Jake Luedtke

Alan Kay: 2003 Turing award winner

Alan Kay was born in Springfield, Massachusetts on May 17, 1940. He grew up in a very diverse environment of art, literature, and science. He was taught music early and by the age of three he could already read. Although he was born in Massachusetts his family moved to New York City where he went to Brooklyn Technical High School. After graduating high school he attended the University of Colorado but he left to join the air force before he graduated. It was here where he got his first experience of computing and computers by becoming an IBM 1401 programmer. Also he learned here that programs could be designed with procedures that didn’t know what data was being used, this idea supported the later development of SmallTalk; the first object-oriented language. After the air force he finished his degree in Mathematics, Molecular Biology, Electrical Engineering and a PH.D in Computer Science. He then went on to develop FLEX, a personal computer. And along with leading the team designing SmallTalk, Alan Kay has also been credited with being the father of personal computers.

To recognize his accomplishments in computer science, from leading the team that designed SmallTalk and standardizing personal computers, in 2003 Alan Kay received the Turing Award. For this award people are invited to lecture on a topic that they chose and Alan did his on how “First Courses in Computing Should be Child’s Play”. Alan starts off his lecture with a history of the field of computer science and how it first got named. After this he goes on about architecture and how it’s similar to computer science because you build up a function or bridge by any means, stress it, analyze it, and build more of them from what you learned. He then finishes off this section of his talk with an important note which is that computer science, when being taught to students, should be shown as something that isn’t done yet and that the next generation of students will help it grow further and become even better. The next section of his speech then goes on to describe motivations and how that everything we do is subject to other people’s motivation. Finally the main part of his speech starts off with him saying how when trying to teach kids object-oriented programming people often make it too much complex and there is too much work for the kids to do but if they taught the math better and differently, the kids would have to do less. He says that this problem is the fault of how people delight in the complexity of programs, so they make it hard when it could be more simplified. He then gets off track about his past but then comes back to say that we should have more scratch programming (making your own basic functions, not using predefined ones) because today is too focused on learning the libraries of a program. If people just learn how to use the libraries, he believes they actually don’t know how to code, that they don’t understand the ideas necessary to become a computer scientist. He comes out and says that beginners shouldn’t even be shown the libraries of the language. His style of teaching that he does with children focuses on thinking about ideas, and making pictures of them and this is the way he believes programming should be learned. To end his lecture he says that if people can’t get kids interested in the romance of computer science, then they have failed. It is our duty, he says, to look inside ourselves and find what got us interested in computer science and to tell the children and students of this. It is also important to remind the kids of the fact that computer science has just started. Once they know these two things, it will help them do a better job in computer science than any of the previous computer scientists have done so far.