Abstract—In this article, I will discuss "What Every Computer Science Student Should Know" an article trying to answer its title tough question.

Index Terms—Languages, Computer science education, \LaTeX, paper.

I. BLOG POST: YOU SHOULD NOT BELIEVE EVERYTHING YOU READ ON THE INTERNET WHEN IT COMES TO YOUR STUDIES

A. Introduction

In our current world and society, computer science has taken a central place in a very wide variety of domains and applications. As a result, a computer science student can easily see himself working in pretty much every company, government or organization he can imagine. Trying to come up with a set of knowledge that will cover everything is a mission that the author of "What Every Computer Science Student Should Know" fails to fulfill from my point of view. Still, his attempt is not completely wrong and some of his points are valid; the fact is, in my opinion, it is humanly impossible to gather a list of required knowledge needed in such a broad domain. Furthermore, I will demonstrate that his vision does not fit well neither with a company vision (the author writes about getting a job) nor with a more academic path (graduate school and computer scientists).

B. Company point of view

On his first point, the author tries to emphasis that the resume does not fit well with gauging a programmers ability. First of all, hiring someone is never solely based on the resume itself. The resume IS your portfolio. This is the place to put your achievements and experiences. I would be delighted to see the author having to dig through the public repositories of 50 candidates for a position and to have this only mean to choose the candidate he wants to hire. A code portfolio teaches you very few about the candidate except maybe the programming languages that person knows but it completely forgets the import skills that companies look: teamwork, ability to understand and pick concept quickly, alignment with corporate values and policies, etc. We all know that the skills acquired at school wont exactly be aligned with the job position, the major thing a company wants is to be sure the candidate will be able to fill the blanks quickly and become up to speed fast enough. School projects are nice and can lead to some achievements but they will only, and always be, a pale imitation of what a programmer will expect in the real life with real stakes and pressure. Besides, the scale of the project, the size of the team is also diminished compared to real-life experiences. My take on this point is to tell students to look for internships leading to real project experiences, always exceed expectations when doing school projects to learn more out of them and of course to write a nice resume.

C. Academic point of view

Overall the article digs too deep on every point the author presents. If the student wants to take the academic path, he will need to have a broad knowledge of the computer science but he does not need to be an expert on everything. The goal of a computer scientist (and the author uses this term a lot), is not to know exactly how everything works. What pilot pays attention on how his plane works in details? To continue on the pilot metaphor, he will need to know generally how the plane works to give meaning to the instrument readings but knowing precisely the inner workings of the engine itself is useless knowledge for him. In which case, asking a student to learn how to build a compiler is completely overrated except if the student wants to focus his future area of research on code optimization. Even though if a student wants to later focus his research on this subject, he will have the tools and the mind to get up to speed. I do not want to push the reason too far and say that we do not need to learn anything but just to learn how to learn: A general understanding of the field is mandatory and I insist on general. A computer scientist will be asked to think, invent, imagine, and become an expert in his field. He will not need to know how to perfectly program in ten different languages and compile his own Linux distribution on a server where he would have built his own motherboard from scratch Furthermore, school cannot provide all the knowledge that will be used in a lifetime. School needs to provide an understanding of a particular field that will allow to kickstart the knowledge acquirement process when the student will encounter a new challenge in this field.

D. Conclusion

"What Every Computer Science Student Should Know" is not a bad article, it is just an incomplete utopia and should be considered as a general list on main things you can do in computer science on an undergraduate level. I think if you are looking for a nice job, you would better focus on either being an expert in a particular technology/set of technologies or staying very general to embrace a more high level position.
If academic path is more interesting for you, I think digging deep on a particular subject is what research is made of. Either way, this article does not fit with my vision of how computer science works in our world.

II. PROGRAMMING EXPERIENCE

A. Web programming

Below is presented a table with my experience in various web programming languages.

<table>
<thead>
<tr>
<th>Language</th>
<th>Experience (Years)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML</td>
<td>10+</td>
</tr>
<tr>
<td>CSS</td>
<td>10+</td>
</tr>
<tr>
<td>Javascript</td>
<td>5+</td>
</tr>
<tr>
<td>Python (Django)</td>
<td>.5</td>
</tr>
<tr>
<td>JAVA (Struts)</td>
<td>1</td>
</tr>
<tr>
<td>C# (ASP.NET)</td>
<td>4+</td>
</tr>
</tbody>
</table>

TABLE I: Experience in web programming

B. Desktop programming

Below is presented a table with my experience in desktop programming languages.

<table>
<thead>
<tr>
<th>Language</th>
<th>Experience (Years)</th>
</tr>
</thead>
<tbody>
<tr>
<td>JAVA</td>
<td>2+</td>
</tr>
<tr>
<td>C#</td>
<td>4+</td>
</tr>
</tbody>
</table>

TABLE II: Experience in desktop programming

III. WARMUP PROGRAMMING

Attached, a figure coming from our Warmup programming assignment.

Fig. 1. Throughput depending on write size to fill a 10MB file

IV. REQUIRED CONTENT

My programming experience can be viewed in table I and table II.

To see a figure coming from the previous assignment please refer to figure 1.

V. BIBTeX

The work I am citing include a conference paper about Opportunistic Wireless Broadcast [1], a journal article about Managing Networks through Context [2] and a conference paper about WiiLab [3].

REFERENCES

