IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 10, Issue 6, June 2022

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.165

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 6, June 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1006071 |

Evolution and Classification of Programming Language

Shubham Thawait, Anuranjan Shrivastava, Shefali Ghanshani, Shipra Agrawal , Savita Sahu

BE Scholar, Department of Computer Science and Engineering, Government Engineering College, Bilaspur(C.G), India

BE Scholar, Department of Computer Science and Engineering, Government Engineering College, Bilaspur(C.G), India

BE Scholar, Department of Computer Science and Engineering, Government Engineering College, Bilaspur(C.G), India

BE Scholar, Department of Computer Science and Engineering, Government Engineering College, Bilaspur(C.G),India

Assistant Professor, Department of Computer Science and Engineering, Government Engineering College, Bilaspur(C.G), India

ABSTRACT: On this paper we have explained the evolution of computer languages that why we need a programming languages and also shown that the new languages are more robust than the previous languages.It is very hard to learn the machine language which is in the form of 0 and 1 so in initial days to write a simple program was also very hard but day by day evolution of programming language occurs and carry-over the mixed features of older languages, means good features gets added and weak features of older languages gets removed from new version of programming language.In the development of new programming languages we use the features of existing programming languages due to this the consumption of time reduced for the development.In the age of computers the most important tool to communicate with computers are programming languages.

KEYWORDS: Evolution of programming languages, future programming languages, classifications of languages,embedding of new languages.

I. INTRODUCTION

A programming language is a set of commands, instructions, and other syntax use to create a software program.The first functioning programming languages were written in the early 1950s to communicate instructions to a computer. John Mauchly's Short Code,proposed in 1949, it was the first high-level languages which was developed for an electronic computer.Not same as machine code, statements of the Short Code represented mathematical expressions which is in understandable form. Still the program had to be translated into machine code every time when it ran and due to this the process become much slower than running the equivalent machine code due to this we add new features in every update of programming language for this reason evolution takes place and language becomes robust. Languages which is used by the programmers are called "high-level languages." This code can also be compiled into a "low-level language," which is recognized directly by the computer hardware.High-level languages are designed to be easy to read and understand. This allows programmers to write source code in a raw form by using logical words and symbols. Symbols such as != and == this are common operators.There are many high-level languages are similar enough that programmers can easily understand source code written in multiple languages.For example, reserved words like function, while, if, and else are used in most major programming languages.In 1940s Machine-specific assembly language was the first human readable programming language but in 1950s the computer engineers realized that to build entire systems ,assembly language is not suitable and very laborious and thus in 1955 the first modern programming language was born i.e, FORTRAN (FOR-mula TRAN-slation). They are COBOL (COmmon Business Oriented Language), LISP (LISt Processing language), ALGOL (ALGOrithmic Language) followed in the next few years. By using any one of these first four languages every new programming language is developed. In 1964, BASIC (Beginner's all purpose Symbolic Instruction Code) is developed and then C was released in 1969. Unix was famously

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 6, June 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1006071 |

re-written into C, this is the first major Operating System which was not be written in assembly language.Most of the code of Windows and MAC and linux operating systems are written in C programming languages.

II. NEED OF PROGRAMMING LANGUAGE

For the communication to computer or any machine we have to know there language or that machine have to know our language therefore human has created programming language which is used to communicate to the computer language.In initial days human gives the command to the machine in high level language but that language is in the form of 0 and 1 and this was very difficult for human to make simple program.That's why programming language is modified again again to make it robust and user friendly.

III. EVOLUTION OF PROGRAMMING LANGUAGES

Evolution of programming languages in 1970's
• "system programming" languages -- C
– a program ("compiler", "translator") converts into assembler
– efficient and well enough to take on any programming task
writing assemblers, compilers, operating systems
– enormous advantages:
accessible to much wider population of programmers
portable: same program can be translated for different machines
faster, cheaper hardware helps make this happen

```
#include
main() {
int num, sum = 0;
while (scanf("%d", &num) != -1 && num != 0)
sum += num;
printf("%d\n", sum);
}
```

Evolution of programming languages in 1990's
• "scripting", Web, component-based, Java, Python, Perl, Visual Basic, Javascript etc
– writing large programs by using and combining components which are already written.
– these are derived on "virtual machine".
– enormous advantages:
portable: same program we can translate for different machines
faster, cheaper hardware helps make this happen

```
var sum = 0, num; // javascript
num = prompt("Enter new value, or 0 to end")
while (num != 0) {
sum = sum + parseInt(num)
num = prompt("Enter new value, or 0 to end")
}
alert("Sum = " + sum)
```

Evolution of programming languages in 2000's
• so far, maximum of the same
– more specialized languages for specific application areas such as Flash/Actionscript for animation in web pages.
– ongoing refinements / evolution of existing languages these are C, C++, Fortran, Cobol all have new standards in last few years.

• copycat languages
– scripting languages are similar to Perl and Python.
– Microsoft C# strongly related to the Java language.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 6, June 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1006071 |

• using better tools for creating programs without as much programming
– mixing and matching components from multiple languagesis very efficient.

TABLE I. HIGH-LEVEL LANGUAGES.

1) 1951: Regional Assembly Lang. 1983: Ada
2) 1952: Autocode 1984: Common Lisp
3) 1954: IPL (forerunner to LISP) 1984: MATLAB
4) 1955: FLOW-MATIC 1985: Eiffel
(forerunner to COBOL) 1986: Object-C
5) 1957: FORTRAN (First compiler) 1986: Erlang
6) 1957: COMTRAN 1987: Perl
(forerunner to COBOL) 1988: Tcl
7) 1958: LISP 1988: Mathematica
8) 1958: ALGOL 58 1989: FL (Backus)
9) 1959: FACT 1990: Haskell
(forerunner to COBOL) 1991: Python
10) 1959: COBOL 1991: Visual Basic
1959: RPG 1991: HTML
(Mark-up Language)
11) 1962: APL 1993: Ruby
12) 1993: Lua
13) 1962: Simula 1994:CLOS (part of
14) 1962: SNOBOL ANSI Common Lisp)
15) 1995: Java
16) 1963: CPL (forerunner to C) 1995: Delphi
17) 1964: BASIC (Object Pascal)
18) 1964: PL/I 1995: JavaScript
19) 1967: OyII (forerunner to C) 1995: PHP
20) 1968: Logo 1996: WebDNA
21) 1969: B (forerunner to C) 1997: Rebol
22) 1970: Pascal 1999: D
23) 1970: Forth 2000: ActionScript
24) 2001: C#
25) 1972: C 2001: Visual Basic
26) 1972: Smalltalk 2001: .NET
27) 1972: Prolog 2002: F#
28) 1973: ML 2003: Groovy
29) 1975: Scheme 2003: Scala
30) 1978: SQL 2003: Factor
31) 1980: C++ (as C with classes, 2007: Clojure
name changed in July 1983) 2009: Go
32) 2011: Dart

TABLE II. NUMBER OF NEW PROGRAMMING LANGUAGES BORN, IN YEAR-BANDS

S.No.	Year	No. of languages
i)	1951-60	11
ii)	1961-70	12
iii)	1971-80	07
iv)	1981-90	12
v)	1991-00	15
vi)	2001-10	09

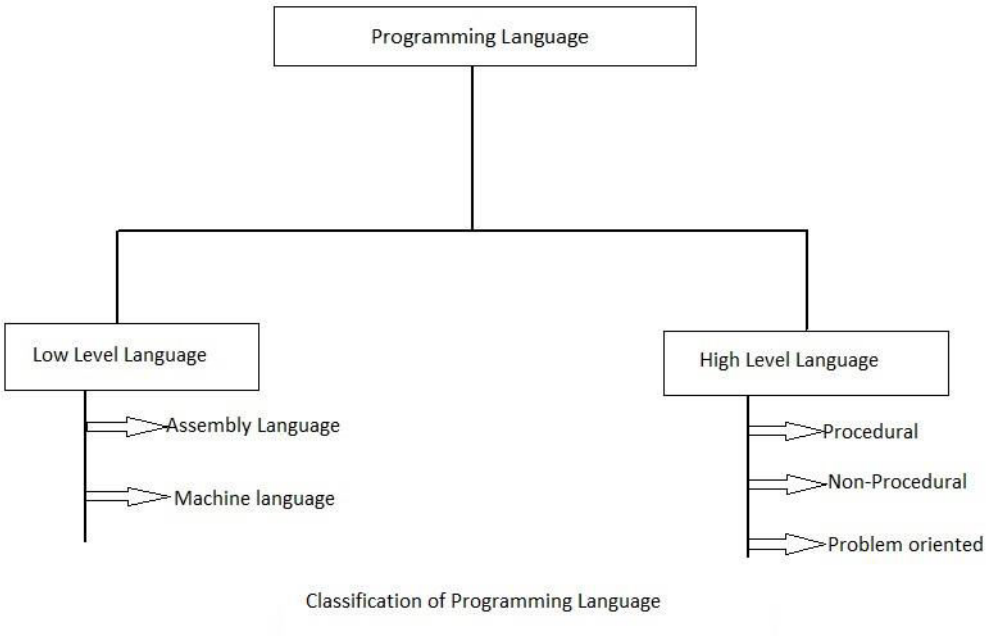
International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 6, June 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1006071 |

PROGRAMMING LANGUAGE CLASSIFICATIONS-



Classification of Programming Language

Mainly there are two classification of programming languages low level language and high level language.A high-level language is one that is user-oriented in that it has been designed to make it straightforward for a programmer to convert an algorithm into program code. A low-level language is machine-oriented. Low-level programs are expressed in terms of the machine operations that must be performed to carry out a task.

TABLE III. CLASSIFICATION OF PROGRAMMING LANGUAGES.

S. No.	Language class
i)	Assembly languages (specific to Processor)
ii)	Authoring languages (HTML, Tutor)
iii)	Compiled languages (Fortran, C, Pascal)
iv)	Command-line interface languages (OS command, shell)
v)	Data flow languages (VHDL, Labview)
vi)	Data oriented Languages (SQL, Foxpro)
vii)	Embedded languages (Lua, other scripting languages.)
viii)	Functional languages (Lisp, Haskell, Erlang)
ix)	Interpreted languages (Java, Python, Ruby, Perl, PHP,PostScript, Python, Lisp, Logo)
x)	Logic based languages (Prolog)
xi)	Macro-languages (ML1, M4, MINIMAC, SAM76)
xii)	Object Oriented languages (C++, Java)
xiii)	Procedural languages (COBOL, Fortran, C)
xiv)	Rule-based languages (Prolog)
xv)	Scripting languages (python, perl, awk, sed)
xvi)	Stack-based languages (Forth, RPL, PostScript, BibTeX)
xvii)	Synchronous languages (Argos, Atom, Averest, LabVIEW, SIGNAL)
xviii)	Syntax handling languages (Ada, Bash, BASIC, C)
xix)	Visual languages (VB .NET, Visual C++)
xx)	XML based languages (AIML, LGML, LOGML)

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 6, June 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1006071 |

important terms=>

Processor- a computer program which processes other computer programs.

Compiler- It is a processor that converts a program written in a programming language i.e. source code into executable code

Interpreter- A processor that accepts a program written in a source code, converts it into some readily executable form and performs a controlled execution. The already executable form code may or may not be object code.

Translator- a processor which is used to convert one source code into another source code .

A. Structures:
it is a data type which contain a group of different data types in it.

Lexical structures: This corresponds to world level structures and decides about the token in the language. This is acknowledged by abstract machines, called finite automata .

Semantic structures:Semantics structures are association of nodes in parse-tree.

Syntax structures: This sentence level structures and expressed by parse-tree, and recognized by the abstract machine push-down automata.

Semantics: Semantics means the meaning of the programming language and it is based on the lexicons .

Ambiguity in languages: When a sentence have more than one syntax trees ,then the language as well as the grammar is called ambiguous.

IV. FUTURE OF PROGRAMMING LANGUAGES

The continuous evolution of technology can be unpredictable.Remember, no people in the past ever thought of having a smartphone till the time it was developed.Obviously, computer programming is part of technological evolution. We can say that nature of logic will stay the same but the syntax or code are changing gradually . Day by day the coding will be easier and it will allow programmers to solve more problems and develop efficient programs.Definitely there will be more programming languages in the future having more functions or blocks of code according to the need of the computer programmers and developers.Is there any need for other new programming language? There are already plenty of programming languages there are procedural languages, functional languages, object-oriented languages, dynamic languages, compiled languages,interpreted languages, dynamic languages, and scripting languages, and it is very hard for any developer to learn all programming languages,keep apart the development.Day by day new languages are developing with surprising frequency.these are designed by students or hobbyists as personal projects . Also the large IT vendors ,small and mid-size companies are also creating languages to fulfill the needs of their industries. Why there is need to develop the existing features? We can say that the language should be powerful and versatile and better than the current languages, not even a single syntax is ideal which can be suited in every purpose. Programming itself is evolving continuously. The rise of the cloud computing, mobility, multi-core CPU's and distributed architectures have created new challenges for developers.But the developers should use the functions of current programming languages then it will be easy to develop new ones.
We have the programming languages which can be used in future and can replace present days
programming languages

i) F#
ii)OPA
iii)FANTOM
iv) Go
v) Dart

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 6, June 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1006071 |

V. CONCLUSION

On this paper we have presented a broad history as well as the evolution of programming languages and the classification of programming language. As well as it has survey of a few experimental languages, which may rule future programming in the multi-core era. In pursuit of the searching goal of an ideal programming language it can be most natural to conclude a most appropriate language.

REFERENCES

1) On the Evolution of Programming Languages K. R. Chowdhary, Professor Dept. of Computer Science and Engineering Jodhpur Institute of Engineering and Technology, Jodhpur.
2) Glass R.L., An Elementary Discussion of Compiler/Interpreter Writing, Computing Surveys, Vol. 1,No. 1.
3) K. R. Chowdhary, Fundamentals of Discrete Mathematical Structures, PHI Learning, Second Ed., 2012, ISBN: 978-81-203-4506-5, pages: 312.
4) Roberto Ierusalims et. al, Passing a Language Through the Eye of a Needle, In Communications of the ACM, July 2011, Vol . 54.
5) P. Bhattacharya and I. Neamtiu. Assessing programming language impact on developmentand maintenance: A study on c and c++. In Proceedings of the 33rd International Conference onSoftware Engineering, ICSE '11, pages 171–180, New York, NY, USA, 2011. ACM.

INNO SPACE
SJIF Scientific Journal Impact Factor

Impact Factor: 8.165

doi cross ref

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details