure 1. Perhaps pointing out there there are two different ways of computing the modulus and that it is interesting that the bitwise operatiosn in Java are slower than the mod operator. Hello this is my intro Attendance and active participation are expected in every class. Participation includes asking questions, contributing answers, proposing ideas, and providing constructive comments.

Please stay in touch with me, particularly if you feel stuck on a topic or assignment and can't figure out how to proceed. Often a quick e-mail, or face-to-face conference can reveal solutions to problems and generate renewed creative and scholarly energy. It is essential that you begin assignments early.

I might also have code as in Figure 2. When I have code, if I describe it

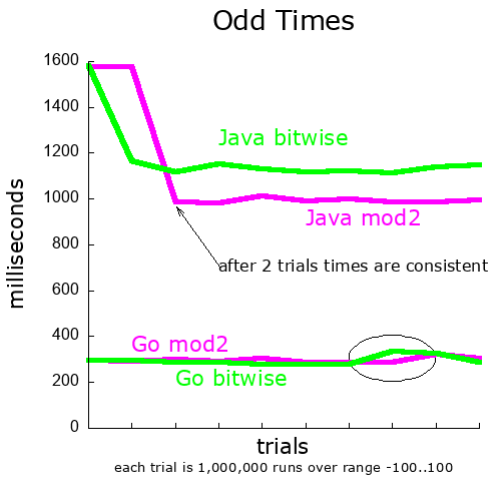

Figure 1: This figure shows the time required for a simple modulus operation in Java and Go. The Y axis gives times in milliseconds to perform 200,000,000 modulus operations. The X axis shows indepented trials. For both java and Go, the "independent trials" all happen in a single unix process. Arguably this figure shoud use points rather than lines as there is no expectation of continuity between trials.

```
1    public int find(int[] A, int f, int idx) {
2        if (idx >= A.length)
3            return -1;
4        if (A[idx] == f) {
5            return idx;
6        }
7        return find(A, f, idx + 1);
8    }
```

Figure 2: An implementation in Java of the "Recursive Linear Search" as described by Cormen (2013).

in the text of my document, the code should be in the text of my document. Alternately, if I do not describe the code explicitly, I might say "... see appendix A so an implementation in Java". So given that my code appears here, assume that I am describing the better linear search algorith, line by painful line. Often it is really useful to number the lines in your code so you can make pithy comments like "..for instance lines 2 and 4 in figure 2 implement the base cases for recursion ..".

5 Summary

Everything worked!

Appendices

A Source Code

I wrote nothing that I want to show.

B Raw data

If I show my raw data, they you could see that the graphs are totally bogus.