Computational Thinking Practices

ScratchEd Webinar Series
Monday, April 25, 2011
7pm-8pm EST
Hosted by Mitch Resnick and Karen Brennan
Computational Thinking
Computational thinking is a fundamental skill for everyone, not just for computer scientists.

Wing, Computational thinking
Computational Thinking

Computational concepts (March 28, Recorded)
Computational practices (Tonight)
Computational perspectives (May 23)
concepts

sequence  conditionals
loops  operators
parallelism  variables
events  lists
practices

incremental/iterative
testing/debugging
reusing/remixing
abstracting/modularizing
perspectives
expressing
connecting
understanding
practices
incremental/iterative
testing/debugging
reusing/remixing
abstracting/modularizing
incremental / iterative

developing a little bit, then trying it out, then developing some more
KB: OK, this is a complicated program. How long have you been working on it?

Scratcher: Maybe three, or maybe two, weeks.

KB: Are you working on it every day?

Scratcher: Like off and on, maybe even a month. Whenever I finished one of the levels, I would show it to my brother.

KB: You talked a bit about how you did a lot of the programming and your brother helped with the concept of the project. What was your process like?

Scratcher: We first came up with it on the way, but for levels 8, 9, and 10 we actually planned beforehand. My brother had this great idea about having level 10 having pins and bowling balls. He said, “That should be level 8” and I said, “No, no that should be level 10, that’s really hard” and he said, “ok, ok, ok”.

KB: Are there any secret codes or do you actually have to play to get to level 10?

Scratcher: You have to play. That’s a really good idea, but now you have to play through the whole game.
testing / debugging

making sure that things work – and finding and fixing mistakes
identify (the source of) the problem

read through your scripts

experiment with scripts

look for examples that work

tell/ask someone else about the problem

try writing scripts again

take a break
reusing / remixing

making something by building on what others – or you – have done
100 levels

welcome to 100 levels. i made originally 51 levels and when i said for every love it i would add a level. but then it got curated and it got to many love its and it got really stressful every ther comment is how do you do this level. how do you do that. and then came the day that i couldnt finish all 100 because my computer fizzed out. well now i would suggest that after you play this game you press love it and then go play some of my other really good games. one for instance is called ninja assissin and it is really fun and before ninja assissin three can come out i need 10 love its.
abstracting / modularizing

building something large by putting together collections of smaller parts
Supporting fluency with CT practices

incremental/iterative
testing/debugging
reusing/remixing
abstracting/modularizing
Thank You!

http://scratched.media.mit.edu
http://events.scratch.mit.edu

Next webinar: Computational Thinking Perspectives
Monday, May 23, 2011
7pm-8pm EST