KURTZ81
Thomos E. Kurtz, 1981: Basic, in History of Programming Languages, Academic Press, 1981, pp. 515-549

"Basic has become the most widely known computer language. Why? Simply because there are more people in the world than there are programmers." Basic was developed by John Kemeny and Thomas E. Kurtz of Dartmouth College in the 1960's. The project actually started with the development of several other languages (DAR-SIMCO, DART, ALGOL30, SCALP and DOPE) where the ideas and language design of Basic were brewed and distilled. The first instruction manual was published in 1964 after about eight years of research and development. The design of Basic and its Operating System went hand-in-hand. As a matter of fact, the language Basic and the time-sharing system in which it resides, were, and still are, inextricably meshed. So, the goal of the project was not just to develop a language, but a system, a user-friendly time-sharing computer system. It was felt that all the languages that existed then were too complex and difficult for a casual programmer to learn. Those languages were also designed for batch processing and thus lacked the features crucial to a time-sharing computing environment. The authors summarized their design goals in the following quote, "In all cases where there is a choice between simplicity and efficiency, SIMPLICITY is chosen." The designers had identified two classes of people: those who know about computing (the programming experts), and the "other" people in the society (the managers, the Greek scholars, the football players etc). One of the design philosophy for Basic was to introduce all these "other" people to programming so as to close the gap between these two groups of people. Most parts of the language was structured with this in mind. For example, short common English words, such as NEW, SAVE, LIST and RUN, were used as system commands so as to encourage computer use by non-experts. Also the user deals directly only with his Basic program. Her needs not even know that such thing as "object code" or "text editor" exists. The user could compile (by typing RUN), receive error messages, edit by typing line-numbered lines, and recompile all within seconds. Execution would automatically occur if there were no errors. It also removes some technical distinctions (such as integer vs. real), and provides some defaults (declarations, dimensions and output formats) where a user probably does not care. One important thing to note is that Dartmouth Basic was always, and still is, intended to be compiled. It was intended to be a compile-and-execute language. Despite this fact, Basic is widely believed to be an interpretive language, since most microcomputer implementations of Basic use interpreters, for reasons unrelated to Basic itself. Basic is closely related to FORTRAN and ALGOL. It obtains its three loop control values (initial, final and step) from FORTRAN and the keywords FOR and STEP from ALGOL. Other traces of FORTRAN includes subroutine definition, and GOTO statement. Its IF-THEN construct is from ALGOL.

Conclusion This paper has definitely illustrated the fact that language design philosophy will influence the language structure and its use. It has succeeded in explaining the reason why Basic is so popular in the non-programming world. Whether you are a Basic programmer or an expert, you will find this paper both enlightening and educative.

Those languages were also designed for batch processing and thus lacked the features crucial to a time-sharing computing environment. The authors summarized their design goals in the following quote, "In all cases where there is a choice between simplicity and efficiency, SIMPLICITY is chosen." The designers had identified two classes of people: those who knew about computing (the programming experts), and the "other" people in the society (the managers, the Greek scholars, the football players etc). One of the design philosophy for Basic was to introduce all these "other" people to programming so as to close the gap between these two groups of people. Most parts of the language was structured with this in mind. For example, short common English words such as NEW, SAVE, LIST and RUN, were used as system commands so as to encourage computer use by non-experts. Also the user deals directly only with his Basic program. He needs not even know that such thing as "object code" or "text editor" exists. The user could compile (by typing RUN), receive error messages, edit by typing line-numbered lines, and recompile, all within seconds. Execution would automatically occur if there were no errors. It also removes some technical distinctions (such as integer vs. real), and provides some defaults (declarations, dimensions and output formats) where a user probably does not care. One important thing to note is that Dartmouth Basic was always, and still is, intended to be compiled. It was intended to be a compile-and-execute language.

Despite this fact, Basic is widely believed to be an interpretive language,since most microcomputer implementations of basic use interpreters, for reasons unrelated to Basic itself. Basic is closely related to FORTRAN and ALGOL. It obtains its three loop control values (initial, final and step) from FORTRAN and the keywords FOR and STEP from ALGOL. Other traces of FORTRAN includes subroutine definition, and GOTO statement. Its IF-THEN construct is from ALGOL.

Conclusion This paper has definitely illustrated the fact that language design philosophy will influence the language structure and its use. It has succeeded in explaining the reason why Basic is so popular in the non-programming world. Whether you are a Basic programmer or an expert, you will find this paper both enlightening and educative.