Conference Paper

# Programming Languages that minimize response times of an Artificial Neuron

**Fernando Mejía[1], Byron Vaca[2], Eduardo Villa[2], Washington Luna[2], and Raul Rosero[2]**

[1]Facultad de Informática y Electrónica/Escuela de Ingeniería Electrónica en Control y Redes Industriales, Escuela Superior Politécnica de Chimborazo
[2]Facultad de Informática y Electrónica/Escuela de Ingeniería en Sistemas, Escuela Superior Politécnica de Chimborazo

## Abstract

In this paper we compare the different programming languages in number of three such as Java, C# and C++, which at the moment are the best in the international ranking for program development, to observe which of them minimizes the response time (output variable) of an artificial neuron algorithm (input variable). The results show how each of the programming languages gives a response time to the proposed algorithm. We discuss the results within the framework of the controversy of the artificial neuron vs programming languages.

**Keywords:** Artificial Intelligence, Neural Networks, Artificial Neuron, Programming Languages.

Corresponding Author:
Fernando Mejía
msmejiaedwinf@yahoo.com

Received: 28 July 2017
Accepted: 5 September 2017
Published: 30 January 2018

Publishing services provided by Knowledge E



Selection and Peer-review under the responsibility of the SIIPRIN Conference Committee.



## 1. INTRODUCCIÓN

El conjunto de órdenes que se dan al computador para que resuelva o ejecute una determinada instrucción se llama programa. Cuando nace la programación las órdenes o líneas de código se escribían en código de máquina o lenguaje de máquina que es el único lenguaje que entiende el microprocesador.

Cualquier notación para la descripción de algoritmos y estructuras de datos pueden llamarse lenguaje de programación; sin embargo nosotros requerimos de lenguajes de programación en donde podamos implementar esta neurona artificial.

En este trabajo se desarrolla un análisis de los mejores lenguajes de programación que hasta el año 2017 tienen auge respecto a la búsqueda de trabajo, los más populares y los de mayor demanda. Esto se realiza mediante varias estadísticas que cada uno de los sitios [1] han realizado un estudio y en base a eso nos brindan toda esa información.

Los lenguajes de programación existentes en el mercado de desarrollo de software y sistemas inteligentes hacen que el programador tenga a la mano una de las herramientas de automatización que mejor se adaptan al ser humano, debido a que la codificación en los mismos tiene mucho que ver con lo que las personas manejamos en la vida diaria.

En este sentido tenemos el algoritmo de aprendizaje de la neurona tipo perceptrón simple, la cual vamos a generar su código, para verificar el tiempo de respuesta que nos brinda cada uno de los lenguajes de programación en estudio, como son: JAVA, C# y C++, con el propósito de utilizar el lenguaje de programación que se demore menos en realizar esta operación en otros proyectos dentro de este campo. Luego selecciona los ciclos más óptimos adaptados al aprendizaje de la neurona artificial, con el fin de que aprenda basado en una función sigmoidea, ahora el algoritmo se tiene que adaptar a cada uno de los lenguajes en donde se va a programar con las mismas técnicas y estructuras para que no afecte en ningún sentido la respuesta que cada uno de ellos nos pueda brindar.

Al momento de realizar el análisis de estructuras y ciclos, con su correspondencia en el código, se deben efectuar actualizaciones en cada línea de programación, ya que cada uno de los lenguajes de programación posee su propio código, sin afectar el uso de la neurona artificial, para la cual se está haciendo el aprendizaje.

El algoritmo de aprendizaje de la neurona artificial que se va a ejecutar es para aprender una compuerta AND, NAND y OR, practica recomendada para poder ver la demora en el tiempo con cada una de las compuertas, por supuesto se deben ir observando el comportamiento de los pesos sinápticos y el umbral de la neurona que se va a ejecutar, tal como se muestra en la figura 1 propuesta por los autores, aquí se va usar solo dos entradas X1 y X2.

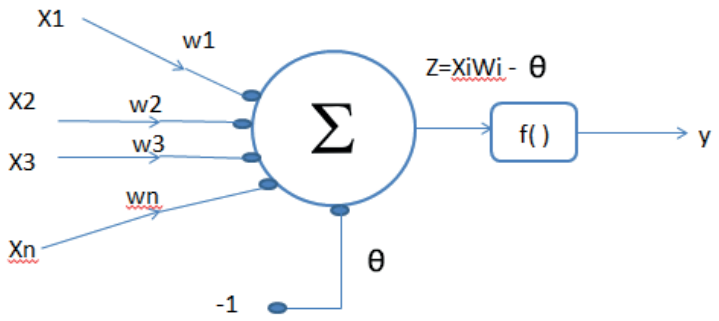


**Figura** 1: Modelo de neurona artificial a utilizar con las compuertas dentro de la observación.

La neurona empieza con la siguiente información mostrada en la tabla 1, donde se indican los datos para las diferentes compuertas a utilizar.

Tabla 1: Compuertas lógicas a usar para el aprendizaje de la neurona artificial.

| X1 | X2 | AND | OR | NAND |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 |

Independientemente de lo que resulte de cada una de las ejecuciones en los lenguajes usados, se debe indicar si la neurona aprendió o no, con qué pesos sinápticos, que umbral y en cuantos ciclos que determinaron la función de activación para obtener los resultados, la duración del ciclo tiene que estar forzosamente ligado al aprendizaje de nuestras entradas que se muestran en la tabla 1. La práctica de este algoritmo que se indica en la figura 2, debe indicarnos por cada lenguaje la demora para su ejecución:

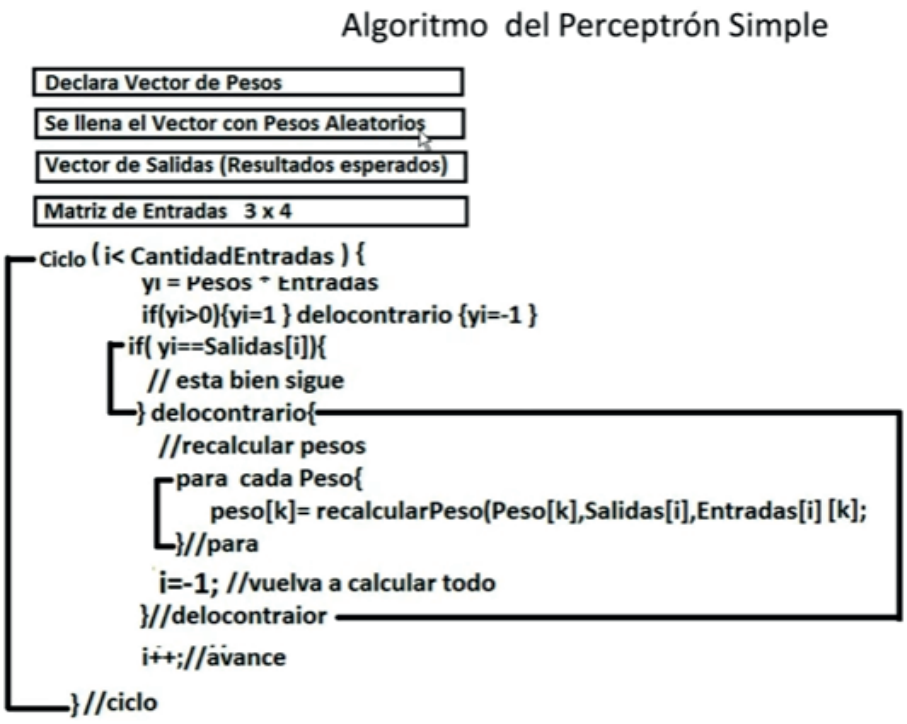


**Figura** 2: Algoritmo del Perceptrón Simple.

Debido a que la buena coordinación que tiene cada una de las estructuras utilizadas, la optimización de los ciclos y fases de los mismos se da cuenta que son herramientas fundamentales para tratar de resolver los problemas de aprendizaje de una neurona artificial y sobre todo resolver este tipo de problemas con los lenguajes de programación java, c# y c++, como se demuestra en numerosos trabajos[4], es por lo que

se ha diseñado un nuevo modelo, que es la principal aportación del presente artículo, para la resolución de dichos problemas.

## 2. MATERIALES Y MÉTODOS

Los materiales usados para este escenario integran una neurona artificial con un tipo de la misma como es el Perceptrón simple para resolver las compuertas AND, OR y NAND, además de los tres lenguajes de programación escogidos basados en las estadísticas de las empresas que miden cuál de ellos es el mejor. Por supuesto se usa fórmulas matemáticas y el método usado es el que se basa en el desarrollo de software estructurado con la metodología para la construcción de una neurona artificial para realizar los procesos como nosotros definimos a continuación en la figura 3:

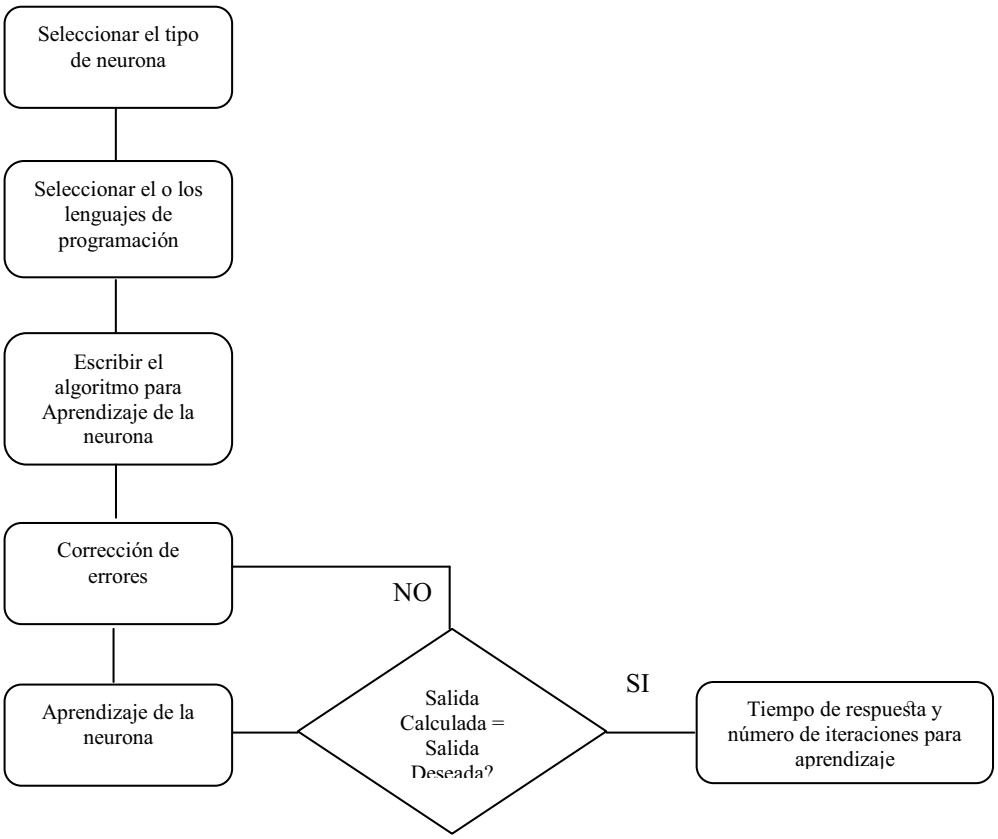


**Figura** 3: Metodología Aplicada.

Es una metodología a seguir, dado que permite tener una neurona que aprende basada en vectores de entrada dados y sus correspondientes pesos aleatorios, usando los tres lenguajes de programación como son Java, C# y C++ sin demasiados cambios en cada algoritmo.

## 2.1. Neurona Natural

La neurona que tenemos todos los seres humanos es la unidad principal y fundamental del sistema nervioso. Cada neurona es un procesador elemental que está ampliamente interconectado con otras para formar las redes de neuronas. En nuestro cerebro cohabitan unas cien mil millones de neuronas operando en paralelo.

Las neuronas tienen como entradas a las llamadas dendritas, salida llamada axón, núcleo o soma y la sinapsis que es la unión entre dos neuronas [2] como se muestra en la figura 4.

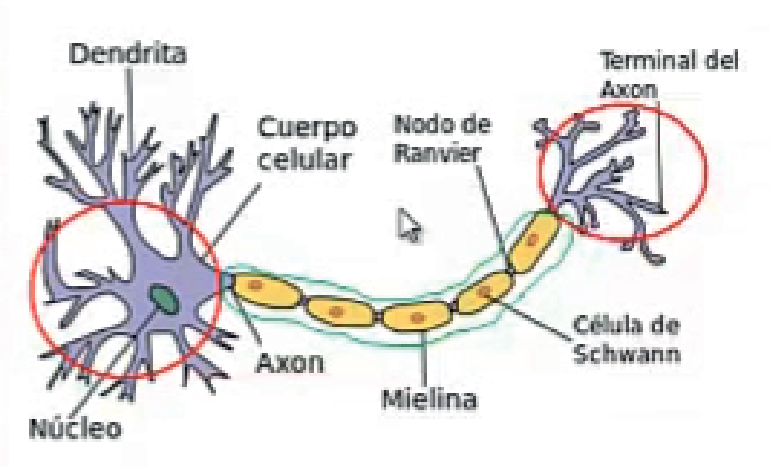


**Figura** 4: Neurona Natural que tiene el ser humano.

## 2.2. Neurona Artificial

Las neuronas artificiales siguen el mismo sentido de las que tiene nuestro cerebro humano. Este tipo de neuronas es un dispositivo simple de cálculo, que tiene como base un vector de entradas provenientes del mundo exterior o también puede ser de otras neuronas y que nos dan una respuesta o salida.

La neurona artificial como dijimos anteriormente tiene entradas y salida, éstas pueden ser binarias (digitales) o continuas (analógicas), van a depender siempre de un modelo y donde sean aplicadas, tal como se muestra en la figura1.

## 2.3. Perceptrón Simple

Una de las características más significativas de las redes neuronales es su capacidad para aprender a partir de alguna fuente de información interactuando con su entorno.

En 1958 el psicólogo Frank Rosenblat desarrolló un modelo simple de neurona basado en el modelo de McCulloch y Pitts y en una regla de aprendizaje basada en la corrección del error. A este modelo le llamó Perceptrón. Una de las características que más interés despertó de este modelo fue su capacidad de aprender a reconocer patrones. El Perceptrón está constituido por un conjunto de sensores de entrada que reciben los patrones de entrada a reconocer o clasificar y una neurona de salida que se ocupa de clasificar a los patrones de entrada en dos clases, según que la salida de la misma sea 1 (activada) o 0 (desactivada). Sin embargo, dicho modelo tenía muchas limitaciones, como por ejemplo, no es capaz de aprender la función lógica XOR. Tuvieron que pasar unos años hasta que se propusiera la regla de aprendizaje de retro propagación del error para demostrarse que el Perceptrón multicapa es un aproximador universal. [3]

Para las pruebas se utilizara las siguientes fórmulas que a continuación se detallan:

$$(\text{Net\_i}) = \sum_{i=0}^{n} Xi \cdot Wi$$

Hallar la salida calculada

$$Yo = \begin{cases} 1 & \text{si Net\_i} >= \Theta \\ 0 & \text{si Net\_i} < \Theta \end{cases}$$

Calcular el error

E = Yd – Yc

Si error distinto de cero, modifico los pesos

Wi(t+1) = Wi(t) + $\alpha$ EXi

Los pesos pueden ser aleatorios o dados por un experto humano en el tema que puede ser un ingeniero del conocimiento o una persona técnica o investigativa en el tema.

Defina los pesos asociados a las entradas (se recomienda un rango [-5,5] para que la red aprenda más rápido).

Defina el factor de aprendizaje $\alpha$ = 0.5 se recomienda, pero puede estar en el rango de (0.4 – 0.9).

## 2.4. Lenguajes de programación

Los lenguajes de programación son una herramienta informática que nos permiten resolver problemas de la vida real de una manera automática, estableciendo una interfaz gráfica muy sencilla y fácil de manejar, dado que al momento de programar las líneas de código son casi como el ser humano se maneja.

Los lenguajes de programación están compuestos principalmente por unas reglas de tipo semántico y sintáctico que lo definen como lenguaje informático. Pero esto no implica que un lenguaje de programación sea sinónimo de lenguaje informático. Sino que dentro del término lenguaje informático caben otros lenguajes que no son lenguajes de programación como es el caso del HTML.

El lenguaje de programación es el instrumento que permite al programador elegir los datos y variables que se deben usar, como serán guardados, como se presentaran al usuario final, que tipo de actuaciones se deben tomar bajo unos determinados supuestos. Para lo cual, los lenguajes de programación intentan estar lo más cerca posible al lenguaje humano o natural. [4]

Los lenguajes de programación son programas escritos que pasan primero por un proceso de compilación. Compilar es traducir al lenguaje máquina. Este paso es necesario para que la computadora pueda entender nuestras peticiones. Luego como segundo paso la computadora ejecuta y procesa los datos que le hemos dado.

Entre los lenguajes de programación más conocidos podemos citar a los siguientes: C, C++, C#, COBOL, Delphi, Eiffel, Ensamblador, FORTRAN, Java, Lisp, Logo, Lua, Modula, Object, Pascal, Perl, PHP, Smalltalk, TCL, VisualBasic, VisualFoxpro, JavaScript, Python, Ruby, etc.

## 2.5. Java

Java surgió en 1991 cuando un grupo de ingenieros de Sun Microsystems trataron de diseñar un nuevo lenguaje de programación destinado a electrodomésticos.

La reducida potencia de cálculo y memoria de los electrodomésticos llevó a desarrollar un lenguaje sencillo capaz de generar código de tamaño muy reducido. Debido a la existencia de distintos tipos de CPUs y a los continuos cambios, era importante conseguir una herramienta independiente del tipo de CPU utilizada. Desarrollan un código "neutro" que no depende del tipo de electrodoméstico, el cual se ejecuta sobre una "máquina hipotética o virtual" denominada Java Virtual Machine (JVM). Es la JVM quien interpreta el código neutro convirtiéndolo a código particular de la CPU utilizada. Esto permitía lo que luego se ha convertido en el principal lema del lenguaje: "Write Once, Run Everywhere" ("Escribir una vez, ejecutar siempre"). [4]

Existen distintos programas comerciales que permiten desarrollar código Java. La compañía Sun, creadora de Java, distribuye gratuitamente el Java Development Kit

(JDK). Se trata de un conjunto de programas y librerías que permiten desarrollar, compilar y ejecutar programas en Java.

Incorpora además la posibilidad de ejecutar parcialmente el programa, deteniendo la ejecución en el punto deseado y estudiando en cada momento el valor de cada una de las variables (es el denominado Debugger). Cualquier programador con un mínimo de experiencia sabe que una parte muy importante del tiempo destinado a la elaboración de un programa se destina a la detección y corrección de errores. Existe también una versión reducida del JDK, denominada JRE (Java Runtime Environment) destinada únicamente a ejecutar código Java (no permite compilar).

Los IDE (Integrated Development Environment), tal y como su nombre indica, son entornos de desarrollo integrados. En un mismo programa es posible escribir el código Java, compilarlo y ejecutarlo sin tener que cambiar de aplicación. Algunos incluyen una herramienta para realizar Debug gráficamente, frente a la versión que incorpora el JDK basada en la utilización de una consola (denominada habitualmente ventana de comandos de MS-DOS, en Windows NT/95/98) bastante difícil y pesada de utilizar.

## 2.6. C#

El lenguaje de programación C# fue creado por el danés Anders Hejlsberg en enero de 1999, que diseñó también los lenguajes Turbo Pascal y Delphi. El C# (pronunciado en inglés "C Sharp" o en español "C sostenido") es un lenguaje de programación orientado a objetos. Con este nuevo lenguaje se quiso mejorar con respecto de los dos lenguajes anteriores de los que deriva el C, y el C++. [5]

Con el C# se pretendió que incorporase las ventajas o mejoras que tiene el lenguaje JAVA. Así se consiguió que tuviese las ventajas del C, del C++, pero además la productividad que posee el lenguaje JAVA y se le denomino C#. Algunas de las características del lenguaje de programación C# son: Su código se puede tratar íntegramente como un objeto. Su sintaxis es muy similar a la del JAVA. Es un lenguaje orientado a objetos y a componentes. Armoniza la productividad del Visual Basic con el poder y la flexibilidad del C++. Ahorramos tiempo en la programación ya que tiene una librería de clases muy completa y bien diseñada.

Microsoft C# es un nuevo lenguaje de programación diseñado para crear un amplio número de aplicaciones empresariales que se ejecutan en.NET Framework. Supone una evolución de Microsoft C y Microsoft C++; es sencillo, moderno, proporciona seguridad de tipos y está orientado a objetos. El código creado mediante C# se

compila como código administrado, lo cual significa que se beneficia de los servicios de Common Language Runtime. Estos servicios incluyen interoperabilidad entre lenguajes, recolección de elementos no utilizados, mejora de la seguridad y mayor compatibilidad entre versiones. [6]

C# se presenta como Visual C# en el conjunto de programas Visual Studio.NET. Visual C# utiliza plantillas de proyecto, diseñadores, páginas de propiedades, asistentes de código, un modelo de objetos y otras características del entorno de desarrollo. La biblioteca para programar en Visual C# es.NET Framework.

C# es un lenguaje elegante, con seguridad de tipos y orientado a objetos, que permite a los desarrolladores crear una gran variedad de aplicaciones seguras y sólidas que se ejecutan en.NET Framework. Puede usar C# para crear aplicaciones cliente de Windows, servicios web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos y muchas, muchas más cosas. Visual C# proporciona un editor de código avanzado, prácticos diseñadores de interfaz de usuario, un depurador integrado y muchas otras herramientas que facilitan el desarrollo de aplicaciones basadas en el lenguaje C# y.NET Framework. [5-6]

En cuanto lenguaje orientado a objetos, C# admite los conceptos de encapsulación, herencia y polimorfismo. Todas las variables y métodos, incluido el método Main, el punto de entrada de la aplicación, se encapsulan dentro de las definiciones de clase. Una clase puede heredar directamente de una clase primaria, pero puede implementar cualquier número de interfaces. Los métodos que invalidan los métodos virtuales en una clase primaria requieren la palabra clave override como una manera de evitar redefiniciones accidentales. En C#, un struct es como una clase sencilla; es un tipo asignado en la pila que puede implementar interfaces pero que no admite herencia.

Además de estos principios básicos orientados a objetos, C# facilita el desarrollo de componentes de software mediante varias construcciones de lenguaje innovadoras, incluidas las siguientes:

Signaturas de método encapsulado llamadas delegados, que permiten notificaciones de eventos con seguridad de tipos.

Propiedades, que actúan como descriptores de acceso para variables miembro privadas. Atributos, que proporcionan metadatos declarativos sobre tipos en tiempo de ejecución. Comentarios de documentación XML insertados Language-Integrated Query (LINQ) que proporciona funcionalidades de consulta integradas en diversos orígenes de datos. [6]

Si tiene que interactuar con otro software de Windows, como objetos COM o archivos DLL nativos de Win32, puede hacerlo en C# mediante un proceso denominado "Interoperabilidad". La interoperabilidad permite que los programas de C# hagan casi todo lo que puede hacer una aplicación C++ nativa. C# admite incluso el uso de punteros y el concepto de código "no seguro" en los casos en los que el acceso directo a memoria es absolutamente crítico. [5-6]

## 2.7. C++

El lenguaje de programación C++ fue creado en los años 80 por Bjarne Stroustrup (también de Laboratorios Bell de AT&T) basado en el lenguaje C. El C++ es un lenguaje orientado a objetos al que se le añadieron características y cualidades de las que carecía el lenguaje C. [7]

De esta forma nació el C++ y como sucedía con el C depende mucho del hardware, tiene una gran potencia en la programación a bajo nivel, y se le agregaron herramientas para permitir programar a alto nivel. El C++ es uno de los lenguajes más potentes porque nos deja programar a alto y a bajo nivel, pero a su vez es difícil de aprender porque es necesario hacerlo casi todo manualmente.

El nombre fue propuesto por Rick Masciatti, al utilizarse C++ fuera de los laboratorios donde se creó. Con el nombre de C++ que quiso dar a entender que el C++ es una extensión del lenguaje C. El C++ es un lenguaje de programación híbrido, al que se le puede compilar. Una de las ventajas que ofrece este lenguaje es que es mucho más sencillo de aprender para los programadores que ya conocen el C.

El C++ mantiene una enorme compatibilidad con el C principalmente por dos razones: Por la gran cantidad de código C que existe, y para facilitar el paso de los programadores de C al nuevo lenguaje C++. [7]

Ya hemos dicho anteriormente que el lenguaje C++ es un lenguaje de programación orientado a objetos, pero no es un lenguaje orientado a objetos puro. El C++ nació como evolución del C, y desde su creación fue un lenguaje de programación hecho por programadores con un diseño muy práctico al que se le fueron añadiendo todos los elementos que se comprobaron eran necesarios sin tener en cuenta aspectos como su imagen, diseño, etc.

Todo esto ha ocasionado que sus detractores lo usen como argumento de crítica sobre el C++. Pero por otra parte precisamente esto es esto es lo que le da mayor valor, el ser un lenguaje más pragmático y sencillo que su antecesor el lenguaje C. [7]

## 2.8. Estadísticas de los Lenguajes de Programación

En la figura 5 se muestra un ranking de los mejores lenguajes de programación del 2014 y 2015. La columna de la derecha hace referencia al ranking de 2014, mientras que la de la izquierda se refiere a la popularidad en el año 2015. Los iconos representan a qué plataformas van dirigidos siendo, de izquierda a derecha en orden de aparición, web, móviles, PCs y microcontroladores.

Como podemos observar tenemos un top 5 inamovible compuesto por Java, C, C++, Python y C# que comprenden la élite de los lenguajes más utilizados. Ninguna sorpresa al respecto.

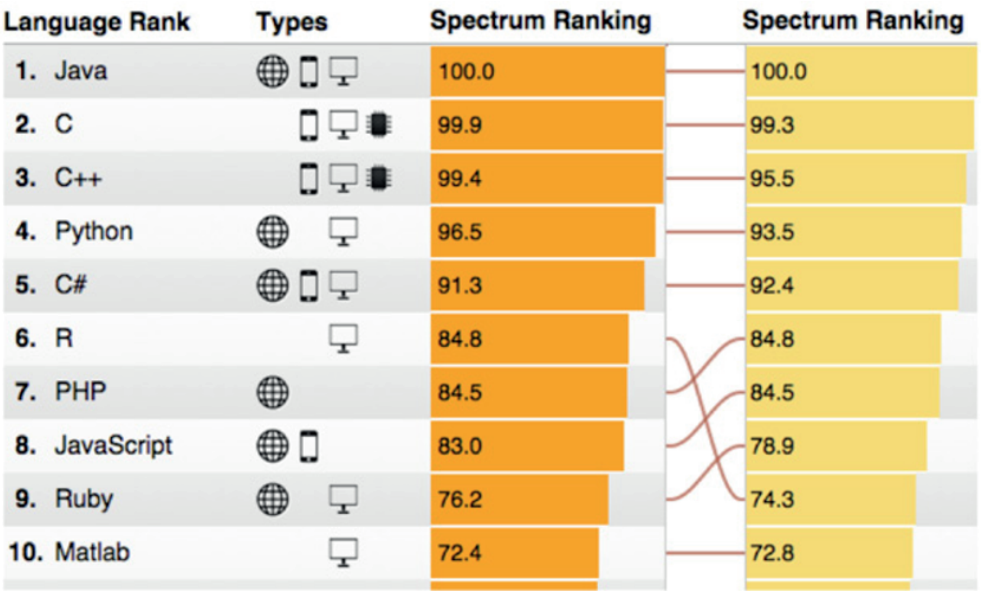


**Figura** 5: Ranking de los mejores lenguajes de programación en los años 2014 y 2015.

Los datos que se muestran en la figura 5 son obtenidos de IEEE Spectrum que ha desarrollado una aplicación web que, a partir de determinados criterios, establece el ranking de los lenguajes de programación más populares de la actualidad. Dicha aplicación toma datos de fuentes como GitHub, IEEE Xplore o CareerBuilder, por lo que nos asegura que los resultados tienen unas bases fiables. [8] tomado del sitio original, donde ya se muestran todos los lenguajes de programación que están dentro de los mejores [9].

Existe una estadística que muestra los lenguajes de programación que están dentro de los más populares y los mejor pagados desde el año 2015 mostrados en [10] y que se obtienen de la empresa TIOBE(The Importance of Being Earnest).

En la figura 6 se muestra el ranking de la empresa TIOBE que hace cada año, esta muestra nos indica los lenguajes de programación más populares a febrero del 2015.


**Figura** 6: Ranking de los lenguajes de programación según la empresa TIOBE.

TIOBE nos tiene un reporte de cuáles son los lenguajes de programación más populares, basándose en la cantidad de programadores cualificados en dichos lenguajes, cursos y estadísticas de los principales motores de búsqueda. [11]

El índice TIOBE (TIOBE, The Importance of Being Earnest) es un informe mensual que elabora y publica la empresa TIOBE Software BV. La misma es una empresa afincada en Eindhoven (Holanda) que se ha encargado de desarrollar y mejorar desde el año 2001 un algoritmo basado en la ejecución de infinidad de "queries" en los mayores motores de búsqueda del mundo (Google, MSN, Yahoo!, Baidu, Wikipedia o YouTube, entre otros). Con los resultados obtenidos, su algoritmo es capaz de cuantificar el uso en el mundo de los distintos lenguajes de programación. Es necesario resaltar que, en ningún caso, esta clasificación se basa en la calidad de uso de unos u otros lenguajes, simplemente cuantifica su uso y los clasifica de más a menos utilizados. [12]

En la figura 7 se muestra los lenguajes de programación mejor pagados en el mercado y en la industria de la ingeniería del software. [13]

A continuación se muestra en la figura 8 un resumen de los lenguajes de programación mejor pagados tanto en el back-end y front-end. [13]

Actualmente en el año 2017 tenemos los 10 mejores lenguajes de programación [14] tal como se muestra en la figura 9.

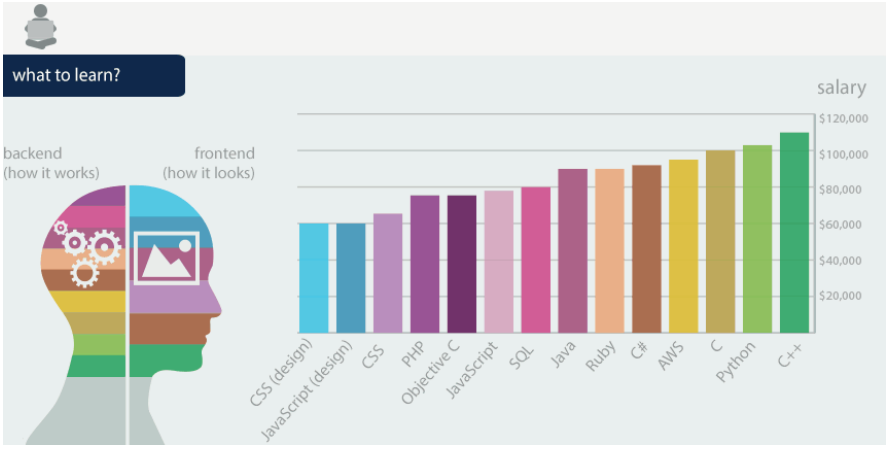


**Figura** 7: Lenguajes de programación mejor pagados en la industria de la ingeniería del software.

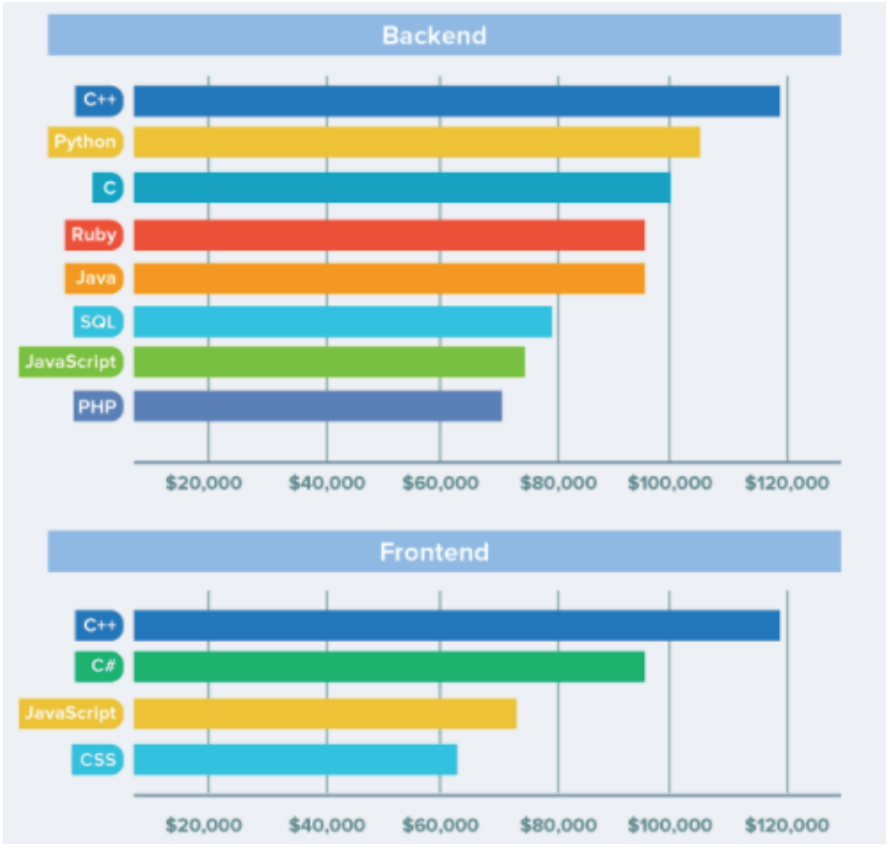


**Figura** 8: Lenguajes de programación mejor pagados en el mercado.

Por tal motivo y observando las figuras mostradas por cada una de las empresas de prestigio que visualizan estas estadísticas, se escoge a Java, C# y C++.

## 3. RESULTADOS Y DISCUSIÓN

| Jun 2017 | Jun 2016 | Change | Programming Language | Ratings | Change |
|---|---|---|---|---|---|
| 1 | 1 | | Java | 14.493% | -6.30% |
| 2 | 2 | | C | 6.848% | -5.53% |
| 3 | 3 | | C++ | 5.723% | -0.48% |
| 4 | 4 | | Python | 4.333% | +0.43% |
| 5 | 5 | | C# | 3.530% | -0.26% |
| 6 | 9 | ▲ | Visual Basic .NET | 3.111% | +0.76% |
| 7 | 7 | | JavaScript | 3.025% | +0.44% |
| 8 | 6 | ▼ | PHP | 2.774% | -0.45% |
| 9 | 8 | ▼ | Perl | 2.309% | -0.09% |
| 10 | 12 | ▲ | Assembly language | 2.252% | +0.13% |

**Figura** 9: Ranking de los lenguajes de programación del año 2017.

## 3.1. Código para el aprendizaje de una neurona en Java

En la figura 10 se indica el código Java para que aprenda una neurona la compuerta AND, se tiene 2 entradas y una salida, en la matriz "x" las dos primeras columnas son las entradas de la misma y la tercera columna es la salida deseada, es decir lo que tiene que aprender la neurona.

```
1
2   package neurona_perceptron;
3   /**
4    * @author Fernando Mejia
5    */
6   public class Neurona_Perceptron {
7       public static void main(String[] args) {
8           // TODO code application logic here
9           int x[][] = new int[4][5];
10          int y[] = new int[4];
11          int ite[] = new int[4];
12          int i,j,e,ban,cont=1;
13          double w0=10.5,w1=6.5,net,alfa=0.5,umbral=1;
14          //llenado de la matriz de entrada y salida
15          x[0][0]=0;x[1][0]=0;x[2][0]=1;x[3][0]=1;
16          x[0][1]=0;x[1][1]=1;x[2][1]=0;x[3][1]=1;
17          x[0][2]=0;x[1][2]=0;x[2][2]=0;x[3][2]=1;
18
19          for(i=0;i<4;i++)
20          {
21              ban=0;
22
23              net=(x[i][0]*w0)+(x[i][1]*w1);
neurona_perceptron.Neurona_Perceptron > main > for (i = 0; i < 4; i++) > if (ban
```

**Figura** 10: Código en Java con las declaraciones de variables y la matriz de entrada y salida, para el aprendizaje de la neurona.

En la figura 11 se indica el uso de las fórmulas que tenemos para el aprendizaje de la neurona y las condiciones a cumplir, mientras que en la figura 12 tenemos el código en


**Figura** 11: Fórmulas y código en Java ¿para el aprendizaje de la neurona.

java para el cálculo de las iteraciones que realiza la neurona para que pueda aprender la compuerta AND.

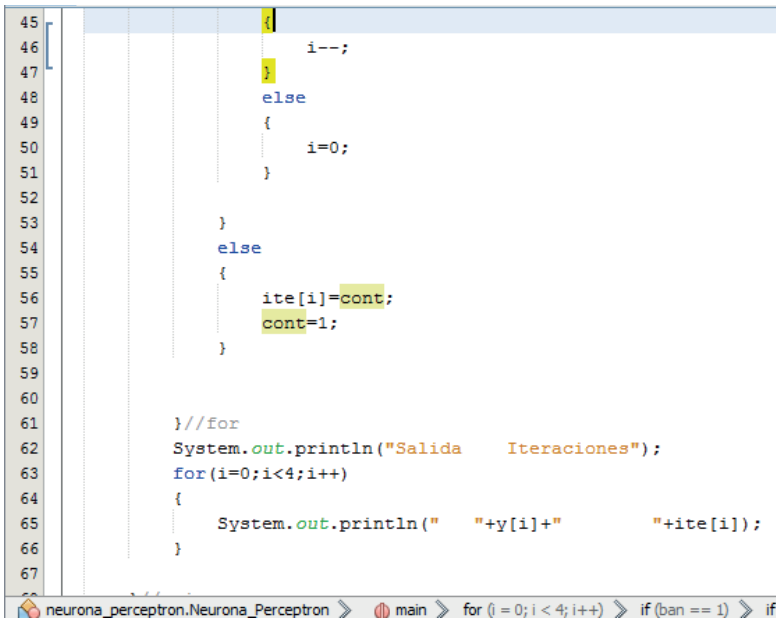


**Figura** 12: Código en java con el cálculo para las iteraciones hace la neurona.

En la figura 13 indicamos la respuesta de la neurona a cada salida deseada con las correspondientes iteraciones que tiene en cada entrada para poder aprender, además se indica el tiempo en milisegundos que se demora en obtener la respuesta deseada.

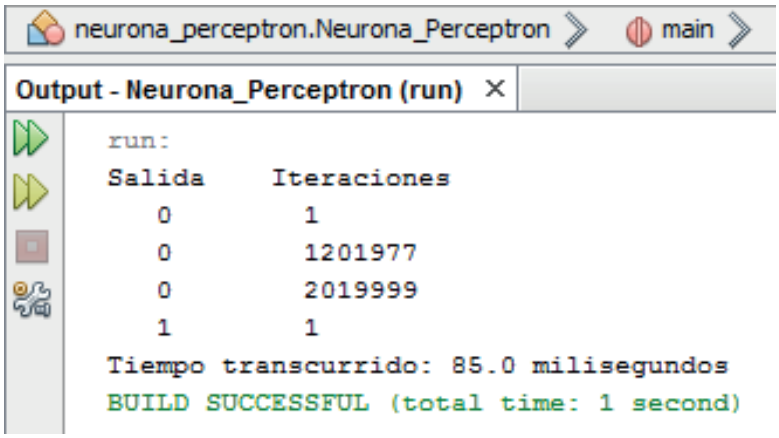


**Figura** 13: Respuesta a la ejecución del código en java para el que aprendizaje de la neurona y su tiempo en milisegundos.

El código que se utiliza en Java para poder obtener el tiempo de respuesta es el siguiente:

```
double Tinicio, Tfin, tiempo;
Tinicio = System.currentTimeMillis();
.....Código de la Neurona
Tfin = System.currentTimeMillis();
tiempo = Tfin - Tinicio;
System.out.println("Tiempo transcurrido: " + tiempo + " milisegundos");
```

## 3.2. Código para el aprendizaje de una neurona en C#

En la figura 14 se indica el código C# para que aprenda una neurona la compuerta AND, se tiene 2 entradas y una salida, en la matriz "x" las dos primeras columnas son las entradas de la misma y la tercera columna es la salida deseada, es decir lo que tiene que aprender la neurona.

En la figura 15 se indica el código en C# utilizando las fórmulas que tenemos para el aprendizaje de la neurona y las condiciones a cumplir.

En la figura 16 se indica el código en C# para el cálculo de las iteraciones que realiza la neurona para que pueda aprender la compuerta AND.

En la figura 17 indicamos la respuesta en C# de la neurona a cada salida deseada con las correspondientes iteraciones que tiene en cada entrada para poder aprender, además se indica el tiempo en milisegundos que se demora en obtener la respuesta deseada.

Program.cs*

Neurona_Perceptron.Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Neurona_Perceptron
{
    // realizado por Fernando Mejia
    // y Byron Vaca
    class Program
    {
        static void Main(string[] args)
        {
            int[,] x = new int[4,5];
        int[] y = new int[4];
        int[] ite = new int[4];
        int i,e,ban,cont=1;
        double w0=10.5,w1=6.5,net,alfa=0.5,umbral=1;
        //llenado de la matriz de entrada y salida
        x[0,0]=0;x[1,0]=0;x[2,0]=1;x[3,0]=1;
        x[0,1]=0;x[1,1]=1;x[2,1]=0;x[3,1]=1;
        x[0,2]=0;x[1,2]=0;x[2,2]=0;x[3,2]=1;

        for(i=0;i<4;i++)
        {
            ban=0;

            net=(x[i,0]*w0)+(x[i,1]*w1);
            if(net>=umbral)
```

**Figura** 14: Código en C# con las declaraciones de variables y la matriz de entrada y salida, para el aprendizaje de la neurona.

El código que se utiliza en C# para poder obtener el tiempo de respuesta es el siguiente:

Stopwatch cronometro = new Stopwatch(); cronometro.Start();

..... Código de la neurona

cronometro.Stop();

TimeSpan ts = cronometro.Elapsed;

System.Console.WriteLine("Tiempo transcurrido: " + (ts.TotalMilliseconds) / 10 + " milisegundos");

System.Console.ReadLine();

```
            if(net>=umbral)
            {
                y[i]=1;
            }
            else
            {
                y[i]=0;
            }

            if(y[i]!=x[i,2])
            {
                e=x[i,2]-y[i];
                w0=w0+alfa*e*x[i,0];
                w1=w1+alfa*e*x[i,1];
                ban=1;
                cont++; //contador de iteraciones

            }
            if(ban==1)
            {
                if(i!=0)
                {
                    i--;
                }
                else
                {
                    i=0;
                }

            }
```

**Figura** 15: Fórmulas y código en C# ¿para el aprendizaje de la neurona.

```
            else
            {
                ite[i]=cont;
                cont=1;
            }


        }//for
        System.Console.WriteLine("Salida    Iteraciones");

        for(i=0;i<4;i++)
        {
            System.Console.WriteLine("   "+y[i]+"        "+ite[i]);

        }
        System.Console.ReadLine();
        }//main
    }
}
```

**Figura** 16: Código en C# con el cálculo para las iteraciones que hace la neurona.

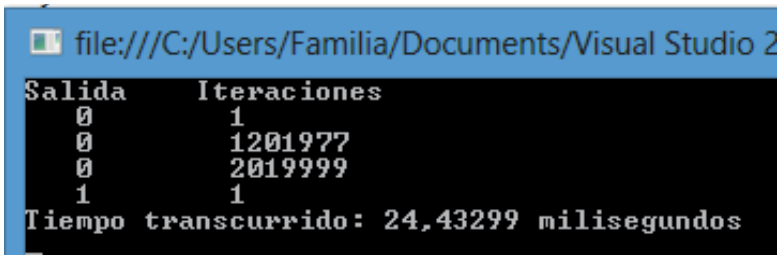


**Figura** 17: Respuesta a la ejecución del código en C# para el aprendizaje de la neurona y su tiempo en milisegundos.

## 3.3. Código para el aprendizaje de una neurona en C++

En la figura 18 se indica el código C++ para que aprenda una neurona la compuerta AND, se tiene 2 entradas y una salida, en la matriz "x" las dos primeras columnas son las entradas de la misma y la tercera columna es la salida deseada, es decir lo que tiene que aprender la neurona.

En la figura 19 se indica el código en C++ utilizando las fórmulas que tenemos para el aprendizaje de la neurona, las condiciones a cumplir y las iteraciones que realiza.

En la figura 20 indicamos la respuesta en C++ de la neurona a cada salida deseada con las correspondientes iteraciones que tiene en cada entrada para poder aprender, además se indica el tiempo en milisegundos que se demora en obtener la respuesta deseada.

El código que se utiliza en C++ para poder obtener el tiempo de respuesta es el siguiente:

```
clock_t sec1=clock();
  double tiempo;
.....Código de aprendizaje de la Neurona
tiempo = ((double)clock() - sec1)/CLOCKS_PER_SEC;
printf("Tiempo transcurrido: %.16g milisegundos", tiempo*1000);
getch();
```

## 3.4. Eficiencia del código planteado

Los datos que se indican en la tabla 2 son los que se realizaron para cada ejecución en el lenguaje de programación Java en cada instante de tiempo durante días y semanas en 6 meses, cada columna tiene 11 datos promedios siendo 22 datos los escogidos.

```
#include <stdio.h>
#include <conio.h>
#include <iostream.h>
// realizado por Fernando Mejia
// y eduardo villa
main()
{
   int x[4][5];
   int y[4];
   int ite[4];
   int i,j,e,ban,cont=1;
   double w0=10.5,w1=6.5,net,alfa=0.5,umbral=1;
   //llenado de la matriz de entrada y salida
        x[0][0]=0;x[1][0]=0;x[2][0]=1;x[3][0]=1;
        x[0][1]=0;x[1][1]=1;x[2][1]=0;x[3][1]=1;
        x[0][2]=0;x[1][2]=0;x[2][2]=0;x[3][2]=1;

        for(i=0;i<4;i++)
        {
            ban=0;

            net=(x[i][0]*w0)+(x[i][1]*w1);
            if(net>=umbral)
            {
                y[i]=1;
            }
            else
            {
                y[i]=0;
            }

            if(y[i]!=x[i][2])
            {
                e=x[i][2]-y[i];
                w0=w0+alfa*e*x[i][0];
                w1=w1+alfa*e*x[i][1];
                ban=1;
```

**Figura** 18: Código en C++ con las declaraciones de variables y la matriz de entrada y salida, para el aprendizaje de la neurona.

Gráficos estadísticos de las muestras encontradas en Java para el aprendizaje de una neurona se indican en la figura 21 y 22.

La figura 21 muestra los primeros datos promedio obtenidos en las 11 semanas iniciales que están en la segunda columna de la tabla 2, que se ejecutó el algoritmo planteado en este lenguaje de programación, mientras que en la figura 22 se muestran los siguientes 11 datos promedio obtenidos en las semanas finales que se encuentran en la tercera columna de la tabla 2.

En la figura 21 el dato mayor es 132 ms (milisegundos), el dato menor es 45 ms, mientras que en la figura 22 el dato mayor es 179 ms y el dato menor 42 ms.

```
                cont++; //contador de iteraciones

            }
            if(ban==1)
            {
                if(i!=0)
                {
                    i--;
                }
                else
                {
                    i=0;
                }

            }
            else
            {
                ite[i]=cont;
                cont=1;
            }


        }//for
        printf("Salida    Iteraciones \n");
        for(i=0;i<4;i++)
        {
            printf("   %d        %d \n",y[i],ite[i]);
        }
        getch();
    }
```

**Figura** 19: Fórmulas y código en C++ ¿para el aprendizaje de la neurona y al final del código sus iteraciones.

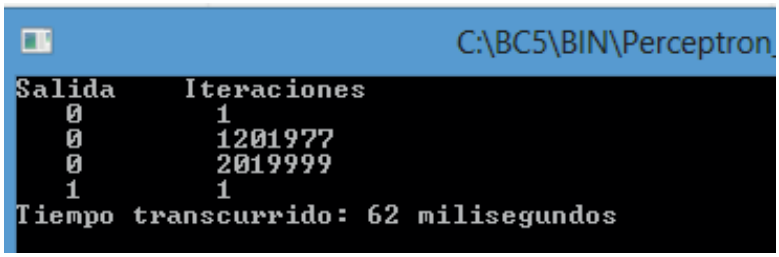


**Figura** 20: Respuesta a la ejecución del código en C++ para el aprendizaje de la neurona y su tiempo en milisegundos.

Los datos de la columna 2 y 3 de la tabla 2, como los de la figura 21 y figura 22 nos dan como resultado que 42 milisegundos es el tiempo mínimo que se demora Java para que la neurona aprenda la compuerta AND, tal y cual como se muestra en la figura 22.

Los datos que se indican en la tabla 3 son los que se realizaron para cada ejecución en C# en cada instante de tiempo durante días y semanas en 6 meses, cada columna tiene 11 datos promedios siendo 22 datos los escogidos.

Gráficos estadísticos de las muestras encontradas en C# para el aprendizaje de una neurona se indican en la figura 23 y 24.

Tabla 2: Muestras de tiempos en milisegundos del código ejecutado en Java para el aprendizaje de una neurona.

| | JAVA | |
|---|---|---|
| 1 | 132 | 179 |
| 2 | 101 | 106 |
| 3 | 77 | 93 |
| 4 | 81 | 90 |
| 5 | 81 | 144 |
| 6 | 93 | 62 |
| 7 | 79 | 112 |
| 8 | 83 | 157 |
| 9 | 62 | 74 |
| 10 | 81 | 61 |
| 11 | 45 | 42 |

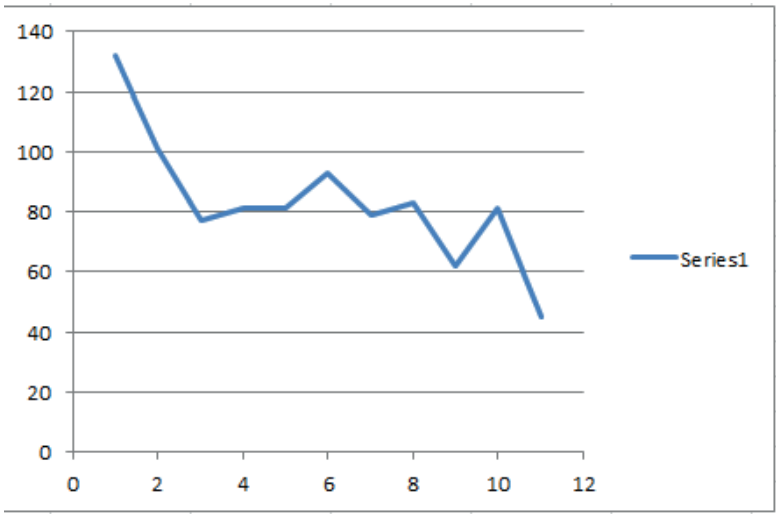


**Figura** 21: Primeros datos obtenidos para el código en Java.

La figura 23 nos indica los primeros datos promedio obtenidos en las 11 semanas iniciales que están en la segunda columna de la tabla 3, que se ejecutó el algoritmo planteado en este lenguaje de programación C#, mientras que en la figura 24 se muestran los siguientes 11 datos promedio obtenidos en las semanas finales que se encuentran en la tercera columna de la tabla 3.

En la figura 23 el dato mayor es 61,32 ms (milisegundos), el dato menor es 20,1 ms, mientras que en la figura 24 el dato mayor es 63,3 ms y el dato menor es 20,01 ms.

Los datos de la columna 2 y 3 de la tabla 3, como los de la figura 23 y figura 24 nos dan como resultado que 20,01 milisegundos es el tiempo mínimo que se demora C# para que la neurona aprenda la compuerta AND, tal y cual como se muestra en la figura 24.

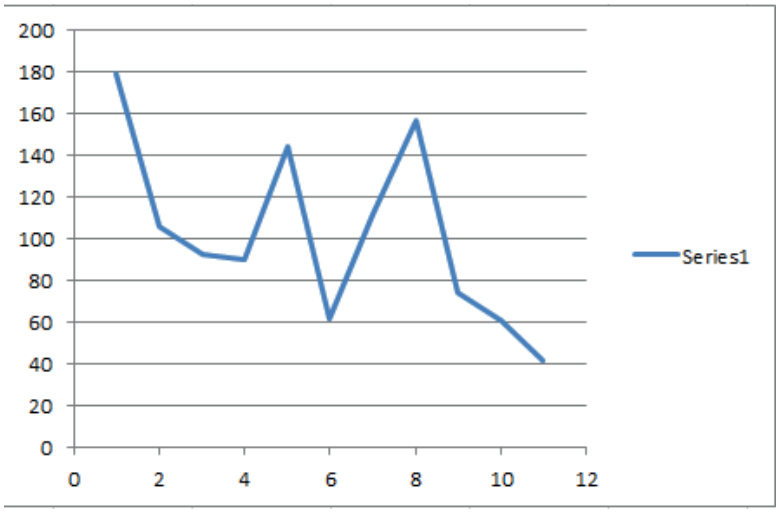


**Figura** 22: Datos fianles obtenidos para el código en Java.

Tabla 3: Muestras de tiempos en milisegundos del código ejecutado en C# para el aprendizaje de una neurona.

| | C# | |
|---|---|---|
| 1 | 20,6 | 63,3 |
| 2 | 21,9 | 40,3 |
| 3 | 20,8 | 23,4 |
| 4 | 20,9 | 24,2 |
| 5 | 32,6 | 21,1 |
| 6 | 34,5 | 20,01 |
| 7 | 61,32 | 30,2 |
| 8 | 20,1 | 21,2 |
| 9 | 40,4 | 30,3 |
| 10 | 21,3 | 45,2 |
| 11 | 20,2 | 60,2 |

La información que se indica en la tabla 4 son los que se realizaron para cada ejecución en C++ en cada instante de tiempo durante días y semanas en 6 meses, cada columna tiene 11 datos promedios siendo 22 datos los escogidos..

Gráficos estadísticos de las muestras encontradas en C++ para el aprendizaje de una neurona se indican en la figura 24 y 25.

La figura 25 nos indica los primeros datos promedio obtenidos en las 11 semanas iniciales que están en la segunda columna de la tabla 4, que se ejecutó el algoritmo planteado en este lenguaje de programación C++, mientras que en la figura 26 se muestran los siguientes 11 datos promedio obtenidos en las semanas finales que se encuentran en la tercera columna de la tabla 3.

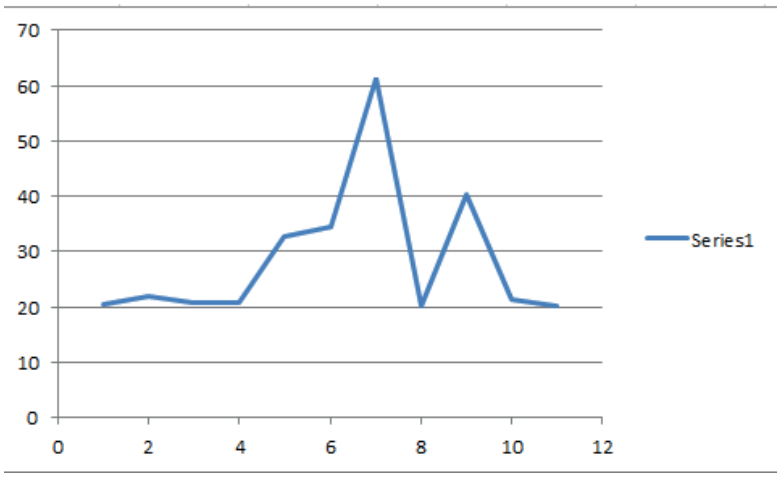


**Figura** 23: Primeros datos obtenidos para el código en C#.

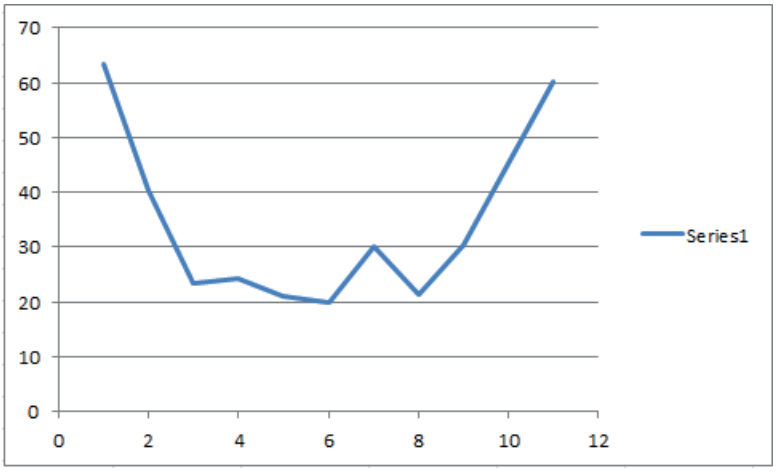


**Figura** 24: Datos finales obtenidos para el código en C#.

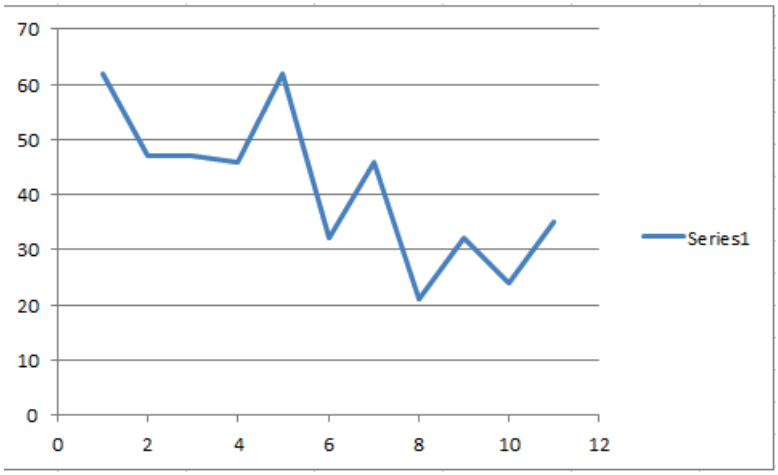


**Figura** 25: Primeros datos obtenidos para el código en C++.

En la figura 25 el dato mayor es 62 ms (milisegundos), el dato menor es 21 ms, mientras que en la figura 26 el dato mayor es 62 ms y el dato menor es 23 ms.

Tabla 4: Muestras de tiempos en milisegundos del código ejecutado en C++ para el aprendizaje de una neurona.

| | C++ | |
|---|---|---|
| 1 | 62 | 47 |
| 2 | 47 | 46 |
| 3 | 47 | 46 |
| 4 | 46 | 47 |
| 5 | 62 | 46 |
| 6 | 32 | 46 |
| 7 | 46 | 62 |
| 8 | 21 | 47 |
| 9 | 32 | 32 |
| 10 | 24 | 46 |
| 11 | 35 | 23 |

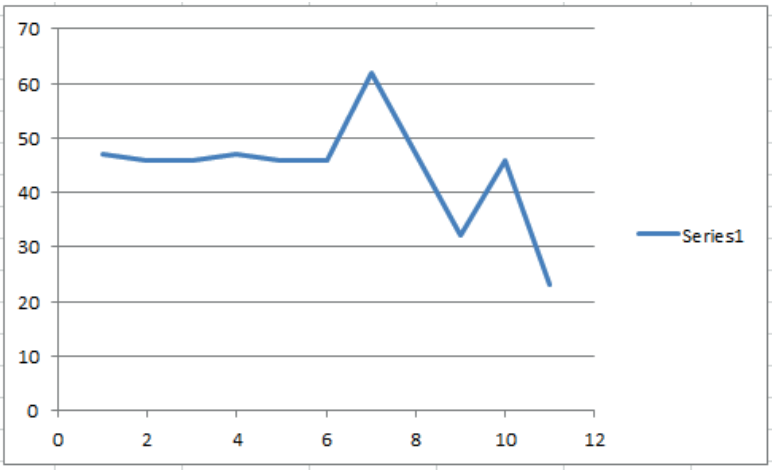


**Figura** 26: Datos finales obtenidos para el código en C++.

Los datos de la columna 2 y 3 de la tabla 4, como los de la figura 25 y figura 26 nos dan como resultado que 21 milisegundos es el tiempo mínimo que se demora C++ para que la neurona aprenda la compuerta AND, tal y cual como se muestra en la figura 25.

## 3.5. Discusión

La toma de decisiones la debe generar el código en cada uno de los lenguajes planteados para que la neurona aprenda en el instante de tiempo en las tablas 2, 3 y 4 para las iteraciones en cada una de las entradas con cada peso sináptico y un umbral de activación referente, la función de activación nos da una respuesta con respecto a

cada salida calculada dentro de la neurona, en el caso planteado que es la de obtener un aprendizaje igual a una compuerta AND y tomar las decisiones correctas para que el flujo de la información sea dinámico, resulta muy beneficioso utilizar esto.

La dimensión del tiempo en cuanto a respuesta del sistema operativo, ya que se utilizó Windows 8.1 esta medido en milisegundos. Los datos que se indican en las tablas se obtienen de las ejecuciones de cada código y en los diferentes lenguajes de programación estudiados, que están de igual manera en cada uno de ellos, siendo el código el mismo, pero como se puede observar solo se diferencian en ciertos aspectos de la definición de variables en cuanto a vectores o matrices y cuando se quiere obtener o visualizar el resultado ya que cada lenguaje tiene sus propias funciones para este tipo, también para medir los tiempos en cada uno son diferentes por supuesto, ellos permiten medir en milisegundos lo que se demora el código en dar la respuesta.

Ahora también hay que decir que cada instante que se vaya ejecutando el código, el mismo irá aprendiendo a cada instante de tiempo en el día, semanas, o los seis meses como en este caso, se debe mencionar que el código esta lo más simple posible para que ustedes estimados lectores puedan entenderlo, se utiliza estructuras condicionales simples como el if y una estructura repetitiva como es el for.

Se debe decir que cuando se desarrollen este tipo de código en sentido empresarial, tener en cuenta que el almacenamiento se realice en una base de datos y se utilice el mejor que basado en este estudio nos dio que es C# con 20,01 milisegundos.

## 4. CONCLUSIONES

En este artículo se ha demostrado que con una herramienta de programación para codificar los algoritmos se puede lograr llegar a la respuesta deseada, que en este caso fue que aprenda una compuerta AND la neurona artificial.

Sin dejar de lado la metodología para la construcción de los programas que puede ser estructurada u orientada a objetos y selección del mejor lenguaje basado en su respuesta al tiempo que se demora, también hay que saber hacer el número de iteraciones y cuál es la mejor codificación en ese momento para que el código que se está ejecutando nos pueda mostrar un resultado óptimo y en un mínimo tiempo.

El uso de las diferentes estructuras de programación como son las condicionales y repetitivas es de gran ayuda para este tipo de trabajos, pero hay que saber escogerlos, al momento de realizarlos en la realidad.

El número de variables aumenta o disminuye de acuerdo con el número de resultados que nosotros le demos, cuanto menos variables tengamos se puede llegar a la mejor solución y con un tiempo menor.

El número de matrices o vectores es de acuerdo a la practicidad y relación que puedan tener con la neurona que deseamos que aprenda, porque debemos indicar que en este campo existen diferentes tipos de neuronas que podamos usar como es adaline, bacpropagation, que lo mencionaremos en otro estudio.

En un próximo trabajo se estará mostrando las simulaciones realizadas con ejemplos prácticos y los tiempos que se demoran cada una de las neuronas para las compuertas OR, NOR y NAND, con las diferentes tipos que existen en el mundo de la inteligencia artificial y con una nueva metodología como la orientada a objetos.

## Referencias

[1] Piedrafita R (2000) Ingeniería de la Automatización, Alfaomega.

[2] Verdegay J.L. (2005). Una revisión de las metodologías que integran la Soft computing. Actas del Simposio sobre Lógica Fuzzy y Soft Computing LFSC 2005 (EUSFLAT).151-156.

[3] Perceptrón descrito en: http://www.lcc.uma.es/$\sim$munozp/documentos/modelos_computacionales/temas/Tema4MC-05.pdf.

[4] Lan D. (2005) Curso de Java – Soluciones y ejemplos para desarrollladores de Java, ANAYA O'REILLY

[5] Lenguajes de Prograamción C# detallado en: https://msdn.microsoft.com/es-es/library/aa287558(v=vs.71).aspx.

[6] Beneficios del Common Language Runtime en C# descrito en: https://docs.microsoft.com/es-es/dotnet/csharp/getting-started/introduction-to-the-csharp-language-and-the-net-framework.

[7] Lenguaje de Programación C++ descrito en: http://www.larevistainformatica.com/C++.htm.

[8] Ranking de los mejores Lenguajes de Programación en los años 2014 y 2015 presentado en el sitio: http://programacion.net/noticia/ranking_del_ieee_de_lenguajes_de_programacion_2015_2228.

[9] Interactive: The Top Programming Languages 2017, detallado en: http://spectrum.ieee.org/static/interactive-the-top-programming-languages-2015

[10] Lenguajes de Programación más populares y mejor pagados detallados en: https://devcode.la/blog/ranking-lenguajes-de-programacion/.

[11] Ranking de los Lenguajes de Programación según la empresa TIOBE, mostrado en el sitio: https://www.tiobe.com/tiobe-index//?6671423=1.

[12] Índice TIOBE, uso en el mundo de los distintos Lenguajes de Programación, detallado en el sitio: https://www.paradigmadigital.com/dev/el-indice-tiobe-como-medir-que-lenguajes-de-programacion-son-los-mas-usados/.

[13] Lenguajes de Programación mejor pagados en la industria de la ingeniería del software, obtenido del sitio: http://fundersandfounders.com/software-engineer-salary-2014/.

[14] Ranking de los 10 mejores Lenguajes de Programación descritos en el sitio: https://www.tiobe.com/tiobe-index//?6671423=1.