I am a Strong, Independent Programmer Who Don't Need No Mythical Man

INTRODUCTION

This poster analyzes a transcript taken from CSCI 1300: CS 1: Starting Computing on February 11, 2015. The researcher used ethnographic observation and a voice recorder to collect data.

TRANSCRIPT

This transcript has only one speaker, the TA for the class. He is giving an account of his life story as well as giving the students advice on how to be a successful programmer.

RESEARCH

The outstanding goal of this research is to determine how Computer Science (and other degrees) build and enforce cultural norms and identities within their courses. This poster identifies a few cultural norms about how to be a "good programmer" presented within the course.

"ZUCKERBERG" STORY

Coding Outside of College & Personal Projects

"Find something you love and just start coding.
This is actually how you get good at coding.
I didn't learn coding in college, I've learned very little coding in college."

Very early in his story, the TA gives an account of how he learned code. He lists many activities outside of school that helped teach him coding. He brings up the idea of doing personal projects outside of school. Personal projects are set up as the ideal way to learn code and are mentioned many multiple throughout the transcript.

This position is juxtaposed by the classroom setting he is teaching in. These students are sitting in a traditional lecture hall learning coding. The TA is fulfilling the role of instructor and encouraging the students to continue learning in this setting, while further confusing the idea of what is learning and where it occurs.

"Now we're back to the ***standard, boring, lecture stuff."***

On many occasions, the TA explicitly states that doing personal projects is how to learn CS. He gives many examples in his personal story where he has done personal projects and learned from them. Personal projects are compared to coursework, which is set up as unhelpful and non-informative.

When the TA is talking about personal projects, they are always referred to projects of fun and passion. Students should enjoy doing these projects. However, traditional schooling, such as lecture and college, is difficult, painful, and boring.

"You've got to just keep doing ***personal projects****,*
keep writing ***code for fun****,*
and ***SLOWLY it get's less and less painful."***

GENIUS PROGRAMMER

Showing Mistakes

"Google has this awesome talk called
'The myth of the Genius Programmer'.
And it really is a myth, ***there's no such thing."***

The TA brings up the idea of the Myth of the Genius Programmer. This is a common idea and caricature within the CS community, but would be unknown to the first-year students. The TA sets up a comparison between a genius programmer and a "good programmer". He goes on to define a genius programmer as someone who makes mistakes and isn't afraid to show them. There is no relationship between knowledge and a genius programmer, all references are to making, showing, and learning from mistakes.

The TA encourages the students to "show their mistakes", so they can be a "good programmer" and not a "genius programmer". He gives many personal examples where he has failed, however, all of these are shown as points of pride. The TA is setting himself up as a non-genius programmer, but he is incredibly proud of his mistakes and announces them as a badges of honor. He also makes the claim that a "strong man" is not a genius programmer, marginalizing the women in the class and distancing them from becoming a "good programmer".

"Don't be afraid to fail.
Don't be afraid to SHOW our failures.
Don't be afraid to look like an idiot."

"So I sat there programming the Minecraft mods some more.
And it was ***just terrible code.*** *I wish I still have it, but I don't.*
I would have loved to show it,
cause it was aweful."

'I make mistakes all the time
I just made one in front of Professor *[redacted] the other day,*
that was just, like, completely moronic.
And it's ***something that I should know better****.*
But we all make stupid mistakes."

*"****Strong men don't not make mistakes****, that's not something a human can do.* ***They make mistakes and learn from it****, and that shows a lot more progress than someone who hides their mistakes and pretend they are a genius programmer."*

EGOS

Showing Knowledge

Early on, the TA introduces the persistence of egos within CS. Egos are shown directly connected to the amount of knowledge one has and their attitude towards CS. He gives examples of someone who is egotistical: a jerk, difficult to work with, possibly just starting learning CS, and thinking they know a lot.

"In CS especially,
I don't know why specifically in this field,
but there are some huge, huge egos."

Despite his views on egotistical programmers, when the TA is going through his life story and gives personal programming examples, he demonstrates that he is knowledgeable about CS and states how proud he is of his work. There are many instances where the TA shows both mistakes and achievements within the same story; however, in all of these cases, he is proud of his work and efforts. In the examples he gives, there is a fine line between being egotistical and proud, a genius programmer and someone who completes a project alone In this transcript, the TA muddles these ideas. He aligns himself with the traits of a "good programmer", while unintentionally also fulfilling the "bad" traits. This creates a confusing the idea of what kind of programmer these introductory students should become.

"Stanford has their first three CS courses posted on YouTube . . .
I finished all three courses in, like,
a month and a half."

"I actually wrote the ***first line of code*** *for [redacted] and*
a lot of their code repo is still mine."

"I started writing a game engine . . .
The current code base is something like
25,000 lines of code . . .
It can actually do some pretty cool things and
I pretty proud of this project."

Brittany Ann Kos – *ATLAS Institute* – *University of Colorado Boulder*