Throughout the history of computing, many different programming languages have emerged. The three main paradigms of languages being that of procedural, object-oriented, and functional programming languages. Each language within each paradigm was created with specific intentions and use cases in mind. Therefore, each group has their advantages and disadvantages.

The first paradigm of programming languages to arise was that of procedural programming languages. A procedural language is a type of computer programming language that specifies a series of well-structured steps within its programming context to compose a program (Techopedia, 2018). It contains a systematic order of statements, functions, and commands to complete a computational task or program (Techopedia, 2018). Procedural Languages were evolved from the 1950’s symbolic programming languages like that of assembly language. However, with these assembly languages, it was easy to make mistakes (Past Programming Languages, 2011). Assembly code also lacked portability and was highly dependent on hardware, preventing it from running on different computers (Past Programming Languages, 2011). Thus, procedural languages were created to make programming easier and less prone to errors. The languages that arose were considered high-level languages due to the fact that the program statements were not closely related to the internal characteristics of the computer (Past Programming Languages, 2011). These high-level procedural languages include COBOL, FORTRAN, BASIC, and C (Past Programming Languages, 2011). Compilers were also included in these languages, which in turn improved speed, and introduced abstract ideas. These languages brought the step-by-step method to programming and enabled portability, with the running of programs on different machines.

Procedural languages have advantages and disadvantages, like all programming languages. Advantages include that the programmer has a good level of control without having to know precise target CPU details (Chegg, 2018). Procedural programming also implements the use of compilers and interpreters and has the ability to be strongly modular and structured (Chitnavis, 2014). The source code is also portable and can use different compilers to target different CPUs (Chegg, 2018). Since these languages are older there are many books and references available on well-tried and tested code and algorithms (Chegg, 2018). Disadvantages include the fact that data is exposed to whole programs hence it is lacking data security (Chitnavis,2014). Real world objects are difficult to relate to when using these languages and it is difficult to create new data types (Chitnavis, 2014). Importance is also given to the operation on data rather than the data itself (Chitnavis, 2014). Also, programmers tend to have to specialize in a particular language when it comes to procedural languages (Chegg, 2018). Programmers also need to be extremely precise and knowledgeable about programming instructions which make debugging a program extensive and time-consuming (Chegg, 2018). Procedural programming is also less efficient than other hand-crafted source code written in a low-level language (Chegg, 2018).

Structured programming can be seen as a subset of procedural programming. Its characteristics include removing or reducing the use of global variables, relying on the “GOTO” statement, as well as introducing variables local to blocks such as procedures, functions, subroutines, or methods (Chitnavis, 2014). The approach is associated with the top-down method. Pieces of the problem are broken into smaller pieces and solved one at a time. This method is helpful and is successful, but revisions sometimes must be made later and that brings about issues. Some languages in the structured paradigm include Pascal and C.

Structured Languages also have their pros and cons. Pros include the fact that the complexity is reduced due to the divide and conquer methodology, the logical structures ensure a clear flow of control, and allows for more people to work on an individual project increasing productivity (What are the advantages and disadvantages of structured programming, 2017). Structured languages are also easier to update or fix through the replacement of individual modules rather than a larger amount of code. The ability to either eliminate or reduce the need for the “GOTO” statement (What are the advantages and disadvantages of structured programming, 2017). Disadvantages include the lack of encapsulation, same code repetition, Lack of information hiding, and the meticulous planning involved. There also is not much reusability of code and changes in the data structure or exceeding the complexity level of a project can make programs difficult to manage (What are the advantages and disadvantages of structured programming, 2017).

Object-oriented programming (OOP) is a software programming model constructed the use of objects. This model compartmentalizes data into objects (data fields) and describes object contents and behavior through the declaration of classes (methods) (Techopedia, 2018). Key characteristics of OOP include class, abstraction, encapsulation, inheritance, and polymorphism (Chitnavis, 2014). Abstraction separates the interface from implementation, encapsulation insulates the data through wrapping it using various methods and allows the internal implementation of a class to be shared by specifying what information in an object can be exchanged with others (Chitnavis, 2014). Inheritance enables hierarchical relationships to be represented and refined (Chitnavis, 2014). Polymorphism allows objects of different types to receive the same message and responds in different ways. Java and C++ are two popular examples of OOP languages.

There are many advantages and disadvantages to object-oriented programming languages. Productivity can be improved due to the modular aspect of OOP. There is also improved software maintainability and makes OOP easy to maintain (Advantages and Disadvantages of Object-Oriented Programming, 2018). Another advantage is that OOP languages come with rich libraries of objects, and code developed during projects is also reusable in future projects. This also brings about a lower cost for development, which in turns allows for higher-quality software (increased resources through lower cost and more time effective) (Advantages and Disadvantages of Object-Oriented Programming, 2018). Disadvantages include a steeper learning curve and that more code is needed than procedural programs. OOP is also typically slower than procedure-based programs and requires more instructions to be executed (Advantages and Disadvantages of Object-Oriented Programming, 2018). Object-oriented languages are also not suitable for all types of problems.

There are many programming languages that have been developed over the history of computing and there will be more to come. Looking at three categories of previous and present languages three categories stand out, procedural, structured, and object-oriented languages. All have their specific use cases and are used to solve different problems. The advantages of each type of languages are highlighted and over time the disadvantages are overcome through new developments. Thus, keeping the evolution of programming languages in a continuous cycle seeking advancement.

Work Cited

Ankit92Chitnavis Follow. “Procedural Programming.” *LinkedIn SlideShare*, 30 Jan. 2014, www.slideshare.net/Ankit92Chitnavis/procedural-programming-30623196.

Black, Kelly. *R Object-Oriented Programming: a Practical Guide to Help You Learn and Understand the Programming Techniques Necessary to Exploit the Full Power R*. Packt Publishing, 2014.

Cargill, T. A. “Pi: a Case Study in Object-Oriented Programming.” *Conference Proceedings on Object-Oriented Programming Systems, Languages and Applications - OOPLSA '86*, vol. 6, no. 1, 2018, doi:10.1145/28697.28733.

Chegg. “3. Procedural Concepts.” *Teach ICT - GCSE ICT - Output Devices*, 2018, www.teach-ict.com/as\_as\_computing/ocr/H447/F453/3\_3\_6/types\_language/miniweb/pg3.htm.

Eid, Chaker, and Richard Millham. “Which Introductory Programming Approach Is Most Suitable For Students: Procedural Or Visual Programming?” *American Journal of Business Education (AJBE)*, vol. 5, no. 2, 2012, p. 173., doi:10.19030/ajbe.v5i2.6819.

“Past Programming Languages and Their Influences on Today's Languages and Programming Paradigms.” *Telephones - Engineering and Technology History Wiki*, 2011, ethw.org/Past\_programming\_languages\_and\_their\_influences\_on\_today's\_languages\_and\_programming\_paradigms.

Saylor Foundation. “Advantages and Disadvantages of Object-Oriented Programming (OOP).” *Advantages and Disadvantages of Object-Oriented Programming (OOP)*, The Saylor Foundation, 2018, resources.saylor.org/wwwresources/archived/site/wp-content/uploads/2013/02/CS101-2.1.2-AdvantagesDisadvantagesOfOOP-FINAL.pdf.

Techopedia. “What Is a Procedural Language? - Definition from Techopedia.” *Techopedia.com*, 2018, www.techopedia.com/definition/8982/procedural-language.

Techopedia. “What Is Object-Oriented Programming (OOP)? - Definition from Techopedia.” *Techopedia.com*, 2018, www.techopedia.com/definition/3235/object-oriented-programming-oop.

“What Are Disadvantages of Structured Programming?” *Reference*, IAC Publishing, 2018, www.reference.com/technology/disadvantages-structured-programming-cf52c64ac4c51d97.