International Journal of Information Technology and Management
Vol. 12, Issue No. 01, February-2017, ISSN 2249-4510

Study on the Levels and Generations of Language Programming

Mr. Amarnath Awasthi*

Assistant Professor, Department of Computer Science, Sai Meer Degree College, Uttar Pradesh

Abstract – Computer programmers use programming dialects to create programs. These programming dialects were created over time using the faster dialects called Unique Jergon, which used machine code to program the personal computer; this dates back to 1940-1950. This language is known as a low-level language. The accompanying advance was the second language of 1950-58, which used low-level figurative constructions to handle the rules of machine language. These were later modified into machine code by a construction specialist. That language was a low language at the time. Until then, the dialects of the third period, which are at least called irrefutable programming dialects like C, Pascal, FORTRAN and COBOL, can be followed until 1958-85. These dialects were easier to use, as the processing and machine code were developed at a low level and helped improve quality and productivity. Compilers and interpreters are used to translate the rules from a global language into machine language. Also available for the fourth time are the dialects from 1985. These dialects combine report generators to reduce programming effort. The programming dialects of the fifth period from 1990 are used mainly in the field of artificial consciousness Each period of the programming language has its own characteristics, and the newer dialects are better than the more established ones.

Keywords – Generations, Programming Languages

----------X----------

INTRODUCTION

“Laptops are gone. Whether in department stores, apartments, nuclear power plants, or security. Computers help human energy run a business efficiently. The improvement of personal computers is remarkable, as is the need for great programmers. " my designers who can provide blueprints, at least engineers who can make amazing action plans. Extraordinary designers do not consider themselves more than prepared. "A programmer kills himself well with experience and practice" (Priya & Ranjeet, 2006). Fittingly, this engaging unit familiarizes you with learning how to write computer programs and then moves on to learning how to write computer programs. They correspond to terms that are often used in the programming scene. "Around the completion of the unit it will be conveyed in the language of the designers." We're most excited about the idea of how and probably when scheduling started and where we are in scheduling; that is, the current state of programming in terms of the era of programming languages, consolidating types of programming dialects and different programming styles in all cases called programming principles.

OBJECTIVE

1. Explain the basics of programming.
2. Describe developments in programming languages and influences on the development of language designs.

Computer software

To be performed by personal computers, the estimates must be like a "program". “A program is written in a programming language and the task of making a quote as a program is called programming. In calculations, steps are indicated in the form of direction or verbalization. As a result, a PC program contains a move of instructions that tell the PC what action to take. The programming language used coordinates the possibility of certificates in a program".

Algorithms and their meaning

To use a computer to run processes, you need:

- Design the algorithm to describe how the process will be done.

- Use a programming language to express the algorithm in a program.
- Run the program on your computer

To do this, "it is necessary to understand that the estimates are governed by the programming language used and that each calculation can be transmitted in different programming dialects and performed on different PCs. This is the inspiration that explains why estimating classification is such an integral part of programming. Making an estimate is an intuitive development that is more or less tedious than passing the calculation as a program. Expected skills in estimating design include imagination and information, while there is no expansive rule, which implies that there are no calculations for the calculation plan".

Algorithmic design

- Provides a set of instructions for completing a task.
- Move the problem from the modelling phase to the operational phase
- The set of instructions must be sequential, complete, precise, and with a clear ending point.

In this article, we analyze algorithms considering their structure, composition, and expression in executable form.

Algorithms, programs and programming languages

As we said, an algorithm is a description of how a task or process is performed and there are algorithms to perform practically all kinds of tasks / processes. From building a model airplane to operating a bulldozer.

Figure 1 - Algorithms, programs and programming language

In case of doubt, the cycle represented by an estimate speaks of its current state, of sustainable inputs and products (Goldschlager & Lister, 2012). The cycles carried out on the PC imply an even more regular commitment to the type of data and the output that is implemented is always also in the form of data. For example, consider the path to discovering the trade-off that requires data sources such as step-by-step costs, days worked, etc. and transmitting returns such as fees to be paid, delegate / director responsibilities, and expenses to be deducted. Data sources and returns are important to ensure that the docking processor itself continues to handle a cycle at this point.

Another critical element of cycles is the end. The collaboration may or may not end. It is the clarification that the end or not end of a cycle is one of its great qualities.

The definition of programming

"The instinctive importance of programming does not cause much trouble and interprets the spectacle of requests as code transmitted in a particular programming language. In any case, it is much more embarrassing to portray the inherent essence of writing computer programs. This arises from the need to use certain terms that must be presented from now on (among them: calculation, program, programming language), and from the degree of formalization of the displaced substance. The importance of writing computer programs was learned for the purpose of this broadcast: Numerous activities related to the organization of a high-level machine program (Encyclopedia, 1988). Since the PC is the most common machine, the term will be used throughout the manual to refer to that electronic automated machine. The term program, we derive the set of estimation data of a particular task as a plan explanations and rules in one of the dialects programming (Encyclopedia, 1988). Estimation is described as a representation of the answer to a problem (task) imparted by such exercises that the computer interpreter can understand and perform (Encyclopedia, 1988). So, scheduling has to do with a task that needs to be emphasized in view of a specific goal. This technique is taught in a specific design that is the programming language. This language is created on behalf of PC data programmers (Encyclopedia, 1988). By remembering these assumptions, programming thoughts can be properly organized (Figure 3)".

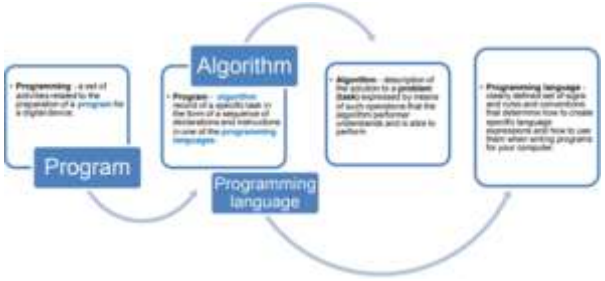

Figure 2: Links between the terms program, programming, algorithm and programming language

Therefore, programming is the representation of a calculation in which the representation of an answer to a particular problem is conveyed using a presentation content medium: the programming language. Dialect programming is explained later in this manual. This justifies the reference regardless of whether the programming language, like any other language, has its own letters, which are systematically Latin letters, Arabic numerals and characters for mixing numbers and reasoning exercises, like phenomenal characters. The commands recorded by the set of letters of the programming language are translated by the PC into its internal language, which is in the form of a double code. In such a data set, there are only two numbers "0" and "1" and through them the information is transferred to the PC. Since it can only transmit two pieces of information with two images, for example, "0" and "1", two digits , the same system combines combined numbers in social events to transmit more ordered messages (Wałaszek, 2018). Thanks to a two-digit social relationship transmitted by a composite digital image, the PC can reproduce the reference action. For example, the unique letters of the letters are recorded in total as follows (BiMatrix, 2013):

"A" - 01100001 "b" - 01100010

An example of the word "hello" is given with "01001000 01100101 01101100 01101100 01101111" and the phrase "hello!" - "01001000 01100101 01101100 01101100 01101111 00100000 01110100 01101000 01100101 01110010 01100101 00100001".

The programming language allows you to create any command to allow a computer to proceed according to this instruction. It is important that the commands are algorithms that they use (Encyclopedia, 1988):

- A clearly defined beginning
- A specified operation: action to start using the algorithm.
- Precise instructions to perform other actions in the exact order
- End date (deadline for completion of the algorithm).

However, “it should be noted that inserting a troubleshooting record into your computer is just coding, not programming (Soukup, 2010). Programming is a much broader term and includes analyzing the problem to be solved and the idea of using available digital technologies to find that solution”.

LOW-LEVEL LANGUAGES

Low-level dialects are machine codes immediately or near the machine. It hardly reflects the technology of major games on a computer. The word "weak" here refers to the degree of weak or absent reflection between language and machine language. These types of dialects are close to the team. The interpreter, like the compiler, the translator for this type of dialect is not necessary. Each low-level project runs instantly on PC computers and requires little storage space compared to undeniable language programs. These types of dialects are simple, but difficult to use due to these many special subtleties that engineers need to remember. A personal computer can simply understand and execute the dialect instructions which are 0 and 1 (binary number frames). Low-level dialects are on the machine and systematically require general information about PC technology (PC equipment and PC design).

There are two types of low-level dialects known as machine dialects and low-level computer constructions.

Machine languages

These are the most important and least important programming dialects. It was the main type of programming language created. The PC can only see the extraordinary signs, which are general current and low current, and the ones and zeros treat them separately. These two digits are known as paired digits. The PC can only display projects written in pairs. These types of projects are known as machine language programs. These are the solitary dialects that are perceived directly by the PCs. Various codes are provided to the PC which will recognize the powered code and transform it into control panels to run as needed. For example:

1010	0011	0001	1001	(*Machine Language*)
ADD	R3,	R1,	R9	(*Assembly Language*)

The language that uses two-digit digits is known as the machine-level language. Machine language has its advantages and disadvantages.

1. First Generation Languages: It was the first language to be programmed by the programmers. It is the first generation computer languages.
2. Machine Dependent: The internal design of every machine (architecture) is different from another machine, so the machine coding also differs. It says that the program instructions which are designed for one kind of machine cannot be utilized on another kind of machine.
3. Fast processing: All the machine instructions are directly recognized by the computing machine; the machine language programs are very quick to run.
4. Error prone: The instructions are written using 0's and 1's, so it is a very cumbersome task. Hence, there are more

chances of error prone codes in these languages.

5. Difficult to use: The binary digits (0 and 1) are used to represent the complete data or instructions, so it is a tough task to memorize all the machine codes for human beings.

6. Difficult of debug: When there is a mistake within the logic of the program then it is difficult to find out the error (bug) and debug the machine language program.

7. Dcult to understand: It is very difficult to understand the existing programs; it requires a great knowledge of machine code with the system architecture. For every machine the machine code is different. We cannot work easily on the other configuration system if we have the knowledge of one machine code.

8. Efficient code for the Machine: The coding of the machine language programs is very efficient for the machine.

9. No need of Translator: The machine code is directly understood by the machines, so no need of the translator.

10. Programmer Requires the Knowledge of Computer Architecture: To program in the machine languages, the programmer need to understand the computer architecture.

11. Need to remember a lot of machine codes: We have to remember a lot of machine codes for programming in these languages.

12. Need to remember all memory addresses: We have to remember all memory addresses for programming.

13. Different Machine language for the different computer Every computer has its own machine language.

Low-level computer design

This is the next level of dialect programming after machine dialects. In low-level computer projects, we use alphanumeric images (such as operands and tasks) instead of two-digit numbers. These cheat sheets can contain up to five of the largest letter combinations, such as: B. MOV, INC, SUB, ADD, MUL, START, etc. Capacity ranges. Projects written on a low-level computer build are known as collection codes. Building low-level computers is more obvious to humans than their archetypal language (machine language). The coding of these two dialects is shown in the table.

Table 1. The machine language code and the corresponding assembly language code

Language Code (Machine) (16-BIT INSTRUCTION SET)	*Assembly Language Code (Equivalent)*
1000000100100101	LOAD R1 5
1000000101000101	LOAD R2 5
1010000100000110	ADD R0 R1 R2
1000001000000110	SAVE R0 6
1111111111111111	HALT

Accounts occupied with writing computer programs in machine language are very boring. At the time of adjusting the dialect programs of the machine, the directions of information things are constantly changing. Engineers systematically process all the program code and the layout of the digits they use. It is the human instinct to be famous for simple deals that are taken in advance. Low-level computer designs are easier to understand. Machine language commands are replaced with memory support commands (activity code) under balanced conditions. Interpreters systematically change the memory assistance code to the same machine code. Engineers use representative locations for time reporting. The engineering agent programs assign machine locations and ensure that distinctive information in the PC memory does not overlap, which is common in machine language program codes. The coding of the acquisition program is usually isolated in several fields, which are divided by spaces or tabs. Here is a general line of the collection code:

[Label] [Op-code] [Operand1], [Operand2] ; Comment

CONCLUSION

This move will help students enjoy the historical background of programming and can further inspire students to see how they can use circumstances and conditions to suggest and plan or improve new dialects.

REFERENCES

[1] Baviskar, D. P. (2009). Fundamentals of programming languages. 1st edition. Technical Publication Pane.

[2] Priya and Ranjeet, (2006). Programming and problem solving through “C” languages. Firewall media.

[3] ISRD, (2008). Programming and PROB solving using C. Tata McGraw-Hill education.

[4] ‘Regan, G. (2012). A brief history of computing. 2nd edition, Springer London.

[5] Mitchell, J. C. (2003). Concepts in programming Languages. Cambridge University Press. Chapter 1.

[6] Gary. M. (2009). Fundamentals of Programming: with Object Oriented Programming. Gary Marrel.

[7] Farrell, J. (2012). Programming Logic and design, Comprehensive. 7th edition. Cengage Learning.

[8] Hanly, J. R. and Koffman, E. F. (2009). Problem solving and program design in C. 6 illustrated. Addison-wesley.

[9] Pratt T W and Zelkowitz M V (1996). Programming Languages. (3rd edition), Prentice Hall of India, New Delhi.

[10] Bird R and Wadler P (1988). Introduction to Functional Programming. Prentice Hall Inc., Englewood–Cliffs, N J, USA.

[11] Ousterhout J K (1998). Scripting: High level Programming for the 21st Century. Computer. (IEEE, U.S.A), Vol.31, No.3.

[12] Wilkes M V (1998). A Revisionist Account of Early Language Development. Computer. (IEEE, U.S.A.), Vol.31, No.4.

[13] Mohan T S (1996). The Java Internet. Resonance. Vol. 1, No. 5.

[14] Shyamsundar R K (1996). Introduction to Algorithms–4 Turtle graphics. Resonance. Vol. 1. No. 9.

Corresponding Author

Mr. Amarnath Awasthi*

Assistant Professor, Department of Computer Science, Sai Meer Degree College, Uttar Pradesh