Robert Chung

Dr. Arias

CSC 3310

Due 10/4

CH 2 Reading + Summary

* Plankalkül  : Developed by a German Scientist during WW II. He worked on it himself, but the language was mostly complete with advanced features like data structures. Each statement consisted of two or three lines.
* Short Code : Developed by John Muauchly in 1949 for the BINAC computer. It was able to store mathematical equations in words. No multiplication code. Implemented with a pure interpreter
* Fortran : Even before the IBM 704 was announced, Fortran was on its way to being developed. It was the first language that delivered syntax error checking. This was the beginning of Fortran aka Fortran 0. Then, Fortran I came out in 1957 with capabilities like I/O formatting, user-defined subroutines, if statements, and do loops. Fortran II compiler came out in 1958 and helped with many bugs and introduced independent compilation of subroutines that made it practical to develop bigger programs. Fortran III was not very popular, but Fortran IV become a huge hit. There were features like explicit type declarations, if constructs, and more. More and more features were built, and went from Fortran 90 to Fortran 95 … all the way to the latest version Fortran 2008. Fortran transformed the way computers were used – it was like a founding father of programming languages.
* Lisp : A functional programming language. It was invented out of need for an AI language. While it wasn’t the first functional programming language, it answered many of the problems that was needed for a functional programming language. Pure Lisps has only two data structures – atoms and lists. It was the dominant functional language from its inception; albeit not as much now.
* Scheme : Was developed in the mid-1970’s in MIT and was a descendant of the Lisp language. It was different in that it was smaller and used static scoping.
* Common Lisp : This was developed to provide portability and a common ground from all the different dialects that emerged from Lisp.
* ALGOL 60 : Result of trying to make a universal programming language for scientific purposes. ALGOL was a descendant of Fortran; however, ALGOL did formalize the concept of a data type, introduced compound statements, allowed for any length variable names, and allowed for any length of an array’s dimension. Furthermore, with ALGOL 60, there were revolutionary features, such as pass by value & pass by reference and block structures. Although it was a good means of communication, ALGOL failed to be the universal programming language it hoped to be. However, it was a crucial language that helped develop subsequent languages such as C++ and Java.
* COBOL : COBOL is used for business purposes. Data division was a cool part of COBOL. It is not as useful for purposes outside of business, but it did get mandated by the Department of Defense for development.
* BASIC : Similar to COBOL but received little respect form computer scientists. Was popular with microcomputers in the late 1970s. It was important in that it was the first widely used program to use the command line and terminal.
* PL/I : Similar to Algol 60 in that it was trying to be a language that had a broad spectrum of applications. It was also similar to Fortran in that it was designed for the IBM machines. It was like a conglomerate of ALGOL 60, Fortran, and COBOL 60 put together. This language is not popular now. It was poorly designed but helped future languages realize that it’s not a good idea to include features just because it was useful in other languages.
* SIMULA 67 : An extension to ALGOL 60. However, one important aspect about SIMULA 67 is that it introduced the concept of classes and abstraction.
* ALGOL 68 : While not as widely used, it was still revolutionary in that it introduced orthogonality. Furthermore, it allowed for user-defined data structures that could fit well with a wide range of problems.
* Pascal : An early descendant of ALGOL 68. Pascal was invented for the purpose of teaching computer science/programming. It lacked several features, but it was designed to help teach concepts in computer science.
* C : C’s characteristics allow for uses in many different applications. One thing that is unique about C is that it lacks complete type checking. For some, this allows for flexibility, but for others, it creates insecurities. It’s popularity was in part due to its integration with the UNIX system
* Prolog : A logic-based programming language; very different from the languages we have seen up to now. There are only a few statements, but they can be complex. It uses fact statements and rule statements to arrive at an answer to a query.
* Ada : it was developed for the Department of Defense (DoD). Ada has major contributions to the world of languages. One contribution is the introduction of packages, which allowed for data abstraction. Another is the extensive development of exception handling. It also allowed for concurrent executions of “tasks.” This was made possible by intertask communication and synchronization. There were also versions like Ada 95 and and 2005 introduced later.
* SmallTalk : First programming language to fully support object-oriented-programming (OOP). Everything is an object. It also uses what is called messages instead of traditional arithmetic and logic expressions. Smalltalk has made huge progress in both GUI and OOP.
* C++ : (My favorite language for now) C++ is both procedural and object-oriented. C++ introduced operator overloading and dynamic binding by virtual functions. Very powerful and widely used. However, it is a big language and suffers some drawbacks of those like PL/I.
* Java : Java was created because there was no satisfactory way of doing certain tasks. Goal to bring greater simplicity and reliability than C++. Does not have poitners, but has reference types similar to pointers. Java supports OOP only.