Programming for Data Science

Rafael C. Alvarado

5/8/21

Table of contents

1 Preface

Welcome to Programming for Data Science, a collection of materials designed to support the course DS 5100 in the Data Science curriculum at UVA.

In this course, you will develop skills in Python and R Programming, as well as GitHub.

The objective of this course is to introduce essential programming concepts, structures, and techniques.

You will gain confidence in not only reading code, but learning what it means to write good quality code.

Additionally, essential and complementary topics are taught, such as testing and debugging, exception handling, and an introduction to visualization.

This course is project based, consisting of a semester project and final project presentations.

2 Data and Code

An important principle for writing effective and intelligible code is that code should be simple — to quote Einstein, as simple as possible but no simpler.

- A contributing factor to code simplicity is how it is related to the data it is designed to process.
- This relationship depends largely on how the data are structured.
- A program is always written with data in mind — what kind of data it is and how it is structured.

2.1 Simplicity of code follows from the structure of data

There is a view among programmers which, although not orthodoxy, is commonplace.

- It is the idea that the complexity of a program — its algorithms — is a function of the quality of the data structure it processes.
- If a data structure is not well designed, algorithms may be excessively complex and hard to understand.
- However if a data structure is well designed, the algorithms that process them are more robust and intelligible.

2.2 Supporting References

Consider these quotes cited in an essay on Data Structures. by Igor Budasov, reproduced here:

Here's a quote from Linus Torvalds in 2006:

> I'm a huge proponent of designing your code around the data, rather than the other way around, and I think it's one of the reasons git has been fairly successful . . . I will, in fact, claim that the difference between a bad programmer and a good one is whether he considers his [sic] code or his data structures more important. Bad programmers worry about the code. Good programmers worry about data structures and their relationships.

Which sounds a lot like Eric Raymond's "Rule of Representation" from 2003:

> Fold knowledge into data, so program logic can be stupid and robust.

Which was just his summary of ideas like this one from Rob Pike in 1989:

> Data dominates. If you've chosen the right data structures and organized things well, the algorithms will almost always be self-evident. Data structures, not algorithms, are central to programming.

Which cites Fred Brooks from 1975:

> **Representation is the Essence of Programming**
>
> Beyond craftsmanship lies invention, and it is here that lean, spare, fast programs are born. Almost always these are the result of strategic breakthrough rather than tactical cleverness. Sometimes the strategic breakthrough will be a new algorithm, such as the Cooley-Tukey Fast Fourier Transform or the substitution of an n log n sort for an n 2 set of comparisons.
>
> **Much more often, strategic breakthrough will come from redoing the representation of the data or tables.** This is where the heart of your program lies. Show me your flowcharts and conceal your tables, and I shall be continued to be mystified. Show me your tables, and I won't usually need your flowcharts; they'll be obvious.

3 Python Object Types