CMSC 124 FINAL PROJECT

An Analysis on the Programming Languages: C++ and Prolog

Submitted by

Acopiado, Jhon Mark

Bagaipo, Fayne Roxenne

University of the Philippines

Cebu

2021

January

Contents

1 C++

1.1 Purpose

C++ is a general-purpose programming language which means having a variety of uses, ranging from microcontrollers to huge commercialised applications. The main purpose of C++ was not really to be used for a specific problem rather to support a wide range of application areas Stroustrup, 2013.

C++ is a powerful programming language. Its name was derived from C plus a post-increment operator which basically means that C++ is a higher version of C. It can be said that that C++ is a superset of C, other than irrelevant minor details, most functionalities present in C is also available in C++. The creation of such powerful programming language goes back to the discovery of object-oriented paradigm.

In 1967, the first object-oriented programming language, Simula was introduced. As the name suggest, Simula was mainly used for simulations. It was the language that Bjarne Stroustrup, a software engineer, works with on his Ph. D Thesis. Stroustrup sought the importance of OOP in software development and its possible contributions in the future. Though Simula possess a powerful technique, it was too slow when compared to other programming languages. Thus Stroustrup have the idea of embedding object-oriented paradigm into a faster programming language. C in his time was considered as one of the fastest programming language. Soon after, he began working on C with classes.

https://www.britannica.com/technology/object-oriented-programming

1.2 History

It all started in 1979, when Bjarne Stroustrup starts working on C with classes in his goal of adding OOP in C. The added features available in the new language were user-defined data type class, basic inheritance, inlining, default function arguments, and strong type checking. The compiler used for the new language is called Cfront, and was later abandoned due to issues on feature adoptions of improved C++.

In 1983, C++ was formally introduced, however it was 2 years later (1985) when the language was officially commercialised and its reference book The C++ Programming Language was released. Aside from classes, there are also new features added such as function overloading, the const keyword, and virtual functions.

The first released C++ version was not in standard yet. Thus in 1989, another update of the language was made mainly focusing on adding protected and static members, as well as the improvement of inheritance in classes.

The first international standard for C++ was produced in 1998 by the C++ standards committee, which became known as C++98. Along some added features, the long time invested standard template library was also appended. However, several problems were reported about the upgraded c++, and in 2003 revision was carried out to address these problems.

A major revision was carried out in mid-2011, where a new standard C++ (known as C++11) was concluded. Together with the standardization, new features were added which include regular expressions, a for-loop equivalent to foreach loop in other languages, new container classes, variable templates, and many more. Defects of the C++11 were resolved in 2014 together with minor improvements.

Another major revision of C++ happen in 2017 (C++17). The improvement mainly focuses on ease of using C++ such that writing codes will be more simple, expressive, and clearer.

https://en.cppreference.com/w/cpp/language/history

1.3 Language Features

Simple. C++ allows code to be broken into parts (structured programming). Aside from that, there are also a rich set of library functions. If a programmer started in C, there only a little change needed to transition to C++.

Object Oriented. C++ is basically an object-oriented programming language hence concepts such as abstraction, polymorphism, inheritance, and encapsulation are visible in C++. Trough this development and maintenance of C++ programs are made easier.

Pointers. Pointers gives more power in data manipulation of C++. Pointer variables store the memory address instead of data allowing the direct interaction to the memory.

Memory management. C++ offers dynamic memory allocation which allows the using and freeing of memory during execution. In this way, the program can save memory space by freeing those that are not usable anymore. Also, programs are able to modify the memory size it intend to use for variables (mostly arrays).

Supports concurrency. Modern computers are adopting to the idea of multicores that means computer of today is most likely to have more done one computing core. C++ blends into this trend through its libraries allowing different tasks to be executed at a time.

Regular expression. Standard C++ library supports the use of regular expression by using the ¡regex¿ header. Regular expression is a powerful tool for searching matches out of an array of characters.

Fast. Having most of C's features, C++'s compilation and execution time is faster compared to other languages.

1.4 Paradigms

Paradigms refers to the style or patterns of solving a problems. The same in the case of programming languages, paradigms refers to there style of programming. C++ is a language with various paradigms such as imperative, structured, and object-based programming paradigms.

Imperative programming means that the language explicitly shows how computations are made. Although there are already lots of built-in functions in C++, there are still the need of writing codes on how data is manipulated during execution. In fact, built-in functions in C++ can be rewritten in a way that it will give same output, but it is not recommended to do so because the new function is highly to end up very less efficient.

In structured programming, the language utilizes nested loops, conditionals, and subroutines instead of gotos to describe control flow. These paradigm is supported by the use of for loops, if-else statements, and functions in C++ which manages the flow of the

program.

An object-based programming languages are those that employs objects. These is possible in C++ through its user-defined data type: Class. Users are allowed to instantiate an object from a class. The object will contains all data and procedures inside the class.

Other than the above mentioned programming paradigms, C++ also supports other programming styles such as; generic programming, distributed programming, and procedural programming.

https://cs.lmu.edu/ ray/notes/paradigms/

1.5 Language Evaluation Criteria

Simplicity. . C++ supports operator overloading and feature multiplicity. Furthermore, it also has many constructs making it a deterrent for language's readability. However, these properties of C++ poses a good impression on writability.

Orthogonality. C++ is less orthogonal. Not all data types in C++ can be combined with all arithmetic operators. This is a merit to readability since operators are restricted to be used only for what we expect it to be. Yet, I don't consider such characteristic of C++ to be a hindrance to writability because some combinations of constructs is impractical (e.g. using division for string data type).

Data types. C++ introduces several data types in addition to C. New data types includes boolean, class, and reference data type. Data types in C++ is advantageous to both readability and writability. For instance, the boolean data type allows the programmers to use true and false instead of 1 and 0 which ba

Syntax design.

2 Prolog

2.1 Purpose

Prolog is a computer programming language used for solving problems that involve objects and the relationships between objects. It is a practical and efficient implementation of many aspects of "intelligent" program execution, such as non-determinism, parallelism, and pattern-directed procedure call.

Prolog, which is short for programming logic, is a programming language used in creating artificial intelligence. Prolog is classified as a logic programming language and relies on the user to specify the rules and facts about a situation along with the end goal, otherwise known as a query. After the rules, facts, and goal are stated, the program will then attempt to derive the solution by analyzing the relationship between the three.

2.2 History

Prolog evolved out of research at the University of Aix-Marseille back in the late 60's and early 70's. Alain Colmerauer and Phillipe Roussel, both of University of Aix-Marseille, collaborated with Robert Kowalski of the University of Edinburgh to create the underlying design of Prolog as we know it today. Kowalski contributed the theoretical framework on which Prolog is founded while Colmerauer's research at that time provided means to formalize the Prolog language.

1972 is referred to by most sources as the birthdate of Prolog. Since its birth it has branched off in many different dialects. Two of the main dialects of Prolog stem from the two Universities of its origin: Edinburgh and Aix-Marseille. At this time the first Prolog interpreter was built by Roussel. The first Prolog compiler was credited to David Warren, an expert on Artificial Intelligence at the University of Edinburgh.

To this day Prolog has grown in use throughout North America and Europe. Prolog was used heavily in the European Esprit programme and in Japan where it was used in building the ICOT Fifth Generation Computer Systems Initiative. The Japanese Government developed this project in an attempt to create intelligent computers. Prolog

was a main player in these historical computing endeavours.

Prolog became even more pervasive when Borland's Turbo Prolog was released in the 1980's. The language has continued to develop and be used by many scientists and industry experts. Now there is even an ISO Prolog standardisation (1995) where all of its individual parts have been defined to ensure that the core of the language remains fixed.

2.3 Language Features

In Prolog, facts are declared. These facts are expressed in definite pattern, containing entities and their relation.

1. Unification : The basic idea is, can the given terms be made to represent the same structure.

2. Backtracking : When a task fails, prolog traces backwards and tries to satisfy previous task.

3. Recursion : Recursion is the basis for any search in program.

2.4 Paradigms

Prolog can be viewed as a *descriptive* language as well as a *prescriptive* language. It's approach is more about describing known facts and relationships about a problem, and less about prescribing the sequence of steps taken by a computer to solve the problem. When a computer is programmed in Prolog, the actual way the computer carries out the computation is specified partly by the logical declarative semantics of Prolog.

Prolog, *Programming in logic*, is a declarative programming language that is based on the ideas of logic programming. Prolog was an idea to make the logic look like a programming language and also the idea of the prolog was to allow the logic to be controlled by a programmer to advance the research for theorem-proving.

The logical paradigm is a programming paradigm that is a road-map to machine learning languages. It tends to impose a certain view of the world on its users.

When a paradigm is applied in problem domains that deal with the extraction of knowledge from basic relations and facts, the logic paradigm fits extremely well. In the

more general areas of computation, the logical paradigm seems less natural. In logic (declarative) paradigm, programmers don't focus on the development of an algorithm to achieve any goal rather they focus only on the 'goals of any computation'. They focus only on the logic, the what, that has to be achieved. The technique is referred to as resolution and involves the interpreter that searching through the set of all possible solutions.

Computer scientists are using logic programming methods to try to allow machines to reason because this method is useful for knowledge representation. The logic paradigm is based on first-order predicate logic.

In logic programming, the logic used to represent knowledge is the clausal form that is a subset of first-order predicate logic. First-order logic is used because of its understandable nature as well as it can represent all computational problems. The resolution inference system is used to manipulate knowledge. It is required for proving theorems in clausal-form logic. The essence of logic programming is shown in the below diagram.

In the logic paradigm, a program consists of a set of predicates and rules of inference. Predicates are fact-based statements i.e. water is wet. On the other hand, rules of inference are statements such as "if X is human then X is mortal".

2.5 Language Evaluation Criteria

One basic concern in programming with the use of Prolog is making it as declarative as possible to increase program clarity and readability. A program must be considered as a whole. It's readability is determined by its physical layout and by the choice of names appearing in it.

An important influence in making programs easy to read is the naming of the various objects in the program. The choice of all predicate names, variable names, constants, and structures appearing in the program affect readability. The aim is to emphasize the declarative reading of the program.

3 Feature 1

4 Feature 2

5 Latex

Final project done in latex using Overleaf

Link to overleaf: https://www.overleaf.com/project/61ad5f37b0905efec59fe6f2

Alternative link in github (if overleaf does not work): https://github.com/fabagaipo/cmsc124-finalproject

References

Clocksin, W. F., & Mellish, C. S. (2003). *Programming in prolog*. Springer Science & Business Media.

Stroustrup, B. (2013). *The c++ programming language*. Addisson-Wesley.