

**Faculty of engineering - Shoubra**

**Benha University**

**Research Article in fulfillment of the requirements of**

|  |  |
| --- | --- |
| **Department** | Engineering Mathematics and Physics |
| **Division** | Preparatory |
| **Academic Year** | 2019-2020 Preparatory |
| **Course name** | Computer |
| **Course code** | ECE001 |

**Title: -**

Programming Language

By:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Name | Edu mail | B.N |
| 1 | Kareem Taha  Abdulfatah Mohamed | Karim195667@feng.bu.edu.eg | 600 |

**Approved By:**

|  |  |
| --- | --- |
| Examiners committee | Signature |
| Dr.Ahmed Bayoumi |  |
| Dr.Shady Elmashad |  |
| Dr. Abdelhamid Attaby |  |

# Research objectives

Knowledge:

1-What is Programming Language? / 2-The importance of artificial language.

3-Style concepts, Types of Programming Languages.

4-History of programming languages. / 5-The Top Programming Languages.

6-Syntax and semantics.

# Abstract

Programming language is one amongst the foremost important languages of our time. Technology is wasted because it's employed in all electronics, from computers, telephones, tablets and every one electronic circuits. All machines and technology controlled by man and also the person will never need any manual machine. Every person should study programming languages in order that they will benefit him later when technology advances all told countries because it will become as we see it in fantasy movies that some are currently photographing.

**Table of contents:**

|  |  |
| --- | --- |
| **Subject / section** | **Page** |
| What is Programming Language? | 3,4 |
| Syntax and semantics and style concepts | 5 |
| Types of Programming Languages and History of programming languages | 6 |
| The Top Programming Languages | 7 |

# Introduction

Modern computers are incredibly fast, and getting faster all the time. However, computers even have some significant constraints: they only natively understand a limited set of commands, and must be told exactly what to try to to.

A Trojan horse (also commonly called an application) could be a set of instructions that the pc can perform so as to perform some task. The method of making a program is termed programming. Programmers typically create programs by producing ASCII text file (commonly shortened to code), which could be a list of commands typed into one or more text files.

The collection of physical computer parts that conjure a computer and execute programs is termed the hardware. When a Trojan horse is loaded into memory and also the hardware sequentially executes each instruction, this can be called running or executing the program.

A computer’s CPU is incapable of speaking C++. The limited set of instructions that a CPU can understand directly is termed code (or machine language or an instruction set).

Here could be a sample machine language instruction: 10110000 01100001

Back when computers were first invented, programmers had to put in writing programs directly in machine language, which was a really difficult and time consuming thing to try to to.

How these instructions are organized is beyond the scope of this introduction, but it's interesting to notice two things. First, each instruction consists of a sequence of

1’s and 0’s. Each individual 0 or 1 is termed a figure, or bit for brief. The quantity of bits that conjure one command vary -- for instance, some CPUs process instructions that are always 32 bits long, whereas another CPUs (such because the x86 family, which you're likely using) have instructions which will be a variable length.

Second, each set of binary digits is interpreted by the CPU into a command to try to to a really specific job, like compare these two numbers, or put this number therein memory location. However, because different CPUs have different instruction sets, instructions that were written for one CPU type couldn't be used on a CPU that didn’t share the identical instruction set. This meant programs generally weren’t portable (usable without major rework) to differing types of system, and had to be written everywhere again. Because machine language is so hard for humans to read and understand, programming language was invented. In a programming language, each instruction is identified by abrief abbreviation (rather than a group of bits), and names and other numbers will be used.

Here is that the same instruction as above in assembly language: mov al, 061h this makes assembly much easier to read and write than machine language. However, the CPU can't understand programming language directly. Instead, the programmer must be translated into machine language before it will be executed by the pc. This can be done by employing a program called an assembler. Programs written in assembly languages tend to be in no time, and assembly continues to be used today when speed is critical.

# Literature Review

## What is Programming Language?(1)

>>An artificial language could be a vocabulary and set of grammatical rules for instructing a computer or computing machine to perform specific tasks. The term artificial language usually refers to high-level languages, like BASIC, C, C++, COBOL, Java, FORTRAN, and Pascal.

Each artificial language incorporates a unique set of keywords (words that it understands) and a special syntax for organizing program instructions.

Computer programming language, any of assorted languages for expressing a group of detailed instructions for a computer. Such instructions will be executed directly after they are within the computer manufacturer-specific numerical form referred to as machine language, after a straightforward substitution process when expressed in a very corresponding programming language, or after translation from some “higher-level” language. Although there are many computer languages, relatively few are widely used.

A PL must be universal – capable of expressing any computation.

>A language without iteration or recursion wouldn't be universal.

>The lambda calculus – a language of recursive functions and zilch else – is universal.

## What is a programming language? (2)

A PL should be reasonably natural for expressing computations in its intended area.

>C is natural for systems programming.

>Java is natural for applications.

>Python is natural for scripting.

A PL must be implementable:

>it must be possible to run every program therein PL on a computer.

>as long because the computer has enough memory.

A PL should be capable of reasonably efficient implementation.

>Running a program shouldn't require an unreasonable amount of your time or memory. >what's reasonable depends on the context. E.g., Python is slow, but acceptable for scripting applications; it might not be acceptable for systems.

**Syntax and semantics:**

The syntax of a PL is worried with the shape of programs: how expressions, commands, declarations, and other constructs must be arranged to create a well-formed program. The semantics of a PL is worried with the meaning of well-formed programs: how a program could also be expected to behave when run on a machine.

Semantics underlies all programming, and language implementation. Syntax provides a structure on which semantics are often defined.

## Style concepts:(1)

Design concepts are the building blocks of PLs:

values and kinds – variables and storage – bindings and scope – procedural abstraction –

data abstraction – generic abstraction – processes and communication (not covered here).

## Style concepts:(2)

A paradigm could be a sort of programming, characterized by a range of key concepts.

>Functional programming focuses on values, expressions, and functions.

>Imperative programming focuses on variables, commands (“statements”), and procedures.

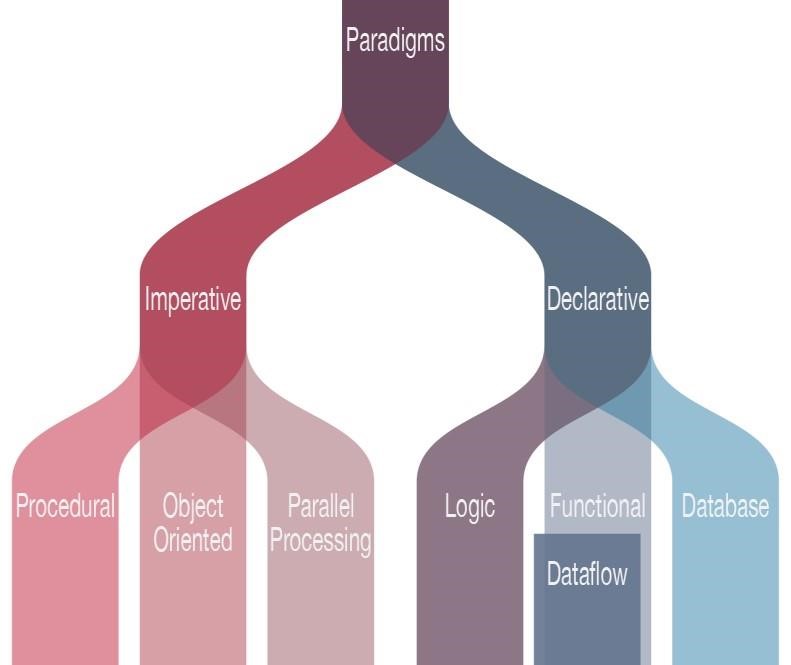
>Object-oriented (OO) programming focuses on objects, methods, and classes.

>Concurrent programming focuses on processes and communication.

Understanding of design concepts and paradigms enables us to pick out PLs for a project..

**Types of Programming Languages:**

The different types of programming languages are discussed below.



There Is C,PHP,Pascal.Fortten,Prel,LISP,Scheme,Object-oriented,Functional ,Logic ,C++Language And A lot Of Programming Language.

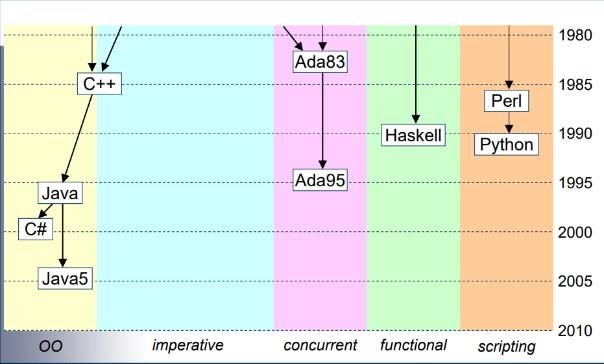
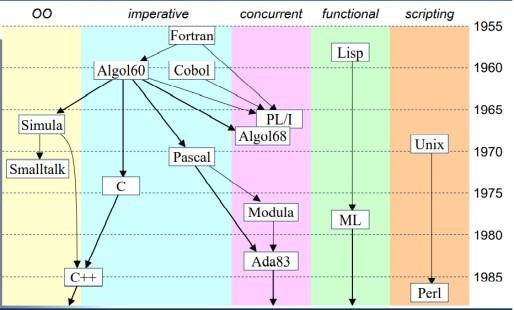
Example (Java Language):

The Java language could be a multi platform language that’s particularly helpful in networking. Of course, mostly this language is employed on the net with Java applets. However, this language is employed to style cross platform programs, Since it kind of like C++ in structure and syntax. For C++ programmers, Java language is incredibly easy to find out and it offers some advantages provided by object oriented

programming. Like reusability and it are often difficult to put in writing efficient code in Java. But, nowadays the speed of the Java language has increased and 1.5 version offers some good features for simple program making.

**History of programming languages:**

**The Top Programming Languages?**



According to IEEE Spectrum's interactive ranking, Python is that the top artificial language of 2017, followed by C, Java and C++. Of course, the selection of which language to use depends on the sort of computer the program is to run on, what style of program it's, and also the expertise of the programmer.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Python | C | Java | C++ | C# | R | JavaScript | PHP | Go | Swift |
| 100% | 99.7% | 99.5% | 97.1% | 87.7% | 87.7% | 85.6% | 81.2% | 75.1% | 73.7% |



# Results and discussion

Why you must Learn Several Programming Languages & Where to find out Them ?

Taking the time to find out a replacement programing language (whether at a giant Nerd Ranch bootcamp or elsewhere) could be a huge accomplishment, particularly for those people that are looking to search out full-time work as a developer. However, successfully learning one language doesn't mean that the work is completed. However, successfully learning one language doesn't mean that the work is completed. In fact, learning only one programing language can dramatically limit career opportunities and income potential, particularly as technology continues to evolve and extra programming languages begin to seem. You are Better Prepared for Success:

Good programmers don’t just know the way to code—they know the way to wrap their brains around a posh problem and produce efficient code to resolve it. And while most programming languages can do most jobs, each has its own strengths and weakness. Since you'll not be ready to produce the identical results with every language, it helps to grasp multiple. The more languages you learn, the larger content you'll have, and also the more likely you'll be ready to solve problems faster and more efficiently.

You Can Choose Which Jobs to Pursue:

While many companies are searching for developers in any given month, only a fraction of them could also be hiring for the language that you just know. By default, only knowing one programing language limits your job search to fewer roles. However, a bit like being bilingual gives people a footing in their job search, knowing multiple programming languages opens more possibilities to you and allows you to settle on which career path you'd wish to take. You have got more of a chance to choose the role (and the company) that you just love, instead of being cursed with limited options.

You Increase Your Salary Potential:

Trends in programming languages change over time and also the likelihood of you being required to find out a replacement language over the course of your career is nearly guaranteed. While most employers can pay for you to find out the new language if it's needed for business reasons, they're usually willing to pay more upfront for those that already know the language or those that have already proven that they need the power to find out quite one language. These people are more valuable to the corporate and thus are paid quite developers who only know (or care to know) one language. You Have More Fun:

Depending on your job responsibilities, you'll be doing the identical thing (or similar things) each day, which may quickly result in boredom. However, knowing multiple languages can keep your mind alert and your job responsibilities varied because you'll have the power to figure on multiple projects, even perhaps at the identical time. With new opportunities and new challenges, you're more likely to enjoy the work that you just do and have more fun with it within the process.

# Conclusions

Learning a programing language could be a lot like learning a distant languageonce you recognize the other, it's much easier to find out the third and fourth. The rationale for this can be because you hung out learning the way to learn while learning the second language, and this process becomes easier over time. You begin recognizing similarities between languages, which helps accelerate the training process and your ability to retain new information. It really isn’t as hard as seems!

Being a flexible developer implies that your skills never become outdated and you'll be able to better adapt to industry trends.

**References**

(1)A lot Of References Books:

1- Iverson, K. E., "A Programming Language", Wiley, 1962.

2-Ettinger, James (2004) Jacquard's Web, Oxford University Press 3-Aaby,

Anthony (2004). Introduction to Programming Languages.

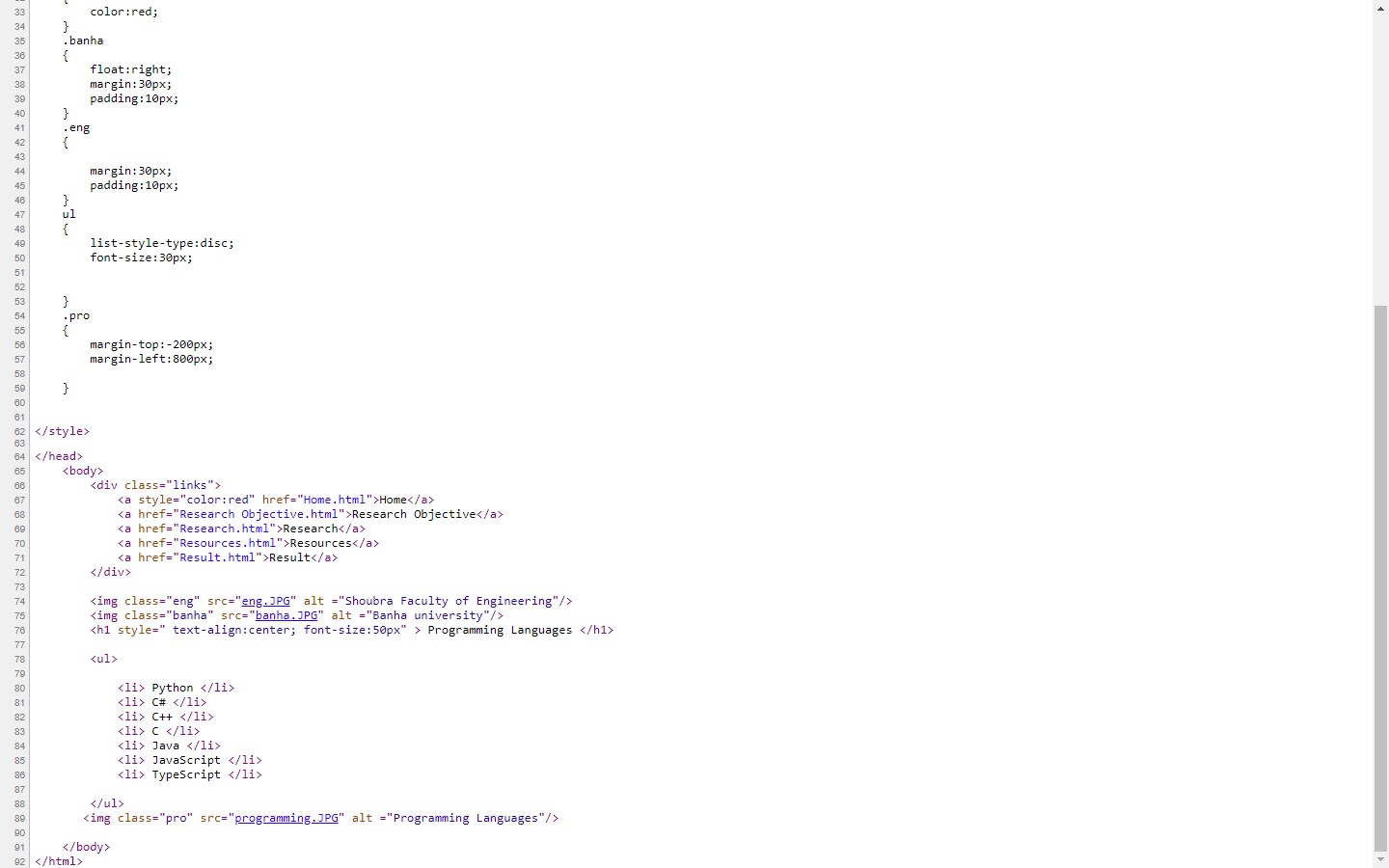
4-Pierce, Benjamin (2002). Types and Programming Languages.

1. Friend, E. H., "Sorting on Electronic Computer Systems", J. Assoc. Comp. Mach. 3, pp 134--168 (March 1956).
2. John C. Reynolds, "Some thoughts on teaching programming and programming languages".
3. David A Watt, University of Glasgow.

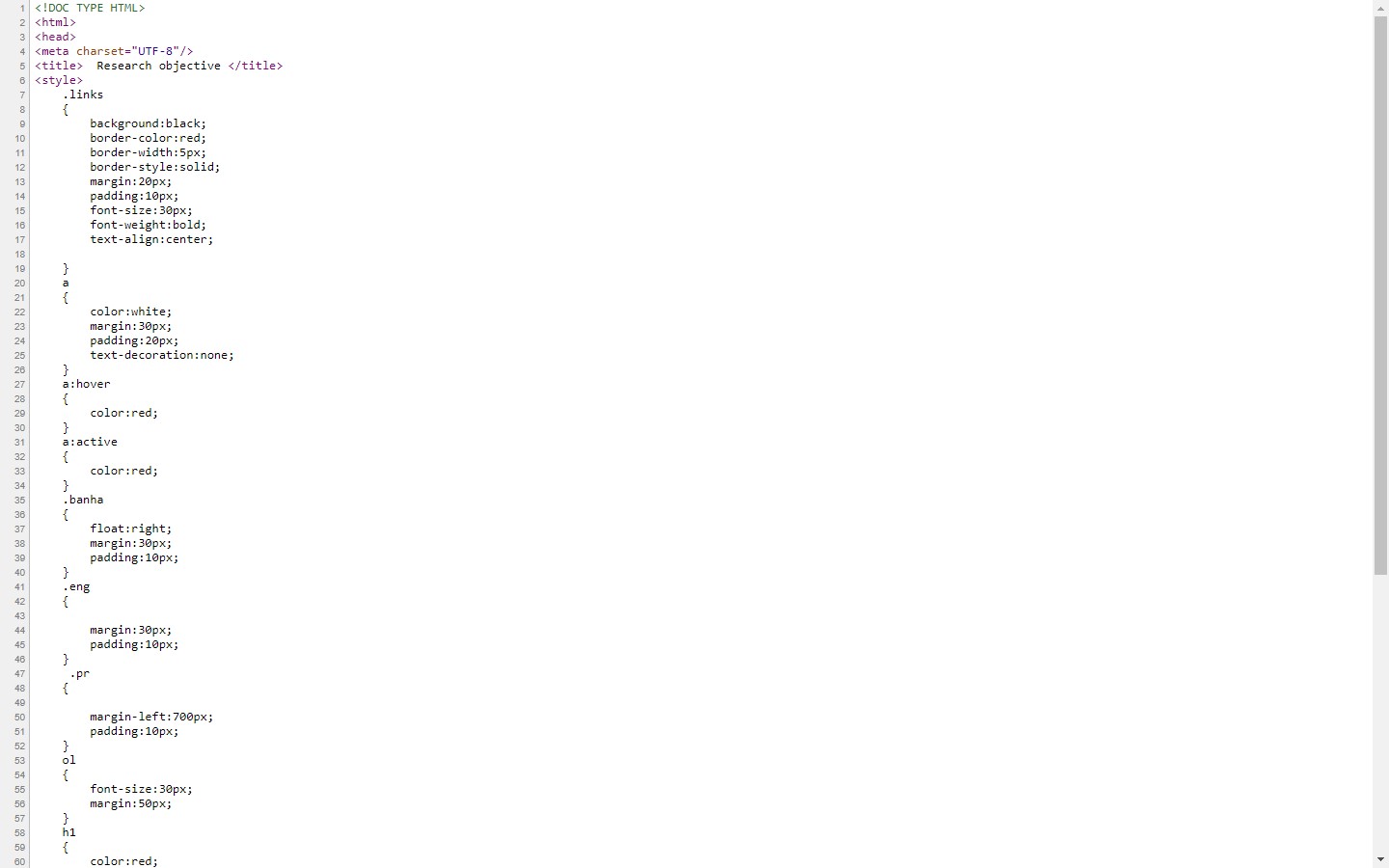
(2)A lot Of Sites:

1. https://www.typesnuses.com/types-of-programming-languages-with-differences/
2. https://www.linkedin.com/pulse/importance-computer-programmingazirulsyazwanjaffar
3. <https://dl.acm.org/doi/pdf/10.1145/236462.236561>
4. http://www.eecs.qmul.ac.uk/~mmh/ItP/aims.html
5. [https://06107hf4r-1104-y-https-academicebcom.mplbci.ekb.eg/levels/collegiate/article/computer-programming-language/25054](https://06107hf4r-1104-y-https-academic-eb-com.mplbci.ekb.eg/levels/collegiate/article/computer-programming-language/25054)
6. https://dl.acm.org/doi/10.1145/1460833.1460872

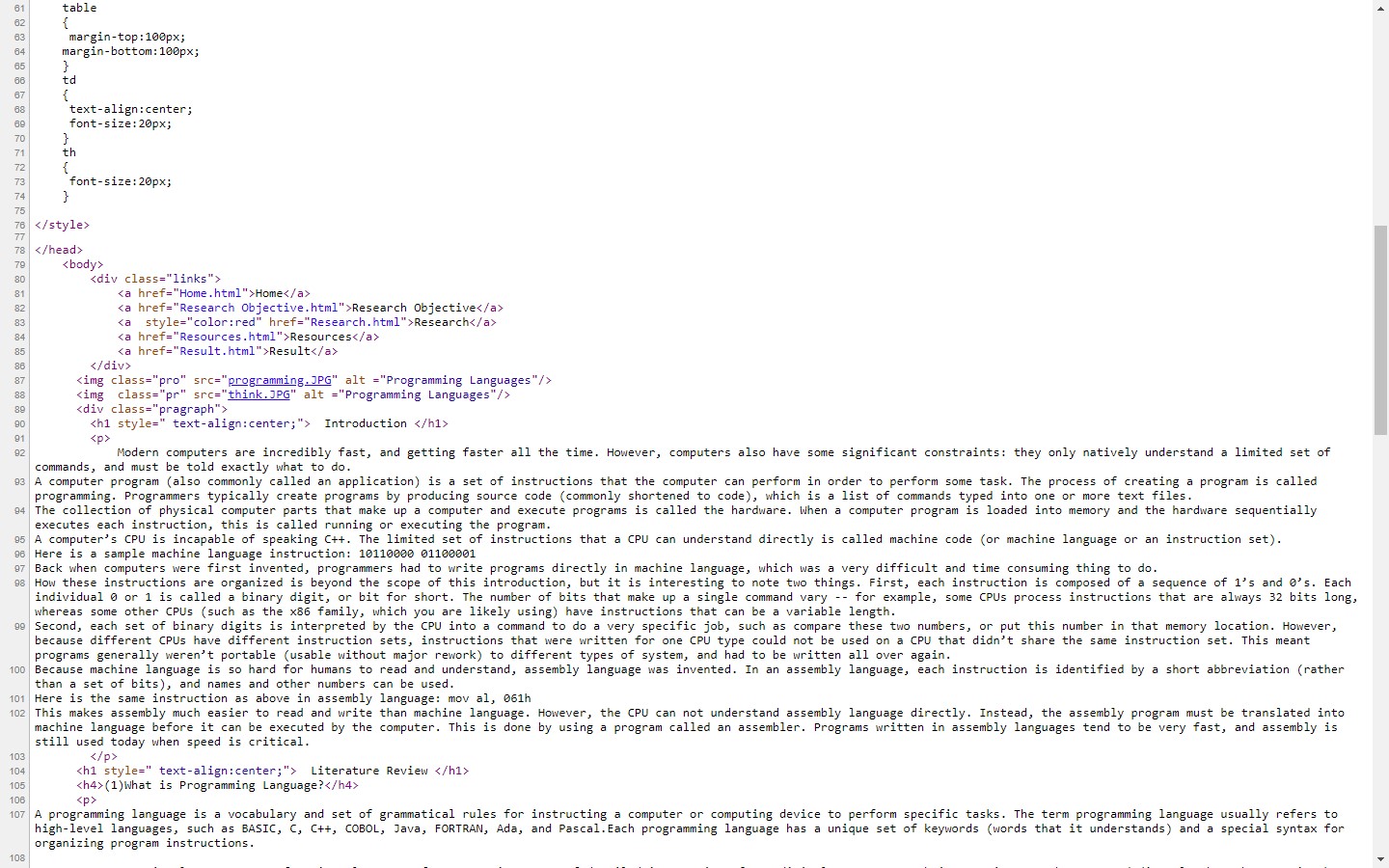
**Code of Home page**



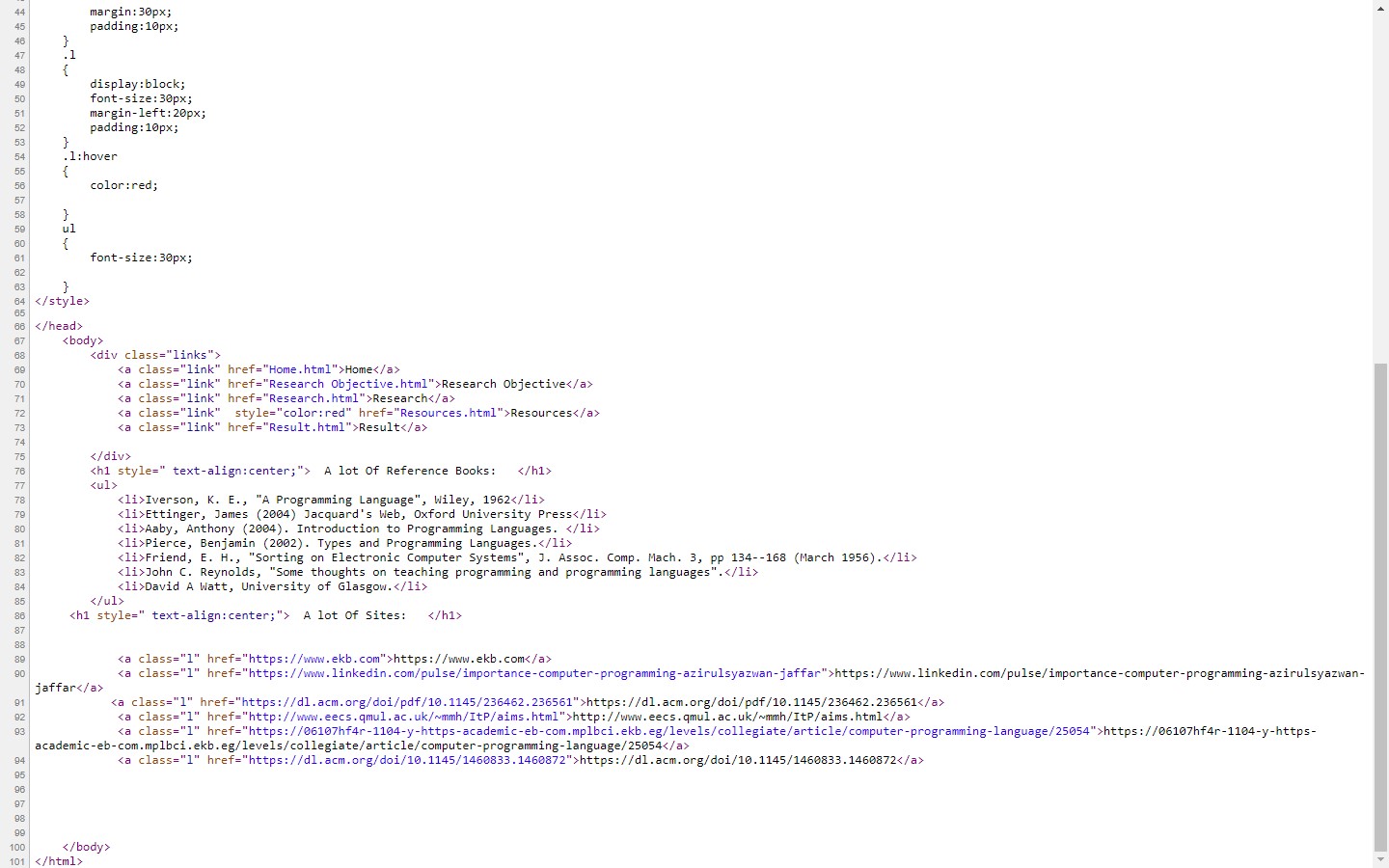
**Code of Research Objective**



**Code of Research**



**Code of Resources**



**Code of Result**



**Screen Shot From Page**

