|  |
| --- |
| OKESOLA AJIBOLA |
| 17/SCI01/064 |
| CSC302 ASSIGNMENT |

|  |
| --- |
| OKESOLA AJIBOLA  4-27-2020 |

UNSTRUCTURED PROGRAMMING LANGUAGES:

This is the earliest programming paradigm capable of creating algorithms that can recognize or decide other data manipulation rule sets (Wikipedia, 2019) and it had the following features:

* It usually had numbered or labelled lines(Wikipedia, 2019)
* It makes use of unstructured control flow with the help of statements like the ‘go-to’ statements(Wikipedia, 2019)
* Program is not broken down into smaller units or blocks but are written as a single continuous block (OGIRI R , 2018)

Examples of Unstructured programming language: JOSS, FOCAL, TELECOMP, ASSEMBLY LANGUAGES, MS-DOS BATCH FILES, EARLY VERSIONS OF: BASIC, FOTRAN, COBOL (Wikipedia, 2019)

|  |  |  |
| --- | --- | --- |
| NAME | WHAT MAKES IT UNSTRUCTURED | FUN FACTS |
| JOSS-**JOHNNIAC OPEN SHOP SYSTEM**  **By cliff shaw, rand. In 1963** | Each statement starts with line numbers (commands typed without line numbers are executed immediately in ‘direct mode’.)  Has commands such as ‘to’ and ‘do’ which are similar to the ‘go to’ statement | It was one of the first interactive, time sharing languages. Statements could be predicted by Boolean decisions, had a built in editor that could perform direct mode |
| FOCAL- **F**ormulating **O**n-Line **C**alculations in **A**lgebraic **L**anguage  BY Richard Merrill, [DEC](https://en.wikipedia.org/wiki/Digital_Equipment_Corporation) IN 1968 | Each statement starts with line numbers  It made use of ‘go to’ statements as well | It was based on joss and it also had a ‘direct’ mode. It was used to create the early versions of the video games [*hamurabi*](https://en.wikipedia.org/wiki/Hamurabi_(video_game)) and [*lunar lander*](https://en.wikipedia.org/wiki/Lunar_Lander_(video_game_genre)). |
| TELECOMP developed at [Bolt, Beranek and Newman](https://en.wikipedia.org/wiki/Bolt,_Beranek_and_Newman) (BBN) 1964/1965 | It makes use of line statements  It has the commands: ‘to part’ and ‘to step’ which act as ‘goto’ statements | It was inspired by joss. There were three versions of TELECOMP: TELECOMP i, TELECOMP ii, TELECOMP iii |
| Assembly languages | Simple assembly code require line numbers and sophisticated allow the use of labels  It allows for unconditional branches to labels and return | The assembly depends on the machine instruction code. It can also be called symbolic machine code |
| BASIC- BEGINNERS ALLPURPOSE SYMBOLIC INSTRUCTION CODE (INITIAL) by [John G. Kemeny](https://en.wikipedia.org/wiki/John_G._Kemeny) and [Thomas E. Kurtz](https://en.wikipedia.org/wiki/Thomas_E._Kurtz)  in 1964 | EACH LINE HAD A LINE NUMBER  IT HAD THE GOTO COMMAND | The name BASIC is from an unpublished letter by Kurtz. It was influenced by FOTRAN II |
| FORTRAN- Formula Translation by John Backus and IBM in 1957 | The use of GOTO command  It had labels which were used during the GOTO command statements(up till it was termed obsolete in Fortran95) | It was originally designed for scientific and engineering applications. As at 1963, over 40 FORTRAN compilers existed making it considered as the first widely used cross-platform programming language |
| COBOL- common business-oriented language BY CODASYL IN 1959 | It made use of GOTO statements  It makes use of line numbers | It is partly passed on FLOW-MATIC (a programming language)  It has an English like syntax |

The unstructured programming languages were initially adequate enough to perform the tasks that they were designed for, but with success came advancement and the need to make more complex and critical systems arose and it became evident that unstructured programming languages were not going to be suitable for creating these programs.

One of the features that these entire languages exhibit excluding the use of unstructured jumping statements, which disrupt the program flow, is that often times the entire program was written as a single continuous block. It was not separated into several different blocks, which could be accessed individually, and the main program worked on global data, which could pose as a problem.

As a result programs got very complicated, they got more difficult to write and even more difficult to understand leading in several mistakes in code and [Edsger W. Dijkstra](https://en.wikipedia.org/wiki/Edsger_W._Dijkstra), in his publication "[Go To Statement Considered Harmful](https://en.wikipedia.org/wiki/Go_To_Statement_Considered_Harmful)" made it very clear and also coined the term ‘Structured Programming’

STRUCTURED (PROCEDURAL) PROGRAMMING LANGUAGES

This type of programs typically follow a structured control flow that aims at making code clearer and easier to read and understand. It emerged in the late 1950s.

Features:

* Controlled Structures: it has the following structures that governed the program flow: Sequence, Selection, Iteration and Recursion
* Subroutines: These are callable units such as procedures used to allow a sequence be referred to by a single statement (Wikipedia, 2019)
* Block: these enable groups of code or statement to be treated as if they were one statement(Wikipedia, 2019)

Examples are Ada, ALGOL, C, PASCAL, PL/1, CPL, JOVIAL.

|  |  |  |
| --- | --- | --- |
| NAME | WHY IS IT STRUCTURED | FUN FACTS |
| ADA by a team led by [Jean Ichbiah](https://en.wikipedia.org/wiki/Jean_Ichbiah) in 1983 | It has standard constructs and deep level early exists such as while...end loop, if...else...end if, for…end loop etc.  It made use of packages, procedures and functions | Ada was named after Ada Lovelace believed to be the first computer programmer.  It is not based on primitive data types but allows the users to define their own data types |
| ALGOL (Algorithmic Language) in 1958 | It made use of procedures  It made use of code blocks  It has structured control structures | ALGOL introduced [code blocks](https://en.wikipedia.org/wiki/Block_(programming)) and the ‘begin...end’ pairs for delimiting them (Wikipedia, 2020).  Versions are ALGOL 58, ALGOL 60, and ALGOL 68. |
| PASCAL BY [Niklaus Wirth](https://en.wikipedia.org/wiki/Niklaus_Wirth) IN 1970 | It has control structures like if, then, else, while, for, case.  It has several structuring constructs like records, variants, pointers, procedures, etc. | It was based on ALGOL 60. Pascal enabled defining complex datatypes and building dynamic and recursive data structures such as [lists](https://en.wikipedia.org/wiki/List_(abstract_data_type)), [trees](https://en.wikipedia.org/wiki/Tree_(data_structure)) and [graphs](https://en.wikipedia.org/wiki/Graph_(abstract_data_type)) (Wikipedia, 2020) |
| C by Denis Ritchie in 1972 | It allows recursion  All executable code is kept within subroutines (or functions)  It has control flow primitives: IF...ELSE, FOR, DO/WHILE, ETC | It uses semi-colons as statement terminator and curly braces to enclose blocks of statements. It was influenced partially by FORTRAN and ALGOL 68 |
| PL/1 (Programming Language One) By IBM in 1964 | The program consists of a set of procedures  It has several structures to control the program flow such as IF, SELECT, RETURN, STOP, etc. | It has an english-lIke syntax. It was created for scientific, engineering, business and systems programming |
| CPL (Combined Programming Language) by  [Christopher Strachey](https://en.wikipedia.org/wiki/Christopher_Strachey), [David Barron](https://en.wikipedia.org/wiki/David_W._Barron) et al in 1963 | It uses similar symbols as opening and closing tags to enclose blocks of code  Uses subroutines  It has structures such as: WHILE, IF, etc. | ALGOL 60 influenced it. IT AS INTENDED FOR A WIDER APPLICATION AREA THAN SCIENTIFIC CALCULATION |
| JOVIAL (**J**ules' **O**wn **V**ersion of the **I**nternational **A**lgebraic **L**anguage) in 1960 | It has a ‘PROC’ keyword used to define procedures  It has constructs such as for, if...else, etc. | It was made for development of embedded systems. It is partly based on ALGOL (Wikipedia, 2020) |

DEDUCTIONS: the structure and order that STRUCTURED programming brought into the world of computer science helped to broaden the minds and to give inspiration for several other programing paradigms

MODULAR PROGRAMMING LANGUAGES

**Modular programming** is a [software design](https://en.wikipedia.org/wiki/Software_design) technique that emphasizes separating the functionality of a [program](https://en.wikipedia.org/wiki/Computer_program) into independent, interchangeable **modules**, such that each contains everything necessary to execute only one aspect of the desired functionality. (Wikipedia, 2020)

|  |  |  |
| --- | --- | --- |
| NAME | WHAT MAKES IT MODULAR | FUN FACTS |
| F by ‘The Fortran Company’ | F places a heavy emphasis on [modular programming](https://en.wikipedia.org/wiki/Modular_programming). Modules in F are called "programs" and Placing procedures outside of a module is prohibited (Wikipedia, 2020) | F was developed as a modern [Fortran](https://en.wikipedia.org/wiki/Fortran), thus making it a subset of [Fortran 95](https://en.wikipedia.org/wiki/Fortran_95). All procedures in F are external by default, and require a result clause that returns the value of a function (Wikipedia,2020) |
| Modula by Niklaus Wirth in 1975 | It has a [module system](https://en.wikipedia.org/wiki/Modular_programming), used for grouping sets of related declarations into program units; hence the name *Modula* | It is a descendant of the Pascal programming language |
| Oberon by Niklaus Wirth in 1987 | The program is grouped into modules which have a beginning and end and have declarations and procedures in between | The name is from the moon of [Uranus](https://en.wikipedia.org/wiki/Uranus), [Oberon](https://en.wikipedia.org/wiki/Oberon_(moon)). Oberon is designed with a motto attributed to [Albert Einstein](https://en.wikipedia.org/wiki/Albert_Einstein) in mind: “Make things as simple as possible, but not simpler.”(Wikipedia,2020) |
| OCaml (Objective *Categorical Abstract Machine Language) in 1996* | OCaml features a [static](https://en.wikipedia.org/wiki/Semantic_analysis_(computer_science)) [type system](https://en.wikipedia.org/wiki/Type_system), [type inference](https://en.wikipedia.org/wiki/Type_inference), [parametric polymorphism](https://en.wikipedia.org/wiki/Parametric_polymorphism), [tail recursion](https://en.wikipedia.org/wiki/Tail_recursion), [pattern matching](https://en.wikipedia.org/wiki/Pattern_matching), first class lexical [closures](https://en.wikipedia.org/wiki/Closure_(computer_science)), [functions (parametric modules)](https://en.wikipedia.org/wiki/Function_object#Other_meanings), [exception handling](https://en.wikipedia.org/wiki/Exception_handling), and incremental generational [automatic garbage collection](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)). | It was created by by [Xavier Leroy](https://en.wikipedia.org/wiki/Xavier_Leroy), Jérôme Vouillon, [Damien Doligez](https://en.wikipedia.org/wiki/Damien_Doligez), Didier Rémy, Ascánder Suárez, and others. |

OBJECT ORIENTED PROGRAMMING LANGUAGES

Object-oriented programming (OOP) is a computer-programming model that organizes software design around data, or [objects](https://searchapparchitecture.techtarget.com/definition/object), rather than functions and logic. An object can be defined as a data field that has unique attributes and behavior. (techtarget.com, 2020)

Object oriented programming has many key elements such as

* Abstraction: hiding program complexity by limiting access to underlying implementation details
* Inheritance: [Inheritance](https://en.wikipedia.org/wiki/Inheritance_(computer_science)) allows one data type to acquire properties of other data types (Wikipedia, 2020) involves the ability of a particular object or class to gain properties and functions from another object or class
* Encapsulation: [Encapsulation](https://en.wikipedia.org/wiki/Information_hiding) is the hiding of information to ensure that data structures and operators are used as intended and to make the usage model more obvious to the developer. (Wikipedia, 2020). This is closely linked with Abstraction, it involves restricting access to particular details and implementations, it is often times linked with the usage of access modifiers
* Polymorphism: this involves the ability of methods of elements in the program to take different forms

|  |  |  |
| --- | --- | --- |
| Name | What makes it OOP | Fun Facts |
| C++ In 1979, by [Bjarne Stroustrup](https://en.wikipedia.org/wiki/Bjarne_Stroustrup) | C++ provides the ability to define classes and functions as its primary encapsulation mechanisms. Within a class, members can be declared as either public, protected, or private to explicitly enforce encapsulation  [Multiple inheritance](https://en.wikipedia.org/wiki/Multiple_inheritance) is a C++ feature not found in most other languages, allowing a class to be derived from more than one base class  C++ supports several kinds of *static* (resolved at [compile-time](https://en.wikipedia.org/wiki/Compile-time)) and *dynamic* (resolved at [run-time](https://en.wikipedia.org/wiki/Run_time_(program_lifecycle_phase))) [polymorphisms](https://en.wikipedia.org/wiki/Polymorphism_(computer_science))  Stroustrup describes C++ as "a light-weight abstraction programming language  (Wikipedia, 2020) | It was designed as an extension of C or ‘C with classes’. It is a general-purpose programming language. One distinguishing feature of C++ classes compared to classes in other programming languages is support for deterministic [destructors](https://en.wikipedia.org/wiki/Destructor_(computer_science)), which in turn provide support for the [Resource Acquisition is Initialization](https://en.wikipedia.org/wiki/Resource_Acquisition_is_Initialization) (RAII) concept.(Wikipedia, 2020) |
| Ruby by Yukihiro Matsumoto in 1995 | every value is an object, including classes and instances of types that many other languages designate as primitives  it has [classes](https://en.wikipedia.org/wiki/Class_(computer_science)) with inheritance,(Wikipedia, 2020) | In Ruby, classes are never closed: methods can always be added to an existing class. Adding methods to previously defined classes is often called [monkey-patching](https://en.wikipedia.org/wiki/Monkey_patch).(Wikipedia, 2020) |
| SmallTalk (Smalltalk-80) in 1972 | It adopted a  [Simula](https://en.wikipedia.org/wiki/Simula)-like class [inheritance](https://en.wikipedia.org/wiki/Inheritance_(computer_science)) model of execution  The central concept of Smalltalk is objects | It was designed by [Alan Kay](https://en.wikipedia.org/wiki/Alan_Kay), [Dan Ingalls](https://en.wikipedia.org/wiki/Dan_Ingalls), [Adele Goldberg](https://en.wikipedia.org/wiki/Adele_Goldberg_(computer_scientist)), [Ted Kaehler](https://en.wikipedia.org/wiki/Ted_Kaehler), [Diana Merry](https://en.wikipedia.org/wiki/Diana_Merry), Scott Wallace |
| Simula (Simula 67) | Simula 67 introduced objects, classes, inheritance, subclasses, virtual procedure (Wikipedia, 2020) | Simula is considered the [first object-oriented programming language](https://en.wikipedia.org/wiki/Object-oriented_programming#History). As its name suggests, Simula was designed for doing [simulations](https://en.wikipedia.org/wiki/Simulation)(Wikipedia,2020) |
| BETA by [Bent Bruun Kristensen](https://en.wikipedia.org/w/index.php?title=Bent_Bruun_Kristensen&action=edit&redlink=1), [Ole Lehrmann Madsen](https://en.wikipedia.org/w/index.php?title=Ole_Lehrmann_Madsen&action=edit&redlink=1), [Birger Møller-Pedersen](https://en.wikipedia.org/wiki/Birger_M%C3%B8ller-Pedersen), [Kristen Nygaard](https://en.wikipedia.org/wiki/Kristen_Nygaard) | classes and procedures revolve around the same concept and classes are defined as attributes of objects  It has strong abstraction mechanisms (Nice, 2017) | Classes are defined as properties/attributes of objects. This means that a class cannot be instantiated without an explicit object context. (Wikipedia, 2020) |
| JAVA by  [James Gosling](https://en.wikipedia.org/wiki/James_Gosling)  in 1995 | All code is written inside classes, and every data item is an object, with the exception of the primitive data types (Wikipedia, 2020)  It makes use of access modifiers for encapsulation reasons  Java supports inheritance but not multiple-inheritance | Java was originally designed for interactive television, but it was too advanced for the digital cable television industry at the time. (Wikipedia, 2020) It has a WORA (Write Once Run Anywhere) functionality so Java software runs on everything |

ASPECT ORIENTED PROGRAMMING LANGUAGES

**aspect-oriented programming** (**AOP**) is a [programming paradigm](https://en.wikipedia.org/wiki/Programming_paradigm) that aims to increase [modularity](https://en.wikipedia.org/wiki/Modularity_(programming)) by allowing the [separation of](https://en.wikipedia.org/wiki/Separation_of_concerns) [cross-cutting concerns](https://en.wikipedia.org/wiki/Cross-cutting_concern). It does so by adding additional behavior to existing code (an [advice](https://en.wikipedia.org/wiki/Advice_(programming))) *without* modifying the code itself, instead separately specifying which code is modified via a "[point cut](https://en.wikipedia.org/wiki/Pointcut)" specification (Wikipedia,2020)

|  |  |  |
| --- | --- | --- |
| NAME | WHAT MAKES IT ASPECT-ORIENTED | FUN FACTS |
| Aspect J by eclipse in 2001 | AspectJ lets programmers define special constructs called [*aspects*](https://en.wikipedia.org/wiki/Aspect-oriented_software_development#Aspects).  It also allows a programmer to add methods, fields, or interfaces to existing classes from within the aspect. (Wikipedia, 2020) | All valid Java programs are also valid AspectJ programs |
| Aspect C++ in 2001 | lt calls to a specific function can be traced using an aspect, rather than inserting 'cerr' or print statements in many places: (Wikipedia, 2020) | It has a [source-to-source compiler](https://en.wikipedia.org/wiki/Source-to-source_compiler), which translates AspectC++ [source code](https://en.wikipedia.org/wiki/Source_code) into compilable C++. (Wikipedia, 2020) |
| E by Y oav Hollander in 1992 | Aspect-oriented programming in *e* allows verification engineers to structure their test bench in aspects. An object is therefore the sum of all its aspects, which may be distributed over multiple files.  Subtyping is an aspect-oriented mechanism | **E** is a [hardware verification language](https://en.wikipedia.org/wiki/Hardware_Verification_Language) (HVL) which is tailored to implementing highly flexible and reusable verification [test benches](https://en.wikipedia.org/wiki/Test_bench). (Wikipedia, 2020) |

Based on domains

SCIENTIFIC DOMAIN:

The first digital computers, which appeared in the 1940’s, are invented for scientific applications. In the Second World War, the US defense department wanted a computer that could easily calculate the values in their applications like trajectory of missiles.

Characteristics:

1. Consists of simple data structures
2. Can do large number of floating point arithmetic.

SOME LANGUAGES UNDER THIS DOMAIN INCLUDE

* **F** is a [modular](https://en.wikipedia.org/wiki/Modular_programming), compiled, numeric programming language, designed for [scientific programming](https://en.wikipedia.org/wiki/Computational_science) and scientific computation.[[1]](https://en.wikipedia.org/wiki/F_(programming_language)#cite_note-js-1) F was developed as a modern [Fortran](https://en.wikipedia.org/wiki/Fortran), thus making it a subset of [Fortran 95](https://en.wikipedia.org/wiki/Fortran_95)
* **FOTRAN:** Originally developed by [IBM](https://en.wikipedia.org/wiki/IBM)[[3]](https://en.wikipedia.org/wiki/Fortran#cite_note-Where-3) in the 1950s for scientific and engineering applications, FORTRAN came to dominate this area of programming early on and has been in continuous use for over six decades in computationally intensive areas such as [numerical weather prediction](https://en.wikipedia.org/wiki/Numerical_weather_prediction), [finite element analysis](https://en.wikipedia.org/wiki/Finite_element_method), [computational fluid dynamics](https://en.wikipedia.org/wiki/Computational_fluid_dynamics), [computational physics](https://en.wikipedia.org/wiki/Computational_physics), [crystallography](https://en.wikipedia.org/wiki/Crystallography) and [computational chemistry](https://en.wikipedia.org/wiki/Computational_chemistry).
* **IDL**, short for **Interactive Data Language**, is a [programming language](https://en.wikipedia.org/wiki/Programming_language) used for [data analysis](https://en.wikipedia.org/wiki/Data_analysis). It is popular in particular areas of science, such as [astronomy](https://en.wikipedia.org/wiki/Astronomy), [atmospheric physics](https://en.wikipedia.org/wiki/Atmospheric_physics) and [medical imaging](https://en.wikipedia.org/wiki/Medical_imaging)
* The **Klerer–May System** is a [programming language](https://en.wikipedia.org/wiki/Programming_language) developed in the mid-1960s, oriented to [numerical](https://en.wikipedia.org/wiki/Numerical_analysis) [scientific](https://en.wikipedia.org/wiki/Computational_science) programming, whose most notable feature is its two-dimensional syntax based on traditional [mathematical notation](https://en.wikipedia.org/wiki/Mathematical_notation).

BUSINESS DOMAIN:

The use of computers for business applications began in 1950’s. Characteristics: The business people wanted a programming language that can

1. produce elaborate reports
2. Have precise way of describing and sorting decimal numbers and character data.
3. The ability to specify decimal arithmetic operations.

Some languages under this domain are:

* **COMTRAN** (COMmercial TRANslator) is an early [programming language](https://en.wikipedia.org/wiki/Programming_language) developed at [IBM](https://en.wikipedia.org/wiki/IBM). It was intended as the business programming equivalent of the scientific programming language [FORTRAN](https://en.wikipedia.org/wiki/FORTRAN)
* **DIBOL** or **Digital's Business Oriented Language** is designed for use in [Management Information Systems](https://en.wikipedia.org/wiki/Management_Information_Systems) (MIS) software development. It has a syntax similar to [FORTRAN](https://en.wikipedia.org/wiki/FORTRAN) and [BASIC](https://en.wikipedia.org/wiki/BASIC_programming_language), along with [BCD](https://en.wikipedia.org/wiki/Binary-coded_decimal)arithmetic. It shares the [COBOL](https://en.wikipedia.org/wiki/COBOL) program structure of data and procedure divisions.
* **Harbour** is a computer [programming language](https://en.wikipedia.org/wiki/Programming_language), primarily used to create database/business programs
* **RPG (Report Program Generator)** is a [high-level programming language](https://en.wikipedia.org/wiki/High-level_programming_language) (HLL) for [business applications](https://en.wikipedia.org/wiki/Business_software). RPG is one of the few languages created for [punched card](https://en.wikipedia.org/wiki/Punched_card) machines, which are still in common, use today, because it has evolved considerably over time.
* **COBOL: (**"common business-oriented language") is a [compiled](https://en.wikipedia.org/wiki/Compiled) English like [computer programming language](https://en.wikipedia.org/wiki/Computer_programming_language) designed for business use. It is [imperative](https://en.wikipedia.org/wiki/Imperative_programming), [procedural](https://en.wikipedia.org/wiki/Procedural_programming) and, since 2002, [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming). COBOL is primarily used in business, finance, and administrative systems for companies and governments. COBOL (Common business oriented language) was the first high-level language for business with the advent of minicomputers small business people are used computers and some of its tools like spreadsheet systems and database system.
* **LINC** ("Logic and Information Network Compiler"): LINC was originally developed as a short cut (or template) by two programmers to reproduce and automate the production of computer applications for different companies that had similar requirements and specifications. The requirements were similar, because the companies followed a common, generic, business model.
* **OpenEdge Advanced Business Language**, or **OpenEdge ABL** for short, is a business application development language created and maintained by [Progress Software Corporation](https://en.wikipedia.org/wiki/Progress_Software) (PSC). The language, typically classified as a [fourth-generation programming language](https://en.wikipedia.org/wiki/Fourth-generation_programming_language), uses an English like syntax to simplify software development

(Wikipedia, 2020)

ARTIFICIAL INTELLIGENCE:

In the years of 1960, artificial intelligence played a very important role.

Characteristics: AI is a broad area of computer applications characterized by:

1. use of symbolic rather than numeric computation i.e. it must contain symbols, names
2. more flexible language for symbolic computation
3. The ability to create and execute some code segments during execution

.The main languages that were developed during this time were the LISP and PROLOG. These were mainly used for logic programming like designing moves of chess.

EXAMPLES OF LANGUAGES TAT FALL UNDER THIS DOMAIN:

* [IPL](https://en.wikipedia.org/wiki/Information_Processing_Language)  was the first language developed for artificial intelligence. It includes features intended to support programs that could perform general problem solving, such as lists, associations, schemas (frames), dynamic memory allocation, data types, recursion, associative retrieval, functions as arguments, generators (streams), and cooperative multitasking.
* [Lisp](https://en.wikipedia.org/wiki/Lisp_(programming_language)) is a practical mathematical notation for computer programs based on [lambda calculus](https://en.wikipedia.org/wiki/Lambda_calculus). [Linked lists](https://en.wikipedia.org/wiki/Linked_list) are one of the Lisp language's major [data structures](https://en.wikipedia.org/wiki/Data_structure), and Lisp [source code](https://en.wikipedia.org/wiki/Source_code) is itself made up of lists. As a result, Lisp programs can manipulate source code as a data structure, giving rise to the [macro](https://en.wikipedia.org/wiki/Macro_(computer_science)) systems that allow programmers to create new syntax or even new languages embedded in Lisp. There are many dialects of Lisp in use today, among which are [Common Lisp](https://en.wikipedia.org/wiki/Common_Lisp), [Scheme](https://en.wikipedia.org/wiki/Scheme_(programming_language)), and [Closure](https://en.wikipedia.org/wiki/Clojure).
* [Smalltalk](https://en.wikipedia.org/wiki/Smalltalk) has been used extensively for simulations, neural networks, machine learning and genetic algorithms. It implements the purest and most elegant form of object-oriented programming using message passing.
* Prolog is a [declarative](https://en.wikipedia.org/wiki/Declarative_programming) language where programs are expressed in terms of relations, and execution occurs by running *queries* over these relations. Prolog is particularly useful for symbolic reasoning, database and language parsing applications. Prolog is widely used in AI today.
* [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) is widely used for artificial intelligence, with packages for several applications including General AI, [Machine Learning](https://en.wikipedia.org/wiki/Machine_Learning), [Natural Language Processing](https://en.wikipedia.org/wiki/Natural_Language_Processing) and [Neural Networks](https://en.wikipedia.org/wiki/Neural_Network).[7]

(Wikipedia, 2020)

SYSTEM SOFTWARE DOMAIN:

In the 1960’s and 1970’s system software was used. The OS & all of the programming support tools of a computer system are collectively known as its system software. System software is almost used continuously. Characteristics: The system software must have the following:

1. As it is continuously used it must be efficient.
2. It must have low level features that allow the s/w interface to external devices to bewritten
3. It must provide security

EXAMPLES OF LANGUAGES THAT FALL IN THIS DOMAIN:

* Assembly languages were used for a very long time operating systems programming because of its power and efficiency. Other languages for systems programming were later developed for this purpose and these include:
* BCPL is the language in which the original [hello world program](https://en.wikipedia.org/wiki/Hello_world_program) was written.[[5]](https://en.wikipedia.org/wiki/BCPL#cite_note-5) The first [MUD](https://en.wikipedia.org/wiki/MUD) was also written in BCPL. Several [operating systems](https://en.wikipedia.org/wiki/Operating_system) were written partially or wholly in BCPL (for example, [TRIPOS](https://en.wikipedia.org/wiki/TRIPOS) and the earliest versions of [AmigaDOS](https://en.wikipedia.org/wiki/AmigaDOS)).
* **LIL**, the **Little Implementation Language**, was a system programming language during the early days of [UNIX](https://en.wikipedia.org/wiki/Unix) history on [PDP-11](https://en.wikipedia.org/wiki/PDP-11) machines. It was written by [P. J. Plauger](https://en.wikipedia.org/wiki/P._J._Plauger) of [Bell Labs](https://en.wikipedia.org/wiki/Bell_Labs)... LIL attempted to fill the gap between assemblers and machine-independent system implementation languages (such as the [C programming language](https://en.wikipedia.org/wiki/C_programming_language)), by basically adding structured programming to the PDP-11 assembly language.
* C provides constructs that map efficiently to typical [machine instructions](https://en.wikipedia.org/wiki/Machine_code) and has found lasting use in applications previously coded in [assembly language](https://en.wikipedia.org/wiki/Assembly_language). Such applications include [operating systems](https://en.wikipedia.org/wiki/Operating_system) and various [application software](https://en.wikipedia.org/wiki/Application_software) for computers, from [supercomputers](https://en.wikipedia.org/wiki/Supercomputer) to [PLCs](https://en.wikipedia.org/wiki/Programmable_logic_controller) and, to [embedded systems](https://en.wikipedia.org/wiki/Embedded_system).
* **PL/I** (**Programming Language One** is a [procedural](https://en.wikipedia.org/wiki/Procedural_programming), [imperative](https://en.wikipedia.org/wiki/Imperative_programming) computer [programming language](https://en.wikipedia.org/wiki/Programming_language) developed and published by [IBM](https://en.wikipedia.org/wiki/IBM). It is designed for scientific, engineering, business and system programming. The language syntax is English like and suited for describing complex data formats with a wide set of functions available to verify and manipulate them
* **BLISS** is a [system programming language](https://en.wikipedia.org/wiki/System_programming_language) developed at [Carnegie Mellon University](https://en.wikipedia.org/wiki/Carnegie_Mellon_University) by [W. A. Wulf](https://en.wikipedia.org/wiki/W._A._Wulf), [D. B. Russell](https://en.wikipedia.org/w/index.php?title=D._B._Russell&action=edit&redlink=1), and [A. N. Habermann](https://en.wikipedia.org/wiki/A._N._Habermann)around 1970. It was perhaps the best-known systems programming language right up until [C](https://en.wikipedia.org/wiki/C_(programming_language)) made its debut a few years later.
* And extended ALGOL. (Wikipedia, 2020)

Developed Languages: I) IBM for mainframes developed PL/s, a dialect of PL/III) for digital systems BLISS was developed III) for Burroughs extended ALGOLIV) Unix OS was written almost in C But it is too dangerous to use C on large importance

GENERAL PURPOSE DOMAIN

In [computer software](https://en.wikipedia.org/wiki/Computer_software), a **general-purpose programming language** is a [programming language](https://en.wikipedia.org/wiki/Programming_language) designed to be used for writing [software](https://en.wikipedia.org/wiki/Software) in the widest variety of [application domains](https://en.wikipedia.org/wiki/Application_domain)(a [general-purpose language](https://en.wikipedia.org/wiki/General-purpose_language)). A general-purpose programming language has this status because it does not include language constructs designed to be used within a specific application domain.

* [BASIC](https://en.wikipedia.org/wiki/BASIC): **BASIC** (**Beginners' All-purpose Symbolic Instruction Code**[[1]](https://en.wikipedia.org/wiki/BASIC#cite_note-1) or **Beginners All-purpose Symbolic Instruction Code**) is a family of [general-purpose](https://en.wikipedia.org/wiki/General-purpose_programming_language), [high-level programming languages](https://en.wikipedia.org/wiki/High-level_programming_language) whose design philosophy emphasizes ease of use. The original version was designed by [John G. Kemeny](https://en.wikipedia.org/wiki/John_G._Kemeny) and [Thomas E. Kurtz](https://en.wikipedia.org/wiki/Thomas_E._Kurtz) and released at [Dartmouth College](https://en.wikipedia.org/wiki/Dartmouth_College) in 1964. They wanted to enable students in fields other than science and mathematics to use computers
* PHP: Hypertext Preprocessor or PHP is a server-side scripting open source language that was developed in the year 1995 for websites. Today, however, the language has found its use more in general-purpose development today. PHP is used for server-side scripting, command line scripting, and for coding applications. PHP as a language can help you create dynamic websites, web applications, and all types of mobile apps as well.
* PYTHON: A high-level programming language, Python has found its use in web development, app development, creating desktop GUIs, analyzing and computing scientific and numeric data, and for software development.
* PL/1: It is designed for scientific, engineering, business and system programming. It has been used by academic, commercial and industrial organizations since it was introduced in the 1960s. L/I's main domains are [data processing](https://en.wikipedia.org/wiki/Data_processing), [numerical computation](https://en.wikipedia.org/wiki/Numerical_computation), [scientific computing](https://en.wikipedia.org/wiki/Scientific_computing), and [system programming](https://en.wikipedia.org/wiki/System_programming)
* [C](https://en.wikipedia.org/wiki/C_(programming_language)): is a [general-purpose](https://en.wikipedia.org/wiki/General-purpose_language), [procedural](https://en.wikipedia.org/wiki/Procedural_programming) computer [programming language](https://en.wikipedia.org/wiki/Programming_language) .  C provides constructs that map efficiently to typical [machine instructions](https://en.wikipedia.org/wiki/Machine_code) and has found lasting use in applications previously coded in [assembly language](https://en.wikipedia.org/wiki/Assembly_language). Such applications include [operating systems](https://en.wikipedia.org/wiki/Operating_system) and various [application software](https://en.wikipedia.org/wiki/Application_software) for computers, from [supercomputers](https://en.wikipedia.org/wiki/Supercomputer) to [PLCs](https://en.wikipedia.org/wiki/Programmable_logic_controller) and, to [embedded systems](https://en.wikipedia.org/wiki/Embedded_system).
* JAVA: **Java** is a [general-purpose](https://en.wikipedia.org/wiki/General-purpose_language) [programming language](https://en.wikipedia.org/wiki/Programming_language) that is [class-based](https://en.wikipedia.org/wiki/Class-based_programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), and designed to have as few implementation [dependencies](https://en.wikipedia.org/wiki/Dependency_(computer_science)) as possible. It is intended to let [application developers](https://en.wikipedia.org/wiki/Application_developer) *write once, run anywhere* (WORA), meaning that [compiled](https://en.wikipedia.org/wiki/Compiler) Java code can run on all platforms that support Java without the need for recompilation.

(Wikipedia, 2020)

EMBEDDED SYSTEMS DOMAIN:

An embedded system is a computer system—a combination of a computer processor, computer memory, and input/output peripheral devices—that has a dedicated function within a larger mechanical or electrical system (Wikipedia, 2020)

THESE LANGUAGES INVOLVE programming onto embedded systems that might appear to be small circuits, which usually serve definitive purposes:

THESE LANGUAGES INCLUDE:

* **CMS-2** is an [embedded systems](https://en.wikipedia.org/wiki/Embedded_system) [programming language](https://en.wikipedia.org/wiki/Programming_language) used by the [United States Navy](https://en.wikipedia.org/wiki/United_States_Navy). It was an early attempt to develop a standardized high-level computer programming language intended to improve code portability and reusability.
* **JOVIAL** is a [high-level programming language](https://en.wikipedia.org/wiki/High-level_programming_language) similar to [ALGOL](https://en.wikipedia.org/wiki/ALGOL), specialized for developing [embedded systems](https://en.wikipedia.org/wiki/Embedded_system) )specialized computer systems designed to perform one or a few dedicated functions, usually embedded as part of a larger, more complete device, including mechanical parts
* **Open Programming Language** (**OPL**) is an [embedded](https://en.wikipedia.org/wiki/Embedded_system) [programming language](https://en.wikipedia.org/wiki/Programming_language) for portable devices that run the [Symbian Operating System](https://en.wikipedia.org/wiki/Symbian_OS)
* **P4** is a [programming language](https://en.wikipedia.org/wiki/Programming_language) for controlling [packet](https://en.wikipedia.org/wiki/Network_packet) [forwarding planes](https://en.wikipedia.org/wiki/Forwarding_plane) in networking devices, such as routers and switches
* **XC** is a [programming language](https://en.wikipedia.org/wiki/Programming_language) for real-time embedded parallel processors, targeted at the [XMOS](https://en.wikipedia.org/wiki/XMOS) [XCore processor architecture](https://en.wikipedia.org/wiki/XCore_XS1). In combination with XCore processors, XC is used to build embedded systems with levels of I/O, real-time performance and computational ability

(Wikipedia, 2020)

WEB DOMAIN:

This involves languages that create web pages, web sites, web APIs, web animations, web applications and other web based services that are required to be programmed and shared. These languages help in the design of web applications, which can usually be viewed on a browser

* **Curl** is a [reflective](https://en.wikipedia.org/wiki/Reflection_(computer_science)) [object-oriented programming language](https://en.wikipedia.org/wiki/Object-oriented_programming_language) for interactive [web applications](https://en.wikipedia.org/wiki/Web_applications) whose goal is to provide a smoother transition between formatting and programming. Curl combines text markup (as in [HTML](https://en.wikipedia.org/wiki/HTML)), scripting (as in [JavaScript](https://en.wikipedia.org/wiki/JavaScript)), and heavy-duty computing (as in [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), or [C++](https://en.wikipedia.org/wiki/C%2B%2B)) within one unified framework.
* **Elm** is a [domain-specific programming language](https://en.wikipedia.org/wiki/Domain-specific_language) for [declaratively](https://en.wikipedia.org/wiki/Declarative_programming) creating [web browser](https://en.wikipedia.org/wiki/Web_browser)-based [graphical user interfaces](https://en.wikipedia.org/wiki/Graphical_user_interface). Elm is [purely functional](https://en.wikipedia.org/wiki/Purely_functional_programming), and is developed with emphasis on [usability](https://en.wikipedia.org/wiki/Usability), performance, and [robustness](https://en.wikipedia.org/wiki/Robustness_(computer_science)).
* **Opa** is an [open-source](https://en.wikipedia.org/wiki/Open-source_software) [programming language](https://en.wikipedia.org/wiki/Programming_language) for developing [scalable](https://en.wikipedia.org/wiki/Scalability) [web applications](https://en.wikipedia.org/wiki/Web_applications). It can be used for both [client-side](https://en.wikipedia.org/wiki/Client-side_scripting) and [server-side scripting](https://en.wikipedia.org/wiki/Server-side_scripting), where complete programs are written in Opa and subsequently compiled to [Node.js](https://en.wikipedia.org/wiki/Node.js) on the [server](https://en.wikipedia.org/wiki/Server_(computing)) and [JavaScript](https://en.wikipedia.org/wiki/JavaScript) on the [client](https://en.wikipedia.org/wiki/Client_(computing)), with the [compiler](https://en.wikipedia.org/wiki/Compiler) automating all communication between the two.
* **PHP** is a popular general-purpose scripting language that is especially suited to web development. PHP code is usually processed on a web server by a PHP [interpreter](https://en.wikipedia.org/wiki/Interpreter_(computing)) implemented as a [module](https://en.wikipedia.org/wiki/Plugin_(computing)), a [daemon](https://en.wikipedia.org/wiki/Daemon_(computing)) or as a [Common Gateway Interface](https://en.wikipedia.org/wiki/Common_Gateway_Interface) (CGI) executable.

(Wikipedia, 2020)

MOBILE DOMAIN:

This involve languages, which can create applications that run on a mobile platform and meet the requirements of that platform. It usually involves languages that can program software that work on phones, tabs, etc.

* JAVA: One of the most popular and preferred languages among android app developers, One of the most popular and preferred languages among android app developers,
* PYTHON: A high-level programming language, Python has found its use app development.  Python as a language is beautiful, explicit, simple, sometimes complex but not complicated, and thoroughly readable
* C#: C Sharp is a multi-paradigm programming language that supports imperative, generic and object-oriented programming. (MEDIUM, 2017). It can be used to make mobile applications
* SWIFT: Considered as the programming language of the future, Swift is the latest programming language to make their way into the Apple ecosystem and it is majorly because of its popularity in writing the code for Apple’s latest APIs
* KOLTIN: This one is comparatively a new object-oriented first-class programming language and is similar to Java when it comes to the structure of the language. Kotlin is unique in the fact that it supports almost all the IDEs including Android Studio and SDK toolkit.
* PHP: Hypertext Preprocessor or PHP is a server-side scripting open source language that was developed in the year 1995 for websites. Today, however, PHP Is a language can help you create dynamic websites, web applications, and all types of mobile apps as well.

(snigdha, 2020)

Compiled and Interpreted Languages

INTERPRETED LANGUAGES

An interpreted language is a programming language for which most of its implementations execute instructions directly, without previously compiling a program into machine-language instructions. The interpreter executes the program directly, translating each statement into a sequence of one or more subroutines already compiled into machine code. (Wikipedia)

Examples are:

AutoIt: It is a freeware automation language for Microsoft Windows. Its main intent is to create automation scripts that can be used for the execution of certain repetitive tasks on Windows.

BASIC: Developed by John George Kemeny and Thomas Eugene Kurtz at Dartmouth in 1964, it is an acronym for Beginner’s All-purpose Symbolic Instruction Code. It was designed with the intent of giving the non-science people an access to computers.

Eiffel: An object-oriented programming language is ISO-standardized and used to develop extensible and reusable software. It is a development platform for many industries such as finance, aerospace and video gaming.

Forth: It is a structured imperative programming language, which bases its implementation on stacks. It supports an interactive execution of commands as well as the compilation of sequences of commands.

Game Maker Language: It is an interpreted computer programming language intended to be used in cooperation with Game Maker, an application for game creation. Mark Overmars, a Dutch computer scientist, designed this language.

ICI: Designed by Tim Long in 1992, ICI is a general-purpose interpreted computer programming language. It supports dynamic typing, flexible data types and other language constructs similar to C.

J: Ken Iverson and Roger Hui developed a programming language that requires only the basic ASCII character set. An array programming language works well with mathematical and statistical operations.

Lisp: Lisp is the second-oldest high-level programming language in widespread use today. The name Lisp is derived from ‘List Processing Language’. One of the important data structures that Lisp supports is linked list. Lisp programs deal with source code as a data structure.

(NICE, 2017)

COMPILED LANGUAGES

A compiled language is a programming language whose implementations are typically compilers (translators that generate machine code from source code), and not interpreters (step-by-step executors of source code, where no pre-runtime translation takes place). (Wikipedia)

Ada: It is a statically typed, structured, imperative programming language that is based on Pascal. A team of CII Honeywell Bull that was led by Jean Ichbiah developed Ada. The Ada compilers are validated for mission-critical systems. Ada is an internationally standardized computer programming language.

ALGOL: Algorithmic Language, as it is called, is actually a family of imperative programming languages that was developed in the middle 1950s. It proved instrumental in the creation of programming languages like BCPL, B and C. Ole-Johan Dahl and Kristen Nygaard of the Norwegian Computing Center in Oslo were the brains behind Simula.

C: Dennis Ritchie at the Bell Telephone Laboratories developed C to be used on the UNIX platform. It is a general-purpose, cross-platform, procedural, imperative programming language. It is used for implementing system software and application software and is one of the most-used computer programming languages of today. C. influenced the development of C++ and C #.

C++: It consists of a combination of high-level and low-level language features and is hence considered as a middle-level programming language. Bjarne Stroustrup of Bell Labs developed C++ as an extension of the C language. Originally known as ‘C with Classes’, it came to be known as C++ from 1983. A multi-paradigm language supports procedural programming, generic programming, object-oriented programming, and data abstraction.

C#: C Sharp is a multi-paradigm programming language that supports imperative, generic and object-oriented programming. It is a part of the Microsoft .NET Framework. It is similar to C++ in its object-oriented syntax and is influenced by Java and Delphi.

CLEO: It is known as the Clear Language for Expressing Orders and is a computer language for the LEO computer.

COBOL: The name stands for Common Business-Oriented Language that is designed for the business and finance domain. COBOL 2002 standard supports object-oriented programming. It is one of the very old programming languages that are still in use.

Cobra: It is an object-oriented programming language that runs on .NET and Mono frameworks. Chuck Esterbrook developed it. Languages like Python and C #influence its design. It supports static and dynamic typing and is suited for unit tests. Today, it is an open source project.

D: Originally designed as an enhancement of C++, it is also influenced by Java, Eiffel, and C#. It is an object-oriented, imperative, multi-paradigm system programming language developed by Walter Bright of Digital Mars.

DASL: Acronym of Distributed Application Specification Language, it is a high-level, strongly typed programming language that was developed at the Sun Microsystems. It was created with an intent to be used for developing web applications.

DIBOL: Acronym of Digital Interactive Business Oriented Language, DIBOL is a general-purpose procedural imperative programming language. It is similar to COBOL as its best suited for the development of Management Information Systems.

FORTRAN: It is a procedural, imperative, general-purpose computer programming language that works well for scientific computations and numeric operations. After IBM developed it in the 1950s, it soon gained popularity in programming. It is very popular in the field of high-performance computing. It is a structured and compiled programming language that is a subset of Fortran95. Fortran 2003, a revised version of FORTRAN supports object-oriented programming.

Java: It is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. Compiled Java code can run on all platforms that support Java without the need for recompilation. It is a very popular language of the modern times.

JOVIAL: It is a high-order computer programming language similar to ALGOL. It is best suited to the design and development of embedded systems.

(NICE, 2017)

CHRONOLOGICAL EVOLUTION OF PROGRAMMING LANGUAGES

Computers were originally programmed in machine languages, which consisted of ones and zeros, but due to the numerous disadvantages of machine languages, other languages were created to give a higher level of abstraction and to combat the disadvantages brought on by machine language.

An example is Short Code, which was proposed by [John Mauchly](https://en.wikipedia.org/wiki/John_Mauchly) in 1949 and originally known as Brief Code. William Schmitt implemented a version of Brief Code in 1949 for the [BINAC](https://en.wikipedia.org/wiki/BINAC) computer, though it was never debugged and tested. The following year Schmitt implemented a new version of Brief Code for the [UNIVAC I](https://en.wikipedia.org/wiki/UNIVAC_I) where it was now known as Short Code (also Short Order Code).

The A-0 system (Arithmetic Language version 0), written by [Grace Murray Hopper](https://en.wikipedia.org/wiki/Grace_Hopper) in 1951 and 1952 for the [UNIVAC I](https://en.wikipedia.org/wiki/UNIVAC_I), was an early [compiler](https://en.wikipedia.org/wiki/Compiler) related tool developed for electronic computers. The A-0 functioned more as a [loader](https://en.wikipedia.org/wiki/Loader_(computing)) or [linker](https://en.wikipedia.org/wiki/Linker_(computing)) than the modern notion of a compiler. A program was specified as a sequence of subroutines and arguments. The first autocode and its compiler were developed by [Alick Glennie](https://en.wikipedia.org/wiki/Alick_Glennie) in 1952 for the Mark 1 computer at the University of Manchester and is considered by some to be the first [compiled](https://en.wikipedia.org/wiki/Compiler) programming language.

Speedcoding or Speedcode was the first [high-level programming language](https://en.wikipedia.org/wiki/High-level_programming_language) created for an [IBM](https://en.wikipedia.org/wiki/IBM) computer. The language was developed by [John Backus](https://en.wikipedia.org/wiki/John_Backus) in 1953 for the [IBM 701](https://en.wikipedia.org/wiki/IBM_701) to support computation with numbers.

The concept for FORTRAN was created back in 1954-1955 but it was not implemented until later in the 1957 implementation of FORTRAN 1. COMTRAN (COMmercial TRANslator) is an early [programming language](https://en.wikipedia.org/wiki/Programming_language) developed at [IBM](https://en.wikipedia.org/wiki/IBM). It was intended as the business-programming equivalent of the scientific programming language [FORTRAN](https://en.wikipedia.org/wiki/FORTRAN) in 1957.

Structured programming started in the late 1950s with the emergence of ALGOL 58 in 1958, but other unstructured programming languages were still being created and widely used even COBOL emerged in 1960 and JOSS in 1963; and then the criticism of unstructured programming languages began; with the help of [Edsger W. Dijkstra](https://en.wikipedia.org/wiki/Edsger_W._Dijkstra) 1968 publication of "[Go To Statement Considered Harmful](https://en.wikipedia.org/wiki/Go_To_Statement_Considered_Harmful)".

The concept of structured programming became the new and desired thing leading ALGOL to inspire many other languages including SIMULA 67 in 1968 after the concept had been made in 1962.

Simula brought on the concept of many object-oriented features and is regarded as one of the first, if not the first, object-oriented programming language.

In 1975, Modula was created and recognized as the first modular programming language and it was believed to have derived its concept by improving on the already laid down paradigms of structured and procedural programming languages since Pascal, one of the most influential structured programming languages, which was created in 1970, inspired it.

In 1976’s MESA was also regarded as one of the earliest modular programming languages, which was also based on ALGOL- a structured programming language.

The features of object oriented programming languages eventually caught on and gained favor in the programming community and this can especially be seen around the late 70’s and late 80’s with the emergence of programming languages like C with classes and C++ taking after Simula

It can also be noticed that the specification in domain has reduced overtime; from the 50s programming languages were usually created to achieve a specified purpose in a specified domain. Programming languages like COMTRAN, FOTRAN, COBOL, etc. eventually led to the inspiration of more general purposed programming languages in the 70s downward like PASCAL, which came in 1970, and C, which came in 1972.

When Java came in the 1995 with its WORA principle, it shed light on the need for cross-platform programming language development as well

Ever since then, the programming languages that have been created have been more dynamical purpose, more structured and object oriented and they usually fall under many different paradigms, they make use of a number of several different subroutines and specified way to guide the programs control but also have different ways to ensure reusability and easy community building.

Old programming languages which could not be upgraded, or could not be revised in order to fit the newer paradigms and to perform the latest functionalities for the applications in that programming languages’ specified application were termed obsolete

While the others which could have managed to survive, like: C, Pascal: which has taken up several different forms after its creation, and also like BASIC which had inspired several different languages like Visual Basic which still remain in use today.

|  |  |  |  |
| --- | --- | --- | --- |
| YEAR | NAME | CREATORS | PREDECESSOR |
| 1949 | SHORT CODE | [John Mauchly](https://en.wikipedia.org/wiki/John_Mauchly) and William F. Schmitt | ENIAC Short Code |
| 1950 | [Birkbeck Assembler](https://en.wikipedia.org/w/index.php?title=Birkbeck_Assembler&action=edit&redlink=1) | [Kathleen Booth](https://en.wikipedia.org/wiki/Kathleen_Booth) | ARC |
| 1951 | [Regional Assembly Language](https://en.wikipedia.org/w/index.php?title=Regional_Assembly_Language&action=edit&redlink=1) | [Maurice Wilkes](https://en.wikipedia.org/wiki/Maurice_Wilkes) | EDSAC |
| 1951 | [Whirlwind assembler](https://en.wikipedia.org/w/index.php?title=Whirlwind_assembler&action=edit&redlink=1) | Charles Adams and Jack Gilmore at [MIT](https://en.wikipedia.org/wiki/Massachusetts_Institute_of_Technology) [Project Whirlwind](https://en.wikipedia.org/wiki/Whirlwind_(computer)) | EDSAC |
| 1951 | [Rochester assembler](https://en.wikipedia.org/w/index.php?title=Rochester_assembler&action=edit&redlink=1) | [Nat Rochester](https://en.wikipedia.org/wiki/Nathaniel_Rochester_(computer_scientist)) | EDSAC |
| 1952 | [A-0](https://en.wikipedia.org/wiki/A-0_System) | GRACE HOPPER | SHORT CODE |
| 1953 | [Speedcoding](https://en.wikipedia.org/wiki/Speedcoding) | [John W. Backus](https://en.wikipedia.org/wiki/John_Backus) | - |
| 1955 | [FLOW-MATIC](https://en.wikipedia.org/wiki/FLOW-MATIC) | Team led by [Grace Hopper](https://en.wikipedia.org/wiki/Grace_Hopper) at UNIVAC | A-0 |
| 1957 | [COMTRAN](https://en.wikipedia.org/wiki/COMTRAN) | [Bob Bemer](https://en.wikipedia.org/wiki/Bob_Bemer) | FLOW-MATIC |
| 1958 | [FORTRAN I](https://en.wikipedia.org/wiki/FORTRAN_I) | [John W. Backus](https://en.wikipedia.org/wiki/John_Backus) at [IBM](https://en.wikipedia.org/wiki/IBM) | FORTRAN concept as at 1954-1955 |
| 1958 | [IPL II](https://en.wikipedia.org/wiki/Information_Processing_Language) (implementation) | [Allen Newell](https://en.wikipedia.org/wiki/Allen_Newell), [Cliff Shaw](https://en.wikipedia.org/wiki/Cliff_Shaw), [Herbert A. Simon](https://en.wikipedia.org/wiki/Herbert_A._Simon) | IPL I (concept) |
| 1958 | [ALGOL 58](https://en.wikipedia.org/wiki/ALGOL_58) (IAL) | ACM/GAMM | FORTRAN, IT, Sequentielle Formelübersetzung |
| 1959 | JOVIAL | [Jules Schwartz](https://en.wikipedia.org/wiki/Jules_Schwartz) at [SDC](https://en.wikipedia.org/wiki/System_Development_Corporation) | [ALGOL 58](https://en.wikipedia.org/wiki/ALGOL_58) |
| 1959 | LISP(IMPLEMENTATION) | [John McCarthy](https://en.wikipedia.org/wiki/John_McCarthy_(computer_scientist)) | LISP CONCEPT (AT 1956–58), IPL |
| 1960 | COBOL 61 | The [CODASYL](https://en.wikipedia.org/wiki/CODASYL) Committee | COBOL CONCEPT(AT 1959), FLOW-MATIC, COMTRAN |
| 1963 | CPL | Barron, [Christopher Strachey](https://en.wikipedia.org/wiki/Christopher_Strachey), *et al.* | ALGOL 60 (CAME 1960) |
| 1963 | JOSS1 | Cliff Shaw, [RAND](https://en.wikipedia.org/wiki/RAND_Corporation) | ALGOL 58 |
| 1964 | BASIC | [John George Kemeny](https://en.wikipedia.org/wiki/John_George_Kemeny) and [Thomas Eugene Kurtz](https://en.wikipedia.org/wiki/Thomas_Eugene_Kurtz) at [Dartmouth College](https://en.wikipedia.org/wiki/Dartmouth_College) | FOTRAN II, JOSS |
| 1964 | I[BM RPG](https://en.wikipedia.org/wiki/IBM_RPG) | IBM | FARGO |
| 1965 | TELECOMP | BBM | JOSS |
| 1966 | [APL](https://en.wikipedia.org/wiki/APL_(programming_language)) | [Kenneth E. Iverson](https://en.wikipedia.org/wiki/Kenneth_E._Iverson) | APL CONCEPT (1962) |
| 1967 | BCPL | [Martin Richards](https://en.wikipedia.org/wiki/Martin_Richards_(computer_scientist)) | CPL |
| 1968 | [Simula 67](https://en.wikipedia.org/wiki/Simula) | [Ole-Johan Dahl](https://en.wikipedia.org/wiki/Ole-Johan_Dahl), Bjørn Myhrhaug, [Kristen Nygaard](https://en.wikipedia.org/wiki/Kristen_Nygaard) at [Norsk Regnesentral](https://en.wikipedia.org/wiki/Norwegian_Computing_Center) | SIMULI CONCEPT(1962), ALGOL 60 |
| 1968 | [DIBOL-8](https://en.wikipedia.org/wiki/DIBOL) | DEC (Digital Equipment Cooperation) | - |
| 1969 | [PL/I](https://en.wikipedia.org/wiki/PL/I) | IBM | ALGOL 60, COBOL, FORTRAN |
| 1969 | B | [Ken Thompson](https://en.wikipedia.org/wiki/Ken_Thompson), with contributions from [Dennis Ritchie](https://en.wikipedia.org/wiki/Dennis_Ritchie) | BCPL |
| 1970 | FORTH | [Charles H. Moore](https://en.wikipedia.org/wiki/Charles_H._Moore) | - |
| 1970 | PASCAL | [Niklaus Wirth](https://en.wikipedia.org/wiki/Niklaus_Wirth), Kathleen Jensen | ALGOL 60, ALGOL W |
| 1970 | BLISS | Wulf, Russell, Habermann at [Carnegie Mellon University](https://en.wikipedia.org/wiki/Carnegie_Mellon_University) | ALGOL |
| 1972 | C | [Dennis Ritchie](https://en.wikipedia.org/wiki/Dennis_Ritchie) | B, BCPL, ALGOL 68 |
| 1972 | PROLOG | [Alain Colmerauer](https://en.wikipedia.org/wiki/Alain_Colmerauer) | 2-level W-Grammar |
| 1973 | ML | [Robin Milner](https://en.wikipedia.org/wiki/Robin_Milner) |  |
| 1975 | MODULA | [Niklaus Wirth](https://en.wikipedia.org/wiki/Niklaus_Wirth) | PASCAL |
| 1976 | MESA | [Xerox PARC](https://en.wikipedia.org/wiki/PARC_(company)) | ALGOL |
| 1977 | IDL | David Stern of Research Systems Inc | FOTRAN |
| 1980 | SMALLTALK | Nevil Brownlee at the [University of Auckland](https://en.wikipedia.org/wiki/University_of_Auckland) | ALGOL 60 |
| 1980 | ADA | [Jean Ichbiah](https://en.wikipedia.org/wiki/Jean_Ichbiah) at [CII Honeywell Bull](https://en.wikipedia.org/wiki/Groupe_Bull) | GREEN |
| 1980 | C WITH CLASSES | [Bjarne Stroustrup](https://en.wikipedia.org/wiki/Bjarne_Stroustrup)[[9]](https://en.wikipedia.org/wiki/Timeline_of_programming_languages#cite_note-9) | C, Simula 67 |
| 1983 | C++ | [Bjarne Stroustrup](https://en.wikipedia.org/wiki/Bjarne_Stroustrup) | C WITH CLASSES |
| 1983 | TURBO PASCAL | H[ejlsberg](https://en.wikipedia.org/wiki/Anders_Hejlsberg) at [Borland](https://en.wikipedia.org/wiki/Borland) | PASCAL |
| 1984 | OPL | [Psion](https://en.wikipedia.org/wiki/Psion_(company)) | BASIC |
| 1985 | QUICKBASIC | MICROSOFT | BASIC |
| 1986 | OBJECT PASCAL | [Apple Computer Inc.](https://en.wikipedia.org/wiki/Apple_Inc.) | PASCAL |
| 1986 | EIFFEL | [Bertrand Meyer](https://en.wikipedia.org/wiki/Bertrand_Meyer) | Simula 67, Ada |
| 1987 | OBERON | [Niklaus Wirth](https://en.wikipedia.org/wiki/Niklaus_Wirth) | MODULA II |
| 1987 | HYPERTALK | [Apple Computer Inc.](https://en.wikipedia.org/wiki/Apple_Inc.) |  |
| 1989 | PYTHON | [Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) | [ABC](https://en.wikipedia.org/wiki/ABC_(programming_language)), [SETL](https://en.wikipedia.org/wiki/SETL) |
| 1990 | J | K[enneth E. Iverson](https://en.wikipedia.org/wiki/Kenneth_E._Iverson), [Roger Hui](https://en.wikipedia.org/wiki/Roger_Hui) at Iverson Software | APL, FP |
| 1991 | VISUAL BASIC | [Alan Cooper](https://en.wikipedia.org/wiki/Alan_Cooper), sold to [Microsoft](https://en.wikipedia.org/wiki/Microsoft) | QUICK BASIC |
| 1993 | APPLESCRIPT | [Apple Computer Inc.](https://en.wikipedia.org/wiki/Apple_Inc.) | HYPERTALK |
| 1995 | JAVA | [James Gosling](https://en.wikipedia.org/wiki/James_Gosling) at [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems) | C, Simula 67, C++, Smalltalk, Ada 83, Objective-C, Mesa |
| 1995 | PHP | [Rasmus Lerdorf](https://en.wikipedia.org/wiki/Rasmus_Lerdorf) | PERL |
| 1995 | RUBY | [Yukihiro Matsumoto](https://en.wikipedia.org/wiki/Yukihiro_Matsumoto) | SMALLTALK, PERL |
| 1996 | CURL | DAVId Kranz, Steve Ward, Chris Terman at [MIT](https://en.wikipedia.org/wiki/Massachusetts_Institute_of_Technology) | Lisp, C++, Tcl/Tk, TeX, HTML |
| 1996 | OCaml | INRIA | Caml Light, Standard ML |
| 1997 | E | [Mark S. Miller](https://en.wikipedia.org/wiki/Mark_S._Miller) | Joule, Original-E |
| 1999 | HARBOUR | Antonio Linares | DBASE |
| 1999 | AUTOIT | AutoIt Consulting Ltd. | BASIC |
| 2001 | ASPECT J | [Gregor Kiczales](https://en.wikipedia.org/wiki/Gregor_Kiczales), [Xerox PARC](https://en.wikipedia.org/wiki/PARC_(company)) | Java, Common Lisp |
| 2001 | D | [Walter Bright](https://en.wikipedia.org/wiki/Walter_Bright), Digital Mars | C, C++, C#, Java |
| 2003 | FACTOR | [Slava Pestov](https://en.wikipedia.org/wiki/Slava_Pestov) | JOY, FORTH, LISP |
| 2005 | XC | INTENDED FOR XMOS | [C](https://en.wikipedia.org/wiki/C_(programming_language)), [occam](https://en.wikipedia.org/wiki/Occam_(programming_language)), [CSP](https://en.wikipedia.org/wiki/Communicating_Sequential_Processes) |
| 2006 | COBRA | ChuckEsterbrook | Python, C#, Eiffel, Objective-C |
| 2007 | ADA 2005 | Ada Rapporteur Group | ADA 95 |
| 2009 | GO | GOOGLE | [C](https://en.wikipedia.org/wiki/C_(programming_language)), [Oberon](https://en.wikipedia.org/wiki/Oberon_(programming_language)), [Limbo](https://en.wikipedia.org/wiki/Limbo_(programming_language)), [Smalltalk](https://en.wikipedia.org/wiki/Smalltalk) |
| 2011 | KOTLIN | JETBRAINS | [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [Eiffel](https://en.wikipedia.org/wiki/Eiffel_(programming_language)), [Gosu](https://en.wikipedia.org/wiki/Gosu_(programming_language)), [Groovy](https://en.wikipedia.org/wiki/Groovy_(programming_language)), [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [ML](https://en.wikipedia.org/wiki/ML_(programming_language)), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [Scala](https://en.wikipedia.org/wiki/Scala_(programming_language)), [Swift](https://en.wikipedia.org/wiki/Swift_(programming_language)) |
| 2012 | ELM | Evan Czaplicki | [Haskell](https://en.wikipedia.org/wiki/Haskell_(programming_language)), [Standard ML](https://en.wikipedia.org/wiki/Standard_ML), [OCaml](https://en.wikipedia.org/wiki/OCaml), [F#](https://en.wikipedia.org/wiki/F_Sharp_(programming_language)) |
| 2013 | P4 | P4 Language Consortium | - |
| 2014 | SWIFT | [Apple Inc.](https://en.wikipedia.org/wiki/Apple_Inc.) | [Objective-C](https://en.wikipedia.org/wiki/Objective-C), [Rust](https://en.wikipedia.org/wiki/Rust_(programming_language)), [Haskell](https://en.wikipedia.org/wiki/Haskell_(programming_language)), [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [CLU](https://en.wikipedia.org/wiki/CLU_(programming_language)) |
| 2015 | RAKU | [The Rakudo Team](https://github.com/rakudo/rakudo/blob/master/CREDITS) | [Perl](https://en.wikipedia.org/wiki/Perl), [Haskell](https://en.wikipedia.org/wiki/Haskell_(programming_language)), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)) |
| 2016 | [Reason](https://en.wikipedia.org/wiki/Reason_(programming_language)) | Jordan Walke | [JavaScript](https://en.wikipedia.org/wiki/JavaScript), [OCaml](https://en.wikipedia.org/wiki/OCaml)[[13]](https://en.wikipedia.org/wiki/Timeline_of_programming_languages#cite_note-13) |
| 2017 | C++17 | C++ ISO/IEC 14882:2017 | [C++](https://en.wikipedia.org/wiki/C%2B%2B), Standard C, [C](https://en.wikipedia.org/wiki/C_(programming_language)) |
| 2018 | FOTRAN 2018 | ISO/IEC JTC1/SC22/WG5 N2150:2018 | [Fortran 2008](https://en.wikipedia.org/wiki/Fortran#Fortran_2008) |
| 2019 | BOSQUE | Mark Marron, [Microsoft](https://en.wikipedia.org/wiki/Microsoft) | [JavaScript](https://en.wikipedia.org/wiki/JavaScript), [TypeScript](https://en.wikipedia.org/wiki/TypeScript), [ML](https://en.wikipedia.org/wiki/ML_(programming_language)) |

WHAT IS THE DIFFERENCE BETWEEN MODULAR PROGRAMMIGN AND OBJECT ORIENTED PROGRAMMING?

The modular programming paradigm is one that has been closely related with the object-oriented paradigm mostly because these two aim at giving an ordered sequence and helping to control the program flow in order to avoid unstructured programming and although, they can be linked together on that basis, they are not in fact similar and this is why:

The concept of modular programming as understood by me is to divide the program into different modules that have independent functionalities, which are responsible for one section of the program.

For example, we imagine a program that is in charge of doing a simple action like preparing rice and this RICE Boiler, has to measure the required quantity of rice, wash the rice, add salt and oil, boil the rice and stop the process when it is ready.

Under the concept of modular programming, we would break the entire program flow into several different modules based on the actions that are to be performed. I would break the program into about five different modules that would get the job done

First module is in charge of measuring the amount of rice that is to be cooked and so on. In this way, we think of the functionalities and we break the program into independent modules that handle these tasks and can have several different subroutines within them to ensure the smooth running of that module

The concept of Object oriented language according to my understanding; is based on objects and classes. In the conceptual model of the program, it works closely with entities and their attributes then translates that into a class, which may have dynamic and static attributes. The dynamic attributes usually embody the functionalities that tat particular entity posses

We use the same example were we imagine a program that is in charge of doing a simple action like preparing rice and this RICE Boiler, has to measure the required quantity of rice, wash the rice, add salt and oil, boil the rice and stop the process when it is ready.

I would create a rice preparer class, which would have several different fields such as quantity of rice, amount of salt, amount of oil, amount of water, etc.

Then it would have methods for the functionalities, which involve; a method for the measuring of rice: where the user may input the weight in KG and the value of rice measured would be updated. The salt and oil adder would be to increase the amount of salt and oil in oz. based on some percentage derived from the quantity of rice that has been inputted, etc. etc.

In this way, we think of the program in terms of entities, of many different components coming together and although not everything or every aspect of an object-oriented program is an object, it still plays a role in the logic of the paradigm

REFERENCES

1. OGIRI, R (2018). THE DIFFERENCES BETWEEN STRUCTURED LANGUAGE AND UNSTRUCTURED LANGUAGE: PROGRAMING LANGUAGE. Afribary.com: Retrieved April 20, 2020, from <https://afribary.com/works/the-differences-between-structured-language-and-unstructured-language-programing-language-4783>
2. Wikipedia (2019, October 6). NON-STRUCTURED PROGRAMMING. Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/Non-structured_programming>
3. Wikipedia (2020, March 21). JOSS. Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/JOSS#Language>
4. Wikipedia (2020, April 16). FOCAL (Programming Language). Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/FOCAL_(programming_language)>
5. Wikipedia (2019, September 30). TELECOMP. Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/TELCOMP>
6. Wikipedia (2020, April 20). Assembly Language. Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/Assembly_language>
7. Maxwell B. A. (2017). Assembly Language Design. cs.colby.edu. Retrieved April 20, 2020 from <https://cs.colby.edu/courses/F19/cs232-labs/labs/lab08/assembly.php>
8. Wikipedia (2020, April 20). BASIC. Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/BASIC>
9. Wikipedia (2020, March 8). Fortran . Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/Fortran>
10. Wikipedia (2020, April 19). COBOL . Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/COBOL>
11. Wikipedia (2019, December 15). Structured programming . Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/Structured_programming>
12. Wikipedia (2020, April 17). ALGOL . Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/ALGOL>
13. Wikipedia (2020, April 20). PASCAL (programming language). Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/Pascal_(programming_language)>
14. Wikipedia (2020, April 4). JOVIAL (programming language). Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/JOVIAL>
15. Wikipedia (2020, April 12). PL/1. Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/PL/I>
16. Wikipedia (2020, April 18). CPL (programming language). Wikipedia.com. Retrieved April 20, 2020 from <https://en.wikipedia.org/wiki/CPL_(programming_language)>
17. Wikipedia (2020, April 6). Modular Programming. Wikipedia.com. Retrieved April 21, 2020 from <https://en.wikipedia.org/wiki/Modular_programming>
18. Wikipedia (2020, April 5). F (Programming Language). Wikipedia.com. Retrieved April 21, 2020 from <https://en.wikipedia.org/wiki/F_(programming_language)>
19. Wikipedia (2020, March 25). Modula. Wikipedia.com. Retrieved April 21, 2020 from <https://en.wikipedia.org/wiki/Modula>
20. Wikipedia (2020, April 9). Oberon (Programming Language). Wikipedia.com. Retrieved April 21, 2020 from <https://en.wikipedia.org/wiki/Oberon_(programming_language)>
21. Techtarget (2020, April). Object Oriented Programming (OOPs). techtarget.com. Retrieved April 21, 2020 from <https://searchapparchitecture.techtarget.com/definition/object-oriented-programming-OOP>
22. Wikipedia (2020, April 21). C++. Wikipedia.com. Retrieved April 21, 2020 from <https://en.wikipedia.org/wiki/C%2B%2B>
23. Wikipedia (2020, April 12). Ruby (Programming Language). Wikipedia.com. Retrieved April 22, 2020 from <https://en.wikipedia.org/wiki/Ruby_(programming_language)>
24. Wikipedia (2020, March 13).Smalltalk. Wikipedia.com. Retrieved April 22, 2020 from <https://en.wikipedia.org/wiki/Smalltalk>
25. Wikipedia (2020, April 17). Simula. Wikipedia.com. Retrieved April 22, 2020 from <https://en.wikipedia.org/wiki/Simula>
26. Wikipedia (2020, April 19). BETA (Programming language). Wikipedia.com. Retrieved April 22, 2020 from <https://en.wikipedia.org/wiki/BETA_(programming_language)>
27. Nice B. (2017, March 18). A Complete List Of Computer Programming Languages. medium.com. Retrieved April 22, 2020 from <https://medium.com/web-development-zone/a-complete-list-of-computer-programming-languages-1d8bc5a891f>
28. Wikipedia (2020, April 22). Java (Programming language). Wikipedia.com. Retrieved April 22, 2020 from <https://en.wikipedia.org/wiki/Java_(programming_language)>
29. Wikipedia (2020, April 20). Aspect Oriented Programming. Wikipedia.com. Retrieved April 23, 2020 from <https://en.wikipedia.org/wiki/Aspect-oriented_programming>
30. Wikipedia (2020, April 7. AspectJ. Wikipedia.com. Retrieved April 23, 2020 from <https://en.wikipedia.org/wiki/AspectJ>
31. Wikipedia (2019, November 25. AspectC++. Wikipedia.com. Retrieved April 23, 2020 from <https://en.wikipedia.org/wiki/AspectC%2B%2B>
32. Wikipedia (2019, August 1). E (verification language). Wikipedia.com. Retrieved April 23, 2020 from <https://en.wikipedia.org/wiki/E_(verification_language)>
33. [**soundarpandiyan**](https://www.scribd.com/user/18539048/soundarpandiyan). Programming Paradigms. Scribd.com. Retrieved April 24, 2020 from <https://www.scribd.com/doc/27463842/Programming-Domains>
34. Wikipedia (2019, March 12). List of Programming Language for artificial intelligence Wikipedia.com. Retrieved April 24, 2020 from <https://en.wikipedia.org/wiki/List_of_programming_languages_for_artificial_intelligence>
35. CSC 272 - Software II: Principles of Programming Languages . Retrieved April 24 2020 from <https://home.adelphi.edu/~siegfried/cs272/272l1.pdf>
36. Wikipedia (2019, July 31). CMS-2 (programming language). Wikipedia.com. Retrieved April 24, 2020 from <https://en.wikipedia.org/wiki/CMS-2_(programming_language)>
37. Wikipedia (2018, August 26).COMTRAN. Wikipedia.com. Retrieved April 24, 2020 from <https://en.wikipedia.org/wiki/COMTRAN>
38. Wikipedia (2020, April 1).Curl (programming language). Wikipedia.com. Retrieved April 24, 2020 from <https://en.wikipedia.org/wiki/Curl_(programming_language)>
39. Wikipedia (2020, March 31).Elm (programming language). Wikipedia.com. Retrieved April 24, 2020 from <https://en.wikipedia.org/wiki/Elm_(programming_language)>
40. Wikipedia (2020, April 6).Habour (programming language). Wikipedia.com. Retrieved April 24, 2020 from <https://en.wikipedia.org/wiki/Harbour_(programming_language)>
41. Wikipedia (2020, April 9).IBM RPG. Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/IBM_RPG>
42. Wikipedia (2020, March 12).IDL (programming language). Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/IDL_(programming_language)>
43. Wikipedia (2020, April 19).IBM Klerer-May System. Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/Klerer%E2%80%93May_System>
44. Wikipedia (2019, July 11). Little Implementation Language. Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/Little_Implementation_Language>
45. Wikipedia (2020, March 9).LINC 4GL. Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/LINC_4GL>
46. Wikipedia (2020, April 23). OpenEdge Advanced Business Language. Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/OpenEdge_Advanced_Business_Language>
47. Wikipedia (2019, September 29). Opa (programming language). Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/Opa_(programming_language)>
48. Wikipedia (2018, November 19). Open Programming Language. Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/Open_Programming_Language>
49. Wikipedia (2020, April 10). P4 (programming language). Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/P4_(programming_language)>
50. Wikipedia (2020, April 23). PHP. Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/PHP>
51. Wikipedia (2020, March 15). XC (programming language). Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/XC_(programming_language)>
52. [**Snigdha**](https://www.appypie.com/author/snigdha) (2020, April 8). 25 best programming Languages for Mobile App Development Tools ^ Framework. applepieb.com. Retrieved April 25, 2020 from <https://www.appypie.com/top-programming-languages-for-mobile-app-development>
53. Wikipedia (2020, April 22). C (programming language). Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/C_(programming_language)>
54. Wikipedia (2020, April 23). BCPL. Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/BCPL>
55. Wikipedia (2019, August 27). BLISS. Wikipedia.com. Retrieved April 25, 2020 from <https://en.wikipedia.org/wiki/BLISS>
56. Wikipedia (2020, April12). Timeline of Programming Languages. Wikipedia.com. Retrieved April 26, 2020 from <https://en.wikipedia.org/wiki/Timeline_of_programming_languages>
57. Richard. Modular Programming Versus Object Oriented Programming (The Good, The Bad and the Ugly). Retrieved April 26, 2020 from <http://petesqbsite.com/sections/express/issue17/modularversusoop.html>