**Section 2.3 Summary:**

Back in the 1950s computer hardware didn’t have the capability to do computation on floating point numbers (floating point numbers being decimal numbers). Because of this any computation done on a computer was done using programs that used the standard hardware provided by computers at the time providing long computation times of calculations that were done faster by hand in many cases. The introduction of the IBM 704 fixed this processing issue by providing hardware that was able to do floating point calculations and FORTRAN I was the programming language created to work with this new computer.

Since IBM 704 and FORTRAN I in 1955 there has been an explosion of computers and programming languages building off the hardware and programming language provided by IBM which has been instrumental in the development of today’s technology.

FORTRAN as a language has gone through several major updates over the years that have improved upon the language all the way up to 2008 where the current version of Fortran being used resides. Bug fixes, the implementation of variables, of fast compilation, If-Else statements, loops, arrays, dynamic arrays, the ability to create subroutines and pass subroutines to other subroutines allowing the functionality of recursion in programs. Fortran in many ways was the first to implement these features in programming and programming languages today have Fortran to thank for all the features we as programmers today have come to understand as standardized across all languages. You cannot find a programming language used today that doesn’t use all the features listed above that Fortran introduced.

**Pros and Cons of Fortran:**

For my team of scientists, it would depend on the nature of the applications needed in whatever project if I chose Fortran over another modern language like C++. Fortran strictly when it comes to computation provides quick and efficient processing through its programs. However, C++ or Java would provide more flexibility and the possibility for features implanted such as data transferring or adding a GUI interface to the computations outside of standard input and output in a console. Fortran would be superb with working directly with system hardware, however. So, depending on the project I would implement Fortran 95 in leu of other languages that are more standard today.

**Lisp and AI:**

Lisp as a functional language works differently with data structures than typical programming languages. Instead of data being stored linearly through addressing. Data in Lisp is passed via function calls or subroutines recursively to mimic the functionality of linear structuring. This provides one major benefit for AI. Passing multiple arguments to be used in these recursive calls which allows for easy on the fly changes in program directions. Thus, this allows a program to learn and change it’s processing on the fly which is the entire schtick of artificial intelligence. This is what makes lisp so popular with AI coders and why even today it is a standard language for AI.

**ALGOL:**

ALGOL was the first to provide a standard syntax across languages and was the first language designed to be machine independent. However, ALGOL lacks efficient flexibility in its code across hardware and is limited in its inputs/outputs.

From the provided program in ALGOL translated over to Java:

A computer screen with text

Description automatically generated

**Screenshots of Addition program (code provided with submission):**

A computer screen shot of a program

Description automatically generated