

INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING & TECHNOLOGY (IJCET)

ISSN Print: 0976-6367
ISSN Online: 0976-6375

Publishers of High Quality Peer Reviewed Refereed Scientific,
Engineering & Technology, Medicine and Management International Journals

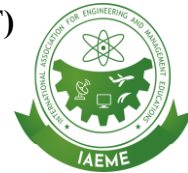


PUBLISHED BY



IAEME Publication
Chennai, India

<https://iaeme.com/Home/journal/IJCET>



PROGRAMMING LANGUAGES CHRONICLE

Hieu D. Vu, Ph.D.

Fort Hays State University

600 Park Street, Hays, KS. 67601, USA.

ABSTRACT

Since the development of computer, scientists, computer experts, and programmers have been trying to find a better way to build software applications. Numerous programming languages were developed, and the newer one has more power, capabilities to facilitate programming. Along with the newer, object-oriented capability languages such as C++, Java, C#, programming paradigm is also changed, from the structured to object-oriented, and to component-oriented programming. This research paper addresses the following questions: What is a programming language? What are special features that a programming language can offer? How they relate to each others? This paper considers books, and research papers on similar topics that might provide a deeper understanding of programming languages, and the answers to these questions above.

Keywords: Computer Languages, Programming, Pascal, Fortran, COBOL, Assembly, Structure Programming, Object-Oriented Programming, Java, C++

Cite this Article: Hieu D. Vu. (2025). Programming Languages Chronicle. *International Journal of Computer Engineering and Technology (IJCET)*, 16(3), 315–329.

https://iaeme.com/MasterAdmin/Journal_uploads/IJCET/VOLUME_16_ISSUE_3/IJCET_16_03_023.pdf

1. Introduction

It is important to know that the lifeblood of computer science is software development. For a computer scientist, programming is a primary tool to communicate with a computer and to program the computer what to do. This is why a part of computer science curricula is the study of programming.

As computing hardware becomes faster, and more powerful, application software also improved and becomes more complicated. New generation of computer's software can do a lot more than the previous generation can. For the programmers, the job of developing the application has become more interesting and challenging.

The heart of software development is problem solving. [1] To develop a software application, the developers or programmers need problem solving skill and a programming language. The programming languages have evolved over the time, from machine language to low-level language (Assembly) then to high-level language such as: COBOL, FORTRAN, PASCAL, etc... and to object-oriented languages such as C++, Java, Visual Basic, and C#. There are numerous programming languages that have been used today such as C++, Java, Visual Basic, COBOL, Assembly, etc... Each programming language has its own characteristics, and usability. This research paper introduces, evaluates some existing popular programming languages. It studies the relationships between low-level (Assembly) and high-level languages such as C++ and Java, and the might be "language of the future" C# (pronounce C-Sharp).

II. PROGRAMMING LANGUAGES (From the early years).

II.1. LOW-LEVEL LANGUAGE

1. ASSEMBLY

The Assembly language is classified as the only low-level language. It is not a complex language but is awkward for human to work with. In the late 1950s, IBM developed the Assembly language as one of the first programming language to run on its System 360. The Assembly language is a one-to-one corresponding with machine language that assembles a series of symbolic representations of machine language operation codes. These symbolic representations are called Mnemonic operation codes. [2]

Since the early years, Assembly language had been used in commerce to develop applications in banking, accounting, and insurance industries. Today, some of these

applications are still using, others were converted to the newer, easier to program, and more powerful programming language.

A sample Assembly program segment.

```
.CODE                ; Code segment
EXTRN putDec:NEAR    ; Use subprogram putDec

datePgm PROC        ; Begin procedure
mov ax, @data        ; Requirement
mov ds, ax           ; Boiler place

    _putStrmsg ; Write output message
    _getDate   ; Get System Date

movsaveDX, dx        ; Save Month and Day
mov al, dh            ; Move month to register AL
mov ah, 0             ; Clear high value in AH

callputDec            ; Write Month
    _putCh '/'        ; Write '/'

mov dx, saveDX        ; Restore Month, Day
mov al, dl            ; Move day to register AL
mov ah, 0             ; Clear high value in AH

callputDec            ; Write Day
    _putCh '/'        ; Write '/'

mov ax, cx            ; Move year to register AX
callputDec            ; Write Year

    _Exit 0          ; Exit program
datePgm ENDP         ; End datePgm procedure

END DatePgm          ; End program
```

; Output of the program datePgm

C:\ASM>datepgm ; Execute the program

Today is: 2/3/2004 ; Output from the program

C:\ASM> ; DOS prompt

As a low-level language, the Assembly allows programmers have a close look at the hardware, especially the Central Processor Unit (CPU) and Memory. Macro `_getDate` gets the system date and the assembler will automatically place the month in register DH, day in DL, and the year in register CX. The other macros `_putCh` used to print the slash (/) will destroy the contents of both registers AX, and DX therefore we need to save values in both registers before calling the macro. To display numeric value such as the month, first we need to move the month into register AX then call an external, built-in subprogram named `putDec` to print the value (whatever value is currently in register AX).

II.2. HIGH-LEVEL LANGUAGES

1. FORTRAN

The name FORTRAN stands for FORMulaTRANslator. IBM developed Fortran in 1957 as a primary language for scientific computation. The early version of Fortran is not really a structured programming language, and it is hard to program. However, with its mathematical computation capability, Fortran was very popular in scientific and engineering fields. In 1966, the standard version of Fortran was recognized then it gradually evolved to newer versions FORTRAN-II, FORTRAN-IV. Each new version made a few changes and added new features to the old one. [3]

A sample Fortran program segment.

Exercise 1-3.3, page 29. (Fortran, A Structured, Discipline Style by Davis, Hoffmann).

- (a) `READ *,A,B,D`
- (b) `PRINT *,B,A,D`
- (c) `PRINT *,A,'IS ALL'`
- (d) `STOP`
- (e) `END`
- (f) `PRINT *,'C: '`
`READ *,C`
`PRINT *,'D: '`
`READ *,D`

The old version of Fortran use simple statements such as READ, PRINT, STOP, END, etc... Just like the Assembly language, Fortran limited the variable's names to eight characters or less thus make it difficult to have a meaningful variable name, and the programmers have to follow the exact format when coding a Fortran program. Today, in many American universities, Fortran is still the required language for course that involved complicated scientific calculation, such as "Numerical Analysis".

2. COBOL

COBOL was developed in 1959 by CODASYL (COntference on DATA SYstem Language) committee. This committee included representatives from academia, user groups, and computer manufacturers. The objective of CODASYL was to develop a standard business-oriented language. As a result, the first Cobol compiler was available in 1960. The name COBOL stands for COmmon Business Oriented Language, it is the most wide spread commercial programming language that dominated the business world for many years. In 1968, the first ANSI (American National Standards Institute) version of Cobol was developed and approved. In 1974, the second version of ANSI Cobol was introduced with more efficient and standardized. IBM also created newer versions of Cobol compilers called COBOL, COBOL-II, and the latest COBOL-MVS to run on their new system IBM/390 mainframe computers. [4]

One special characteristic of a COBOL program is that it looks like an essay, easy to understand even for management and non-technical staff. COBOL dominated the business world for many years. It is a legacy, very well defined language. In COBOL, the procedures are called paragraphs, statements are called sentences, and each statement must begin with a verb.

A sample COBOL program segment.

Partial programming assignment 1.3, page 30. (Structured Cobol Programming by Stern, Stern)

PROCEDURE DIVISION.

A000-MAINLINE.

PERFORM A100-INITIALIZE-RTN THRU A100-INITIALIZE-RTN-EXIT

PERFORM A200-LOAD-TAX-TABLE THRU A200-LOAD-TAX-TABLE-EXIT

PERFORM B000-PROCESS-RECORD THRU B000-PROCESS-RECORD-EXIT

UNTIL NO-MORE-MAST-RECORD

PERFORM Z000-FINAL-RTN THRU Z000-FINAL-RTN-EXIT.

A000-MAINLINE-EXIT.

EXIT.

A100-INITIALIZE-RTN.

OPEN INPUT I-EMPMASTER-FILE
I-TAXTABLE-FILE
OUTPUT O-REPORT-FILE

ACCEPT W-SYSTEM-DATE FROM DATE
MOVE W-SYSTEM-MONTH TO H-MONTH
MOVE W-SYSTEM-DAY TO H-DAY
MOVE W-SYSTEM-YEAR TO H-YEAR

OUTPUT:

03/26/2004 EMPLOYEES INCOME REPORT PAGE:
EMPLOYEE-NAME ANNUAL-SALARY TOTAL-DEDUCTION TAKE HOME INCOME
Adam, John \$65,000.00 \$9,295.00 \$55,705.00
Flores, Tom \$200,000.00 \$5,800.00 \$194,200.00
Nguyen, Hung V. \$17,500.00 \$2,327.50 \$15,172.50

A typical COBOL program is long. There are a lot of codes as seen from the example above. A COBOL program is very well defined, self-documented. It is divided into four divisions: Identification, Environment, Data, and Procedure division. Identification division provides all information about the program, date created, date compiled, and author. Environment division will describe the system to be used to run the program, files in/output. The Data division is important. It describes detail input/output record layouts, and any other data for the program. The Procedure division contains the logic of the program. COBOL program's statement is very much like an English sentence therefore it is not difficult to read and understand a COBOL source program. It is also why COBOL is specially designed for business and it dominated the business world for such a long time.

3. BASIC

The language Basic was originally developed at Dartmouth College in the 60s, and used as a tool for learning computer programming. The name BASIC stands for Beginner's All-purpose Symbolic Instruction Code. As the Basic language became more popular. Newer versions of Basic with new enhanced features were developed by software vendors such as the QuickBASIC, which was developed by Microsoft Corporation, and Turbo BASIC from Borland International. [5]

As one of the early programming language, the Basic is not a structured language. Each Basic statement begins with a line number, which is used as a mechanism in transferring the

control of execution of the program to another statement within the body of a program. To do this, the programmer uses the GOTO (Line number) statement then uses another GOTO (Line number) to return back to the statement after the previous GOTO. If a program contains many GOTO statements, it is difficult for a programmer to follow the logic of the program. Some researchers suggest “GOTO less” programming technique to help programming in Basic easier, and more controllable.

An example of a Basic program.

Programming exercise 1.1, page 29. (Microsoft Basic by Bradley)

```

10 REM PROGRAM TO PRINT PERSON NAME AND AGE.
20 REM WRITTEN BY HIEU VU FOR CS-610
30 PRINT "TEAM RECORD"
40 PRINT
50 PRINT "NAME", "AGE"
60 PRINT
70 PRINT "PATRICIA", "18"
80 PRINT "RONALD", "20"
90 PRINT "MARIA", "21"
100 PRINT "TIEN", "19"
110 PRINT "KENNETH", "22"

```

REM Output from the Basic program segment.

RUN

```

TEAM RECORD
NAME      AGE
PATRICIA  18
RONALD    20
MARIA     21
TIEN      19
KENNETH   22

```

III. OBJECT-ORIENTED PROGRAMMING

1. C++ (Pronounced “C Plus, Plus”).

In the early 1980s, Bjarne Stroustrup of AT&T Bell Laboratories developed the C++ programming language. Stroustrup designed the C++ as an expansion, or a better version of the C language, therefore most components of C is a subset of C++, and so most C programs are

also C++ programs. This is not true in reversed case many C++ programs are definitely not C programs. As a newer programming language, C++ also offers Object-Oriented Programming capability, which is recently developed and became popular, widely used and a very powerful programming technique. [6]

OOP has increased its popularity, since it offers several advantages over the traditional structured procedural programming. A tradition program in C++ is a collection of functions. Each function is a collection of data (local variables) and related statements that work together to perform a single task. In OOP, the programmers concentrate their effort to write *classes*, which are the blueprints for defining objects. An object is a collection of attributes (data) and methods (functions) that work on these data. It is a self-entity that can act on it own. [7]

Object-Oriented capability opened a new era in programming. OOP has some special characteristics and advantages over structured programming.

1. *Encapsulation*. Hiding the inner details. Objects are self-entities that contain data and methods to work on these data. From the example above, the clients (client file) who use the class do not need to find out “How a method works?” but only need to know “What it does? What service it provides?” Encapsulation also protects private data and some methods from “Abuse or misuse”.

2. *Inheritance*. Reusable software. This is the most important character of OOP. A class can inherit attributes (data) and behaviors (methods) from another, existing class just like children inherit characteristics from their parents. Inheritance saves times efforts and cost for a programmer in creating new classes. Instead of starting from scratch, the programmer can reuse some existing classes and add new features or make some adjustments to create a new class as long as they have some similar characteristics.

3. *Polymorphism*. Polymorphism means many forms. This characteristic allows a method behaves differently according to what object it received. A polymorphism method can perform different tasks.

A sample of C++ program segment.

```

/* This program computes an employee's wages for the week *
#include<iostream>           //Include I/O stream
using namespace std;        //Using standard namespace
void calcPay(float, float, float&); //Function prototyping
const float MAX_HOURS = 40.0; //Maximum normal work hours
const float OVERTIME = 1.5;  //Overtime pay rate factor
int main(){                  //Function main()

```

```

float payRate;           //Employee's pay rate
float hours;             //Hours worked
float wages;             //Wages earned
int empNum;              //Employee's ID number
cout<< "Enter employee number: "; //Prompt
cin>> empNum;            //Read employee ID number
cout<< "Enter pay rate: ";   //Prompt
cin>> payRate;            //Read hourly pay rate
cout<< "Enter hours worked: "; //Prompt
cin>> hours;              //Read hours worked
calcPay(payRate, hours, wages); //Compute wages
cout<< "Employee: " << empNum << endl //Output result
    << "Pay rate: " << payRate << endl
    << "Hours: " << hours << endl
    << "Wage: " << wages << endl;
return 0;
}
//*****
void calcPay(float payRate, float hours, float& wages){
    if(hours > MAX_HOURS)
        wages = (MAX_HOURS * payRate) +
            (hours - MAX_HOURS) * payRate * OVERTIME;
    else
        wages = hours * payRate;
}
/* OUTPUT
(a) 327 8.30 48 ---> wages = 431.6
(b) 201 6.60 40 ---> wages = 264
(c) 29 12.50 40 ---> wages = 500
(d) 166 9.25 51 ---> wages = 522.625
(e) 254 7.00 32 ---> wages = 224
*/

```

The C++ program above illustrates a traditional *structured modular programming* technique. Structured programming technique breaks down a large program into many subprograms more manageable called functions therefore structured programming is also called *Functional Decomposition Programming*. The main function calls another function to request a service. Statement `calcPay(payRate, hours, wages)` invokes function `calcPay()` at the bottom of the program and passes three data `payRate`, `hours`, and `wages` to this function. The function `calcPay()` also needs to set up parameters list inside a pair of parentheses to receive the data. This set up the controls for the communication between the functions. Except the function of

type *void*, other functions in C++ have two channels of communication: data passed to the function via parameters list, and something should be returned from the function.

2. JAVA

Java was developed by a team led by James Gosling at Sun Microsystems in the early 1990s. Sun Microsystems was well known for its Sun workstations. The new language was originally designed in 1991 and named Oak for use in embedded consumer electronic applications. In 1995, it was redesigned for developing Internet applications and renamed Java. Java programs can be embedded in HTML (Hyper Text Markup Language) pages and downloaded by a Web browser to provide graphical animation and interaction to Web users.

Java is also a general purpose, high-level programming language, and can be used to develop other applications. As a newer programming language, Java was designed to be a fully object-oriented programming language. Unlike other object-oriented languages that began strictly with structured procedural programming, Java is object-oriented from the start. This could be a problem of difficulty in teaching novice, non-experienced programmers and students in colleges and universities. [8]

A sample Java program.

Programming exercise 1.1 (1-3), page 26,27. (Java Programming by Liang)

```
//This sample Java program prints the words Welcome to Java
public class Welcome{
    public static void main(String[] args){
        System.out.println("\nWelcome to Java!\n");
    }
}
/* OUTPUT
Welcome to Java!
*/
```

This simple Java program shows that Java is fully Object-Oriented language, and all Java programs are classes. The program has only method *main(String[] args)* and it must be embodied inside a class definition *public class Welcome*. Inside the main method, there is a *println("\nWelcom to Java\n");* method that will advance to the next line, after printing the string "Welcome to Java". The main method in Java is different from the main function in C or C++. The main method in Java can receive an array of strings (*String[] args*). This special main method allows a Java program to be run at the *DOS Command* prompt, and the user supplies

values for the string array. For instance, the following command will execute the *Calculator* program and supplies the operator and two operands.

*C:\>java Calculator * 25 15.*

3. C# (Pronounced “C-Sharp”).

C# is the next phase in the evolution of C, C++ programming languages. It was developed by Microsoft Corp., by a team led by Anders Hejlsberg and Scott Wiltamuth as the principle language in the .NET platform. The .NET offers powerful capacity for software development and deployment that included independence from specific language or platform.

As a new language, C# has roots in C, C++, and Java. C# adapted the best features of each language such as object-oriented, and strong graphics capability, and added new features of its own. C# is an event-driven, fully object-oriented, visual programming language. Programmers can write, run, debug C# programs in an Integrated Development Environment (IDE) conveniently.

A sample C# program.

//First C# program to print a string.

```
using System;                                //Using System class package
class Welcome1 {                             //Class Welcome 1
    static void Main(string[] args){         //Main method
        Console.WriteLine("Welcome to C# Programming!");
    }
}
```

The line *using System* is similar to C++ *using namespace std* for simplifying coding the program. Conventional coding in C# is also slightly different from Java or C++, all method names, or class names are capitalized the first letters such as: *Main()*, *WriteLine()*, and *Console*.

Unlike the C, C++ languages, C# like Java accepts numeric inputs as a string then convert the string to numeric data. While Java needs other classes and methods such as *bufferedReader*, and *parseInt()*, *parseDouble()* to get input string and convert to numbers, C# has a better, more simple way to input numeric data as the following illustration program.

```

using System;                                //Using System namespace
class Addition{                               //Class Addition
    static void Main(string[] args){         //Main method
        string numberStr1,                  //First string number
        numberStr2;                          //Second string number
        int number1,                        //Number 1
        number2,                            //Number 2
        sum;                                //Sum of number1 and number2
        //Enter two numbers as strings
        Console.Write("Enter first integer number: "); //Prompt input
        NumberStr1 = Console.ReadLine();           //Input number 1 string
        Console.Write("Enter second integer number: "); //Prompt input
        NumberStr2 = Console.ReadLine();           //Input number 2 string
        //Convert two strings to integers
        number1 = Int32.Parse(numberStr1);          //Convert string 1
        number2 = Int32.Parse(numberStr2);          //Convert string 2

        //Find sum of two numbers
        sum = number1 + number2;                    //sum = number1 + number2
        //Print result
        Console.WriteLine("\nThe sum is {0}.", sum);
    }
}
/* OUTPUT
Enter first integer number: 28
Enter second integer number: 35
The sum is: 63
*/

```

IV. C++ and JAVA RELATIONSHIPS.

To some students with programming experience, the C++ and Java languages appear to be very similar. Many Java statements look exact the same as C++ statements, but we examine the two languages in detail, they have differences in many categories:

a. Primitive data types. Same as C++, Java includes most of the same basic primitive data types such as *int*, *short*, *long*, *float*, *double*, *char*. However, Java adds the *byte* data type. Syntactically, the *String* data type in Java capitalized the letter *S*, and *boolean* instead of *bool* like C++.

b. Main method (function) and other methods. Functions are called methods in Java. The main method in both languages serves the same purpose. It is the entry point for execution

for any C++, or Java programs. The main method in Java has the signature: *public static void main(String[] args)* which enable a Java program to be executed in the *DOS* environment, and the programmer can supply arguments to the Java program. In Java, all methods (all codes) must be coded inside a class.

c. Files and including files. Java uses the *import* statement to include a class or a package of classes. A file in Java also must have the same name as the class name.

d. Class and method definition. All methods (functions) in Java do not require the function declaration, or prototyping. They only have the definition.

e. Pointers. Java does not have pointer type. It handles pointers differently. In fact, all objects in Java are named as references.

f. Method passing parameters. Java has only one mean mechanism of passing parameters called passing by values.

g. Multithreading. This is an advanced special feature of the Java language. Modern computer can run more than one program at the same time. Multithreading allows programs running concurrently that improves the system performance and gives us the feeling that the machine can perform parallel processing. [9]

V. C# and JAVA RELATIONSHIP.

C# is the principle language in the .NET (dot NET) platform, a programming language of choice. C# is built upon the C (high performance), C++ (Object-Oriented), Java (multi-platforms, security), and the Visual Basic (VB, rapid development). As the language of the future, C# is an ideal language for development of software components, Window and Web applications.

C# included most Object-Oriented features of Java such as inheritance and polymorphism, brought back other features from C, C++ like struct (structure), operator overloading, and added new features delegates and events that are very useful in development Windows and Webapplications, since both Windows and Web applications are event-driven.

VI. CONCLUSION.

We are living in a fast changing world. Advanced technology affects our daily lives, and changes everything, the way we do things such as working, learning, playing, entertaining,

and the way we conduct business. For a computer professional, programming is still the most important tool to communicate, to control, and to program a computer to do a specific task. Computer technology changes rapidly, so as the programming languages. Programmers, software engineers or computer professionals need to be aware of the new technology and new programming language, so they can learn, adapt and update their skills easily and quickly according to the change. This research paper provides a general knowledge, information about programming, and programming languages. It also shows the relationships among them, and the advantage, enhance features of the newer language over the older one. This research could be used as a reference, a broad understanding, general concept about the nature, trend of programming language. It would help the students, programmers in predicting, visualization the new features of the future programming languages.

References:

- [1] Riley, David D., (2003). *The Object of JAVA*, Blue Jay Edition. Addition Wesley, Pearson Education, Inc. Page: xxv.
- [2] Cashman, Thomas J., (1981). *Introduction to Computer Programming IBM System/360 Assembler Language (14e)*. Anaheim Publishing Company. Brea, CA. 92621. Page 3.2.
- [3] Davis, Gordon B., Hoffmann, Thomas R., (1978). *FORTRAN A Structured, Disciplined Style*. McGraw-Hill Book Company, New York, NY. 10020. Pages: 5, 42.
- [4] Stern, Nancy, Stern, Robert A., (1991). *Structured COBOL Programming (6e)*. John Wiley & Sons, Inc. Pages: 10, 11, 18.
- [5] Bradley, Julia Case, (1991). *Microsoft BASIC Using Modular Structure (3e)*. Wm. C. Brown Publisher. Dubuque, IA. 52001. Pages: 9 - 11.
- [6] Savitch, Walter. (2004). *Problem Solving with C++, The Object of Programming (4e)*. Addition Wesley. Page 20, 958.
- [7] Gaddis, (2004), *Starting Out With C++ (4e)*, Scott / Jones, Inc., El Granada, California 94018. Page 21
- [8] Liang Daniel D., (2004). *Introduction to Java Programming (4e)*. Prentice Hall. Upper Saddle River, NJ. 07458. Page 4.

[9] Savitch Walter. (2004). *Absolute Java (1e)*, Pearson Education, Inc., Page 958

Citation: Hieu D. Vu. (2025). Programming Languages Chronicle. International Journal of Computer Engineering and Technology (IJCET), 16(3), 315–329.

Abstract Link: https://iaeme.com/Home/article_id/IJCET_16_03_023

Article Link:

https://iaeme.com/MasterAdmin/Journal_uploads/IJCET/VOLUME_16_ISSUE_3/IJCET_16_03_023.pdf

Copyright: © 2025 Authors. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Creative Commons license: Creative Commons license: CC BY 4.0



✉ editor@iaeme.com