Cheatsheet: Guide from a self-taught programmer

Note: If you do not sleep it is going to take exactly 24 hours to go over all the resources. Since it is reasonable to sleep (I do it also), please distribute the load over a few days! 😀

1. Warming up steps. [Optional] Go over an algebra cheat sheet from here. Try to recall some of the concepts and see if you still get the logic. [Optional] Read through some of the algebra examples here.
2. Read this introduction to computer science and then watch a video from here. Familiarize yourself with this computer basics cheat sheet.
3. See here why programming matters. Understand the high-level logic of object-oriented programming from here. Select an object-oriented programming language to learn (Python or Go). Read this post to compare benefits. If you have opted for Python, read from zero to hero here. For Go, see this. No one writes Java any longer but if you decide to have a look at the first steps, see that.
4. [Optional] If you prefer fronted languages, you can read this and that. Then you can continue with VueJS or React. This is not a substitute for object-oriented language to go over.
5. Read every part of this post to understand databases (there are 10 parts - read EVERY). Start interacting with one database, e.g. Redis by reading more here and applying what you learned as shown in this post. Redis is close to me because I work at Redis but it is also great to get up to speed in minutes (feel free to choose e.g. MongoDB or MySQL if you prefer).
6. Understand the high level of networking from my post here. Read also this and that. Skim through a computer network cheat sheet from here and see if you recognize the terms. If you prefer videos you can watch one here.
7. Understand Git basics from here and start a small project. Read here about what a good collaboration in software engineering means. Stick to the Zen commandments of Python from this post while working in engineering (any programming language).
8. Get the main concepts of good software architecture from this post. Keep on reading about microservice software architecture from here and there. Spend some time on understanding computer security in this article.
9. Read the top mistakes in software engineering here and there. Understand scrum and agile from this post. Already start preparing for an interview by looking at the possible questions from here and there.
10. Going on, the main resource to keep close is this. There are all the possible languages you can keep on learning. Follow one or more tech blogs from this list. Follow some of the engineering world authorities from here on Twitter (or other networks).