Programación en la computación cuántica. El futuro es hoy.

Programming in quantum computing. The future is today.

Villafuerte Quiroz Álvaro Luis
https://orcid.org/0000-0001-7346-0153
aviloluis@hotmail.com
Investigador Independiente

Resumen

El objetivo general de esta investigación es analizar los avances de la programación en la computación cuántica en la actualidad. La metodología se basó en un desarrollo bibliográfico con un tipo de investigación documental. Como conclusión, el análisis de la física cuántica es un área bastante importante para poder comprender los fenómenos del comportamiento de los objetos, pero cuando se decide cuantificar a través de ecuaciones matemáticas es que se concibe como mecánica cuántica. El proceso de transformación energética entre los sistemas involucra un proceso vibracional, cinético y potencial de los átomos y elementos, es por ello que esta ciencia es tomada como valor fundamental en la programación de sistemas electrónicos, debido a que puede entregar con mayor seguridad los datos y a su vez ofrecer confidencialidad, aunque todavía se encuentra en fase de estudio, pero su aplicabilidad será el boom tecnológico de la humanidad; este sistema se basa en qubits donde se toma en consideración la superposición, es decir que los bits pueden estar en diferentes momentos a la vez, por lo que su lenguaje es característico de la resolución de sistemas vectoriales unitarios y con una expresión gráfica a través de puertas lógicas características por cada investigador.

Palabras clave: física, mecánica cuántica, programación, computación cuántica.

Abstract

The general objective of this research is to analyze the advances in programming in quantum computing today. The methodology was based on a bibliographic development with a type of documentary research. In conclusion, the analysis of quantum physics is quite an important area to understand the phenomena of the behavior of objects, but when it is decided to quantify through mathematical equations, it is conceived as quantum mechanics. The energy transformation process between systems involves a vibrational, kinetic and potential process of the atoms and elements, which is why this science is taken as a fundamental value in the programming of electronic systems, because it can deliver the data with greater security and in turn offer confidentiality, although it is still in the study phase, but its applicability will be the technological boom of humanity; This system is based on qubits where superposition is taken into consideration, that is, the bits can be at different times at the same time, so its language is characteristic of the resolution of unitary vector systems and with a graphic expression through characteristic logic gates for each researcher.

Keywords**:** physics, quantum mechanics, programming, quantum computing.

Introducción

Un desarrollo revolucionario en la física clásica fue el descubrimiento de la física cuántica y la influencia que tiene el comportamiento de los átomos y sus partes en los fenómenos que rodean a la humanidad. busca resolver el porqué de las situaciones inentendibles gracias a la mecánica cuántica. Esto se puede observar a través de la Figura 1, donde se diferencia la física clásica con la cuántica. Dentro del átomo, existen los electrones los cuales tienen una energía cinética y vibracional, donde esta última tienen el comportamiento de una onda. Esta descripción es la que permite que cierta parte del electrón pueda atravesar la pared por medio de la refracción. Una de las manifestaciones de estos postulados tiene que ver con el desarrollo de la computación en especial la respuesta a través de la emisión de calor en el desarrollo de sus trabajos.

Este fenómeno característico de la computación fue el hincapié a buscar la respuesta a tal incertidumbre. Es aquí donde nace la vinculación de la mecánica cuántica con la computación. Y para ello se postula la teoría de la entropía como factor clave para dar respuesta. En este

sentido la mecánica cuántica postula diversa teoría importante. Introduce también otros conceptos revolucionarios difíciles de asimilar, tales como dualidad onda-partícula, principio de incertidumbre, superposición y entrelazamiento, que generaron en su momento profundas controversias, los que están generando hoy una nueva revolución en el campo de la computación e informática (Rossignoli, 2018).

En este particular, la computación cuántica utiliza estos postulados de la mecánica cuántica para poder comprender muchos fenómenos, pero a su vez generar más confianza y rapidez en el procesamiento de datos. La *computación cuántica* es una vertiente de la computación que hace uso de fenómenos de la mecánica cuántica tales como átomos, superposición, amplitudes de probabilidad o entrelazamiento cuántico (Kleinman Ruiz, 2020). Esto debido a la aparición de algoritmos que podrían suponer una mayor capacidad de cómputo respecto a los ordenadores convencionales e incluso que los superordenadores desarrollados en estos últimos años (Sanz Pérez, 2019).

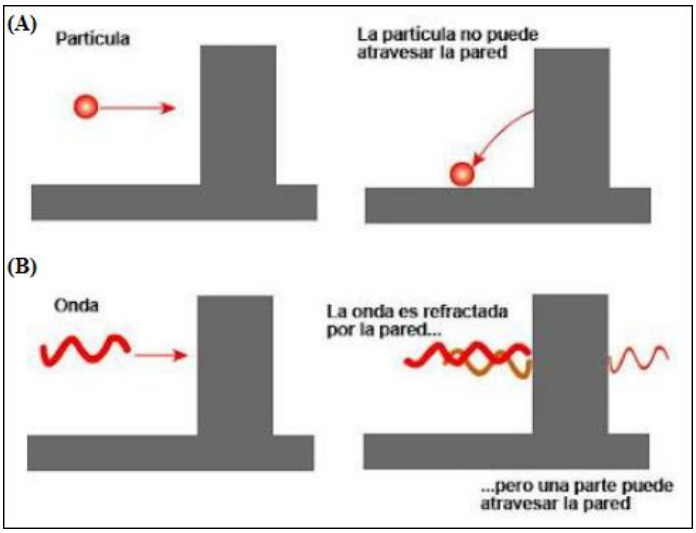

Figura 1. Efecto túnel. (A) Representa la situación descrita por la física clásica y (B) representa la situación que describe la física cuántica. Fuente: (Bonillo, 2013)

Este computador cuántico se basa fundamentalmente en la superposición de estados, aprovechando la virtud de los electrones u otros elementos que poseen dicha virtud, donde tienen una propiedad asociada llamada spin. Este se asocia con el movimiento de rotación de la partícula alrededor de un eje, la cual puede ser en un sentido, o el opuesto, por lo que se puede tomar cierto sentido 1 y el opuesto 0, es decir conllevando a llamarlos qubit (Rúa Vargas & Branch Bedoya, 2009). Esto permitió el desarrollo de tecnologías a partir de 1950. El

desarrollo de dispositivos como el transistor, el láser, el GPS o la resonancia magnética marcaron el inicio de lo que ahora conocemos como la *primera revolución cuántica* (Lierta & Kerstjens, 2020). En la Tabla 1 se detalla las aplicaciones y la Tabla 2 se muestra algunas innovaciones de la computación cuántica.

Tabla 1.
Aplicaciones de la computación cuántica.

Area, Función o Clasificación	Aplicaciones
La Criptografía	Aseguramiento los resultados electorales en Suiza. Mediante la criptografía vinculando la fibra óptica que conectaba el lugar donde se contaron las papeletas con el lugar donde estaban los ordenadores del proceso electoral.
Cuántica	La codificación de información confidencial (debe ser tan sólida que nadie pueda romperla, pero a su vez genera una barrera para cuando se deben analizar delitos informáticos, y se encuentra la información cifrada. Shor)
La Tele Transportación Cuántica	El traslado de la información cuántica desde un punto a hacia b. la tele transportación cuántica es posible por el entrelazamiento de dos fotones para transmitirle información del uno al otro.
Simulación de Sistemas Cuánticos	Un simulador de un sistema de vuelo.
	Simulación de los átomos de un fármaco para indicar cómo prepararlo, y cómo interactuará con otras sustancias químicas.
El Diseño de Nuevos Súper Conductores	Dirección de trenes de levitación magnética.
Búsqueda, Identificación y Detección	Identificación el mercurio en el pescado.
	Búsqueda del plomo en los juguetes.
	Detección de bombas.
Sensores	Realización de sensores más sólidos, más precisos, más sensibles, y si se pueden compactar se podrán usar ampliamente en el entorno medioambiental.

Fuente: (Reyes Alvarez, 2016)

Tabla 2.
Innovaciones de la computación cuántica.

ASPECTO/ELEMENTO	CARACTERÍSTICAS
Chip de silicio que genera sus propios fotones.	Un equipo de investigación internacional dirigido por la Universidad de Bristol, en Reino Unido, ha generado y manipulado por primera vez partículas individuales de luz -o fotones- en un chip de silicio. Un conjunto de científicos e ingenieros, dirigidos por el Dr. Mark Thompson construyeron un chip capaz de exponerse al ataque de un rayo láser.
La tecnología cuántica llega a la vida de las personas.	La computación cuántica se presenta como la gran promesa para seguir construyendo equipos más veloces. A diferencia del ordenador tradicional que se ejecuta con bits binarios, qubits cuánticos pueden ser 0 y 1 a la vez, lo que facilita un aumento importante en la velocidad de procesamiento. Fundamental para acelerar la búsqueda de bases de datos o el aprendizaje automático. Sin embargo, mientras los bits binarios se basan en transistores de silicio de confianza, los expertos aún deliberan sobre el mejor material para los equipos cuánticos.
IBM avanza hacia el primer computador cuántico	Investigadores de IBM han dado a conocer dos avances importantes hacia la realización de un ordenador cuántico de uso práctico. Han mostrado la capacidad de detectar y medir simultáneamente los dos tipos de errores cuánticos que se

	producirían en cualquier ordenador cuántico real. También han mostrado un nuevo diseño de circuito que – afirman – es la única arquitectura física que podría ser escalable a mayores dimensiones con éxito. Los ordenadores cuánticos prometen abrir nuevas posibilidades en el campo de la optimización y simulación que, simplemente, hoy en día no son alcanzables. Esto es por su capacidad de computación. Por ejemplo, un ordenador cuántico de solo 50 bits cuánticos (qubits) superaría a cualquier combinación de supercomputadoras TOP500 actual en capacidad computacional.
Internet cuántica	Las computadoras cuánticas podrían comunicarse en principio entre si mediante el intercambio de fotones individuales para crear una internet cuántica. Un equipo de científicos de la Universidad Autónoma de Barcelona ha desarrollado un material guía y transporta el campo magnético de forma parecida a una fibra óptica lo hace con la luz o una manguera con el agua. El prototipo de fibra magnética mide 14 cm, pero puede implementarse a cualquier escala, incluida la nanométrica. Innovación que acerca más a la aplicación en el mundo de la internet cuántica.

Fuente: (Reyes Alvarez, 2016)

En este sentido, los protagonistas de la computación cuántica son los qubits, así como son los bits en la computación clásica. Estos pueden estar no sólo en dos estados dados, digamos 0 y 1, sino también en *cualquier superposición* de ellos, de acuerdo a uno de los principios básicos de la mecánica cuántica (Rossignoli, 2018).

> Con los bits convencionales si teníamos un registro de tres bits había ocho valores posibles, y el registro sólo podía tomar uno de esos valores. En cambio, si tenemos un vector de tres qubits, la partícula puede tomar ocho valores distintos a la vez gracias a la superposición cuántica. Así, un vector de tres qubits permitiría un total de ocho operaciones paralelas. Como cabe esperar, el número de operaciones es exponencial con respecto al número de qubits. (Bonillo, 2013, pág. 11)

Esta situación hace que la computadora cuántica sea superior a la computadora tradicional, por el efecto que tiene el qubits de superposición, donde los bits están representados por 2^n de los bits tradicionales. con esta función permite obtener respuestas mucho más rápidas. Esto hace los procesos matemáticos más rápidos, lo cual es la base de los algoritmos actuales, que son una estructura lógica podrían hacer que romper un contenido cifrado con ciertas características de seguridad sea cuestión de meses y años (Reyes Alvarez, 2016). Es por ello, que, por ejemplo, en la computadora cuántica 8 qubits podrían asumir todas las combinaciones de estados de 8 bits y procesar todo en un solo ciclo de computación (Díaz, 2009).

Por lo cual, el qubit puede comportarse como un vector unitario de combinación lineal de dos estados básicos. Estos son conocidos como *estado fundamental* o *base* –$|0\rangle$– y *estado excitado* –$|1\rangle$– los cuales son analogías del 0 y 1 del bit clásico (Kleinman Ruiz, 2020). En este particular, se conoce que el nivel secundario de energía como el spin juega un papel importante, entonces el sistema de una partícula de spin ½ es:

> $|\Psi\rangle = a|\uparrow\rangle + b|\downarrow\rangle$ Donde $|\uparrow\rangle$ y $|\downarrow\rangle$ son los estados con $Sz = \pm\hbar/2$. Las amplitudes complejas a y b obedecen la condición de normalización $|a|2 + |b|2 = 1$. Los estados de Sz son usados para representar 0 y 1: $|0\rangle = |\uparrow\rangle$; $|1\rangle = |\downarrow\rangle$. (Dorado & Lander, 2018, pág. 3)

Del mismo modo, según Dirac, P. (1939), citado por Magaz Romero (2020) estos estados se pueden representar como:

$$|0\rangle = \begin{bmatrix}1\\0\end{bmatrix} \qquad |1\rangle = \begin{bmatrix}0\\1\end{bmatrix}$$

Para hacer cálculos más complejos se requiere que existan más de un qubits, por lo tanto, se tiene un registro cuántico:

$$|00\rangle = \begin{bmatrix}1\\0\\0\\0\end{bmatrix} \quad |01\rangle = \begin{bmatrix}0\\1\\0\\0\end{bmatrix} \quad |10\rangle = \begin{bmatrix}0\\0\\1\\0\end{bmatrix} \quad |11\rangle = \begin{bmatrix}0\\0\\0\\1\end{bmatrix}$$

Esto ultimo reflejando en la secuencia de spin sería:

$$|00\rangle = |\uparrow\rangle\,|\uparrow\rangle;\ |01\rangle = |\uparrow\rangle|\downarrow\rangle;\ |10\rangle = |\downarrow\rangle|\uparrow\rangle;\ |11\rangle = |\downarrow\rangle|\downarrow\rangle$$

Una explicación más precisa del comportamiento del qubit es a través de la esfera de Bloch, tal cual se muestra en la Figura 2. De este modo, un qubit puede ser perfectamente representado geométricamente por esta esfera, esto debido a el grado de libertad sobre la asignación de estados sobre la superficie (Dorado & Lander, 2018).

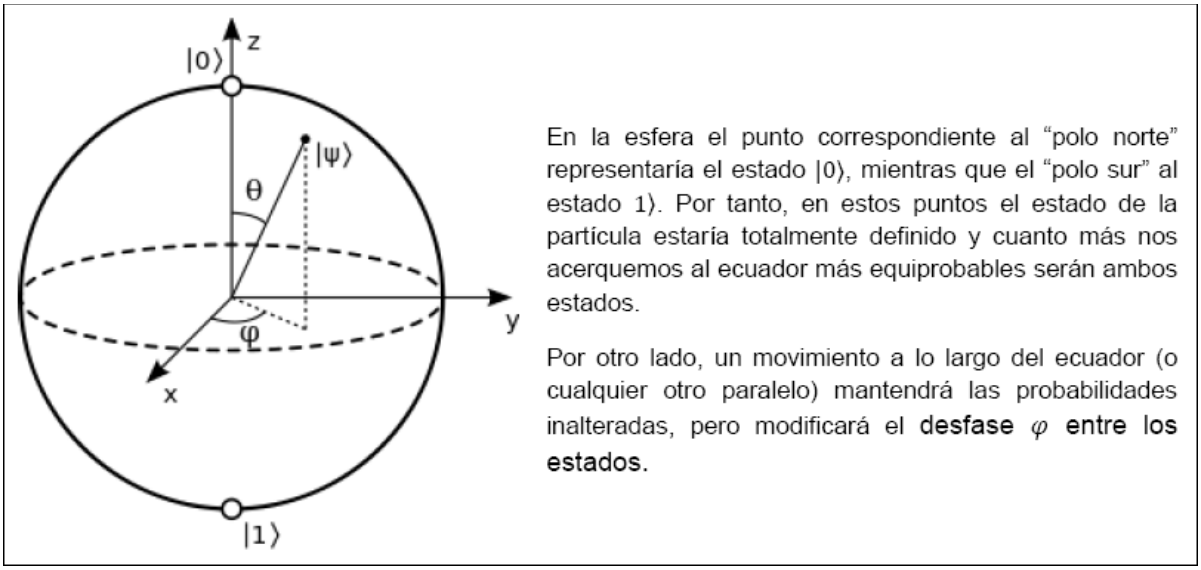

Figura 2. Esfera de Bloch. Fuente: (Dorado & Lander, 2018)

Otro elemento importante dentro del desarrollo de la computación cuántica son las puertas cuánticas. Estas e ven reflejadas en la computación tradicional a través de un circuito, tal como se detalla en la Figura 3. Estas puertas lógicas con nombres como AND, OR, y NOT, conectadas por cables que llevan los estados de los bits clásicos de la salida de una de las puertas a la entrada de la siguiente (Dorado & Lander, 2018). La puerta cuántica se expresa por medio de isomorfos lineales unitarios. Es decir, que $UU^* = 1$, donde U^* es la matriz conjugada traspuesta de U, por lo que una consecuencia inmediata de esta igualdad es que cualquier puerta cuántica es reversible; además, el número de qubits de entrada coincidirá con el de salida (Aguirre Galindo & Pellejero Ortega, 2020). En la Figura 4, se muestra las puertas cuánticas donde recogen las señales desde la izquierda y devuelven respuestas hacia la derecha.

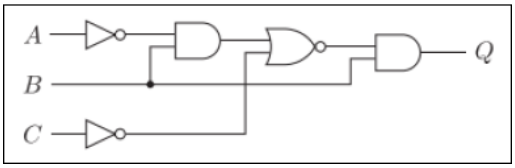

Figura 3. Puertas lógicas en la computación tradicional. Fuente: (Dorado & Lander, 2018)

Para todo el desarrollo del circuito electrónico de la computación cuántica se debe cumplir lo siguiente:

> (1) Para que dichos sistemas sean válidos, inicialmente los qubits deben estar totalmente controlados y estables el tiempo suficientemente para que puedan ser leídos, escritos y, por tanto, manipulados. (2) En segundo lugar, los sistemas deben ser capaces de ejecutar cualquier tipo de algoritmos cuánticos. (3) La tercera etapa consiste en corregir los errores mediante la lectura de los síndromes de error. (Ruiz, Fuster, & Ramírez, pág. 16)

En consecuencia, tras la creación de las puertas lógicas para la computación cuántica queda por descubrir cómo es la programación en este tipo de sistema. Es factible la realización de un lenguaje de programación que permita desarrollar los comandos en el tiempo que según estiman, será el más rápido de todos los ordenadores.

Por lo tanto, el objetivo general de esta investigación es analizar los avances de la programación en la computación cuántica en la actualidad. La metodología se basó en un desarrollo bibliográfico con un tipo de investigación documental.

Metodología

La investigación se basó en un diseño bibliográfico de tipo documental. El diseño se fundamenta en la revisión sistemática, rigurosa y profunda de material documental de cualquier clase, donde se efectúa un proceso de abstracción científica, generalizando sobre la base de lo fundamental, partiendo de forma ordenada y con objetivos precisos (Palella Stracuzzi & Martins Pestana, 2010). Para lograr este propósito se utilizó herramientas como textos, documentos y artículos científicos publicados disponibles en la web.

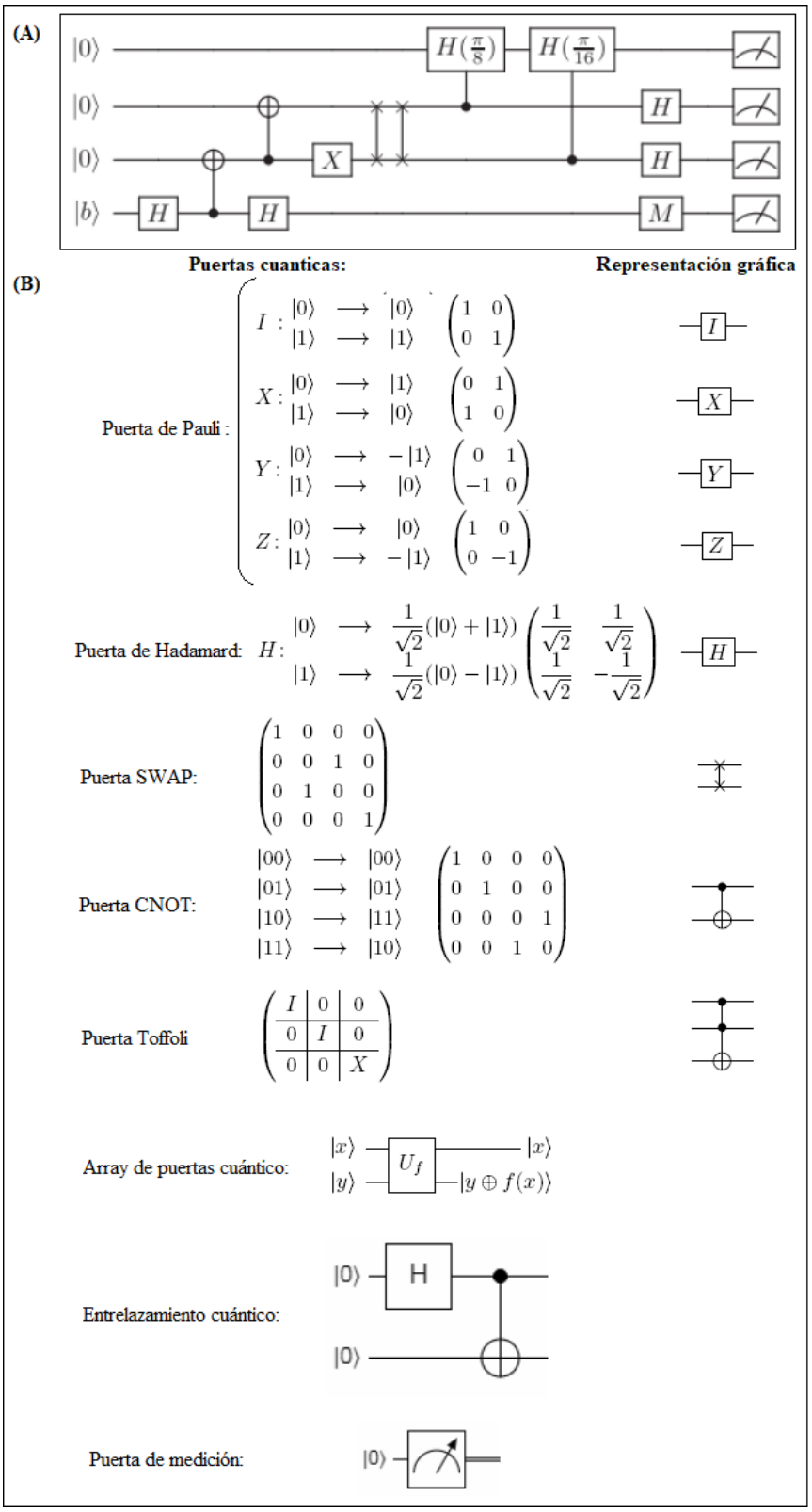

Figura 4. (A) Representación de las puertas lógicas en la computación cuántica y (B) la descripción de las puertas lógicas y su representación gráfica. Fuente: (Aguirre Galindo & Pellejero Ortega, 2020; Dorado & Lander, 2018; Sanz Pérez, 2019)

Resultados y discusión

Características de la programación.

Una de las características del mundo actuales es el dominio que tienen las computadoras y sus aplicaciones. Los computadores digitales, los programas informáticos y las redes constituyen los elementos técnicos fundamentales de la industria moderna (Mazaeda Echevarría, de la Fuente López, González Sánchez, & Moya de la Torre, 2016). Estas se manifiestan a través de redes de comunicación utilizando como vehículo, por ejemplo, el internet. Estas se desarrollan gracias aun conjuntos de programas que permiten ejecutar acciones. Estos programas son la clave y el lenguaje que permite comunicar las acciones que se desean por hechos tangibles en el computador.

El objetivo esencial de la programación es la utilización de las computadoras con el fin de resolver problemas, a través de los recursos que brindan los diferentes lenguajes de programación, para ello el individuo debe manipular los softwares relacionados con la disciplina (Díaz Tejera, Fierro Martín, & Muñoz Pentón, 2018). Es por ello, que se debe ejecutar primer un ordenamiento antes de desarrollar un programa. Se debe ejecutar una idea a través de la realización de un algoritmo, para luego plasmarlo en un lenguaje de programación y así ejecutar el programa en el ordenador.

En este caso, el protagonismo de la programación lo lleva el desarrollo del algoritmo. Según Levy Kortright (1994) una estrategia valedera es comenzar a enseñar programación utilizando los algoritmos como recursos esquemáticos para plasmar el modelo de la resolución de un problema (Moroni & Señas, 2005). Por lo cual, este constituye una lista bien definida, ordenada y finita de operaciones, partiendo de un *estado inicial* y *una entrada*, es a través de *pasos sucesivos* y bien definidos que se llega a un *estado final*, en el que se obtiene una solución (Juganaru Mathieu, 2015).

En este sentido, el algoritmo permite el desarrollo del problema que se tenga propuesto. En la Figura 5 se observa el esquema de la resolución del problema. Los pasos de la resolución de un problema se detallan a continuación:

1. *Diseño del algoritmo,* que describe la secuencia ordenada de pasos —sin ambigüedades— que conducen a la solución de un problema dado. (*Análisis del problema y desarrollo del algoritmo*.)
2. Expresar el algoritmo como un *programa* en un lenguaje de programación adecuado. (*Fase de codificación*.)

> 3. *Ejecución y validación* del programa por la computadora. Para llegar a la realización de un programa es necesario el diseño previo de un algoritmo, de modo que sin algoritmo no puede existir un programa. (FUNDAMENTOS DE PROGRAMACIÓN, 2006, pág. 18)

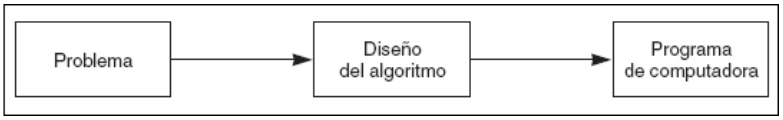

Figura 5. Esquema para la resolución de un problema. Fuente: (FUNDAMENTOS DE PROGRAMACIÓN, 2006)

Una parte importante de los algoritmos es que son independientes del lenguaje de programación. Es decir, en cada problema el algoritmo se puede expresar en un lenguaje diferente de programación y ejecutarse en una computadora distinta; sin embargo, el algoritmo será siempre el mismo (FUNDAMENTOS DE PROGRAMACIÓN, 2006). Un algoritmo puede ser expresado en: Lenguaje natural (a veces, este no resulta muy claro, pero es muy útil para problemas simples); Pseudocódigo; Diagramas de flujo, y; Programas (Juganaru Mathieu, 2015). Por lo cual, las características fundamentales que debe cumplir un algoritmo son:

> • Un algoritmo debe ser *preciso* e indicar el orden de realización de cada paso.
> • Un algoritmo debe estar *definido*. Si se sigue un algoritmo dos veces, se debe obtener el mismo resultado cada vez.
> • Un algoritmo debe ser *finito*. Si se sigue un algoritmo, se debe terminar en algún momento; o sea, debe tener un número finito de pasos. (FUNDAMENTOS DE PROGRAMACIÓN, 2006, pág. 18)

Después, del desarrollo de la secuencia de pasos para conseguir la solución al problema, queda el paso de programas, por lo que es necesario plantear un lenguaje de programación que manifieste el desarrollo del algoritmo. Los lenguajes poseen una estructura (gramática o sintaxis) y un significado (semántica), por lo que los lenguajes de computadoras tienen menos combinaciones aceptables que los lenguajes naturales, sin embargo, estas combinaciones deben ser utilizadas correctamente (FCASUA, 2006). Además, un lenguaje de programación es un caso particular del lenguaje informático; este último permite hacer programas, pero también describir datos, configuraciones físicas y protocolos de comunicación entre equipos y programas (Juganaru Mathieu, 2015).

De la misma manera, existen dos tipos de lenguaje, el procedimental y el relacional. La diferencia es que los primeros exigen tanto lo que se quiere hacer como la forma, mientras que los segundos sólo exigen que se diga lo que se quiere hacer, pero no es necesario que se exprese

el camino para realizarlo (FCASUA, 2006). En resumen, un programa en un lenguaje procedimental es un conjunto de instrucciones o sentencias, donde el programador sólo tiene que crear esta lista de instrucciones en un lenguaje de programación, compilar en la computadora y ésta, a su vez, ejecuta estas instrucciones (FUNDAMENTOS DE PROGRAMACIÓN, 2006). En la Tabla 3 se puede detallar las condiciones para el lenguaje de programación.

Tabla 3.

Condiciones para el lenguaje de programación.

El conocimiento del lenguaje en cuestión; es decir, su sintaxis y la semántica de los conceptos y las instrucciones que lo componen.
El tipo de problema a resolver. Por ejemplo, para consultar datos que se guardan en un formato específico en una base de datos o en una base de conocimiento se utilizan, comúnmente, los lenguajes de tipo declarativo, donde se caracterizan los datos que se esperan en salida, como SQL para la base de datos relacional, PROLOG para la base de conocimiento, XQuery y XSLT para colecciones de datos en el formato XML. En otro ejemplo, para dar las órdenes de instalación de software, es conveniente escribir programas en el *shell* del sistema operativo.
El derecho y la posibilidad material de utilizar un compilador o intérprete de dicho lenguaje, ya que estos tipos de software (compilador, taller de desarrollo, intérprete) suelen tener un costo monetario o licencias restrictivas.
La configuración física que está disponible. Por ejemplo, si está disponible una arquitectura multiprocesador, sería más conveniente utilizar un lenguaje de tipo C o FORTRAN, por medio de los cuales se abstendría de realizarse el cálculo paralelo, o emplear herramientas de paralelización automática. En el caso de que el programa tuviera que explorar y comunicar con una interfaz de un equipo raro, como una máquina de producción o un dispositivo de medición, es preferible escribirlo en un código del lenguaje ensamblador.
La configuración del software que está disponible o que se impone por la construcción del programa y el uso ulterior del producto fi nito. Por ejemplo, para aprender la programación es mejor iniciar con un lenguaje de alto nivel del paradigma imperativo de tipo C o PASCAL. En el caso de que el destinatario del programa utilizara el sistema operativo de plataforma móvil con sistema MAC OS, las herramientas para desarrollar aplicaciones imponen usar el *framework* COCOA o XCode y el lenguaje de programación Objective C.

Fuente: (Juganaru Mathieu, 2015)

También, es necesario conocer las razones por las cuales es necesario estudiar los lenguajes de programación, las cuales se muestran en la Tabla 4.

Tipos de programación.

Se conoce que la programación está basado en 4 aspectos, estos son los lenguajes de programación, programadores, codificación y código fuente. Estos dos últimos significan que el proceso de traducir un algoritmo en *pseudocódigo* a un lenguaje de programación se denomina codificación, y el algoritmo escrito en un lenguaje de programación se denomina código fuente (FUNDAMENTOS DE PROGRAMACIÓN, 2006). Ahora, las computadoras

no entenderán el código fuente sino a través de un código máquina. Esto se puede visualizar a través de la Figura 6.

Tabla 4.

Razones por las cuales estudiar los lenguajes de programación.

ASPECTO/ELEMENTO	CARACTERÍSTICAS
Mejorar la habilidad para desarrollar algoritmos eficaces	Muchos lenguajes tienen ciertas características que, usadas adecuadamente, benefician al programador, pero cuando se usan en forma inadecuada pueden desperdiciar grandes cantidades de tiempo de computadora o de conducir al programador a errores lógicos que hacen perder mucho tiempo, además el costo de la reclusión varia según la implementación del lenguaje.
Mejorar el uso del lenguaje de programación disponible	A través de entendimiento de cómo se implementan las características del lenguaje que uno usa, se mejora grandemente la habilidad para escribir programas más eficientes.
Enriquece el vocabulario de construcciones útiles de programación	Con frecuencia se nota que los lenguajes sirven para una ayuda como para pensar como para construir, los lenguajes sirven también para estructurar lo que se piensa, hasta el punto que es difícil pensar en alguna forma que no permita la expresión directa de las palabras. El entendimiento de las técnicas de implementación es particularmente, porque para emplear un constructor mientras un programa en un lenguaje que no proporciona directamente el programador debe dar una propia implementación del nuevo constructor en términos de los elementos primitivos ofrecidos realmente por el lenguaje.
Permite una mejor selección de lenguaje de programación	Cuando la situación lo amerita, el conocimiento de una variedad de lenguajes permite la selección de lenguaje correcto para un proyecto particular, por tanto, reduce enormemente el esfuerzo de codificación requerido.
Hace más fácil el aprendizaje de un nuevo lenguaje	Un lingüista, a través de un conocimiento de las estructuras en que se basan los lenguajes naturaleza, puede aprender un lenguaje extranjero más rápido y fácil que el esforzado principalmente que entiende poco de su estructura lengua natal.
Facilita el diseño de un nuevo lenguaje	Pocos programadores piensan en sí mismos como diseñadores; es más ningún programa tienen interfaces del usuario que es, en realidad, una forma de lenguaje. La interfase del usuario consiste en unos formatos y comandos que son proporcionado por el para comunicarse con el programa. El diseñador de la interfase de usuario de un programa tal como un editor de textos, un sistema operativo o un paquete de gráficas debe estar familiarizado con mucho de los resultados que están presentes en el diseño de un lenguaje de programación de propósitos generales.

Fuente: (FCASUA, 2006)

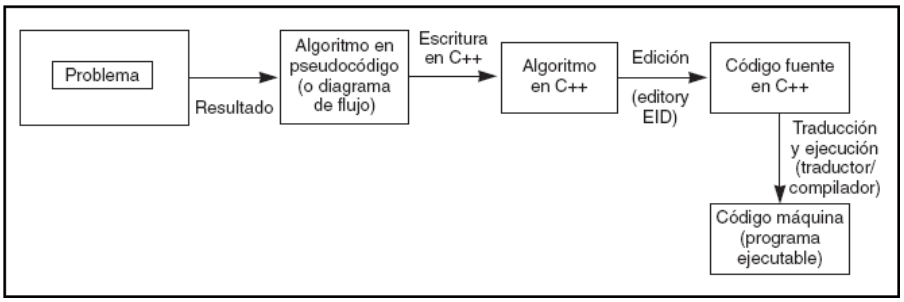

Figura 6. Proceso de transformación de un algoritmo en pseudocódigo en un programa ejecutable. Fuente: (FUNDAMENTOS DE PROGRAMACIÓN, 2006)

Tipos de programación.

Se conoce que la programación esta basado en 4 aspectos, estos son los lenguajes de programación, programadores, codificación y código fuente. Estos dos últimos significan que el proceso de traducir un algoritmo en *pseudocódigo* a un lenguaje de programación se denomina codificación, y el algoritmo escrito en un lenguaje de programación se denomina código fuente (FUNDAMENTOS DE PROGRAMACIÓN, 2006). Ahora, las computadoras no entenderán el código fuente sino a través de un código máquina. Esto se puede visualizar a través de la Figura 6.

A continuación, se detallan algunos códigos fuente más utilizados en la programación de ordenadores tradicionales.

Pascal.

Uno de los principales código fuente que se desarrollan a nivel de las universidades debido a que sirve como base fundamental en el desarrollo cognitivo del individuo y como herramienta para familiarizarse con los términos de programación. Sus principales objetivos era ser eficiente para llevar a cabo y correrse los programas, permite bien el desarrollo de estructuras y también organizar programas, debido a que es un lenguaje estructurado en bloques, donde contiene adentro las definiciones del subprograma usado (FCASUA, 2006).

Lenguaje C.

Uno de los lenguajes más importante es el lenguaje en C basado en alto nivel. La popularidad, eficacia y potencia de C, se ha producido porque este lenguaje no está prácticamente asociado a ningún sistema operativo, ni a ninguna máquina, en especial; por la cual C, es conocido como el lenguaje de programación de sistemas, por excelencia (FUNDAMENTOS DE PROGRAMACIÓN, 2006). Este lenguaje tiene gran poderío en sus operaciones a nivel de bits (propias de ensambladores), por lo que resulta ser el lenguaje preferido para el desarrollo de software de sistemas y aplicaciones profesionales de la programación de computadoras (FCASUA, 2006).

Lenguaje C++.

Este tipo de lenguaje es derivado del lenguaje C debido a que solventa las debilidades de este en la programación moderna. Uno de las dificultades es que C requiere un nivel de sofisticación a sus usuarios que les obliga a un difícil aprendizaje a los programadores

principiantes ya que es de comprensión difícil (FUNDAMENTOS DE PROGRAMACIÓN, 2006). Este lenguaje C++ surgió como una extensión del conocido lenguaje C, capaz de implementar los conceptos de la programación Orientada a Objetos (OO) (Mazaeda Echevarría, de la Fuente López, González Sánchez, & Moya de la Torre, 2016). Una de las críticas más fuertes es la falta de un sistema de tipos seguro, donde la asignación de memoria dinámica explícita, junto con la falta de colección de basura, incrementan los riesgos de errores, tales como referencias nulas o basura (Díaz J. , 2006).

Python.

Otros de los que nace de las debilidades del lenguaje C++ es el Python a finales de los 80 y comienzos de los 90. Cuenta con facilidades para la programación orientada a objetos, imperativa y funcional, por lo que se considera un lenguaje multiparadigmas (Challenger-Pérez, Díaz-Ricardo, & Becerra-García, 2014).

Una de las características fundamentales es que esta tiene una propia librería donde el lenguaje se basa en su sencillez a diferencias de los grandes códigos fuente. Contiene implícitas algunas estructuras de datos que permiten realizar algunas tareas complejas en pocas líneas de código y de manera legible; en otras palabras, la filosofía es que mientras más sencilla y clara se mantengan e implementen las ideas, mejores serán estas (Challenger-Pérez, Díaz-Ricardo, & Becerra-García, 2014).

Java.

Otro factor importante en el desarrollo de las características del lenguaje C y C++ es código fuente tipo Java. Java es un lenguaje de programación totalmente orientado a objetos como respuesta a los problemas de programación de C++ (Roman Arenaza, 2019).

Este lenguaje puede estar soportado por cualquier sistema operativo como Windows, solaris, Linux, etc. Pueden mencionarse muchas características de Java, sin embargo, las más significativas para la implementación del mismo: Orientado a Objetos, Robusto, Independiente de plataforma, Multitarea, además soporta las tres características del paradigma de la programación orientada a objetos: encapsulación, herencia y polimorfismo (Roman Arenaza, 2019).

Script.

Al nacer la era de la internet, los programadores no quedaron de brazos cruzados, tanto así que buscaron crear lenguajes que se desarrollarían en la web.

> Un lenguaje de script (o lenguaje de guiones) es similar a un lenguaje de macros o a un fichero por lotes (batch): una lista de comandos que se pueden ejecutar sin o con la participación del usuario. Se trata en definitiva de un lenguaje de programación, que suele emplearse dentro de un contexto (dentro de una aplicación) y que no permite programar aplicaciones independientes (no permite crear _cheros ejecutables independientes). Los lenguajes de script suelen ser interpretados, aunque algunos pueden poseer una fase de pseudocompilación con el fin de optimizar su ejecución.
> Existen multitud de lenguajes de script que se pueden emplear en páginas web: JavaScript, VBScript, Perl, Rexx, etc. Sin embargo, algunos sólo se pueden emplear en navegadores muy concretos y poco extendidos. Los lenguajes de script más empleados en Internet son JavaScript y en menor medida VBScript. (Luján-Mora, 2002, pág. 175)

Programación cuántica.

Antes de realizar la programación en la computación cuántica es necesario poder establecer el ordenador que recibirá estos códigos fuentes. En la Figura 7 se puede ver la representación estructura del ordenador. Aunque, el desarrollo de la computadora cuántica todavía sigue en proceso por lo que han existido diversas formas para programar. Existen tres tipos de tecnologías que se han desarrollado, los ordenadores cuánticos analógicos, ordenadores universales, y modelos de ordenamiento cuánticos que comercializa la empresa D-Wave (Gómez, Gómez, Gómez, & Villabona, 2019). Pero, se debe tomar en cuanta que para poder realizar una programación es necesario aplicar el algoritmo que incluya los pasos que serán ejecutados por el código fuente.

El algoritmo en la computación cuántica es un mecanismo que manipula los qubits a través de las puertas lógicas. Este algoritmo consiste en transformar un estado inicial Ψ_1 en su correspondiente estado final Ψ_2, con un U sobre el espacio de Hilbert, por lo que la función queda $U\Psi_1=\Psi_2$, donde U sigue los postulados de la mecánica cuántica la cual es lineal (Bonillo, 2013).

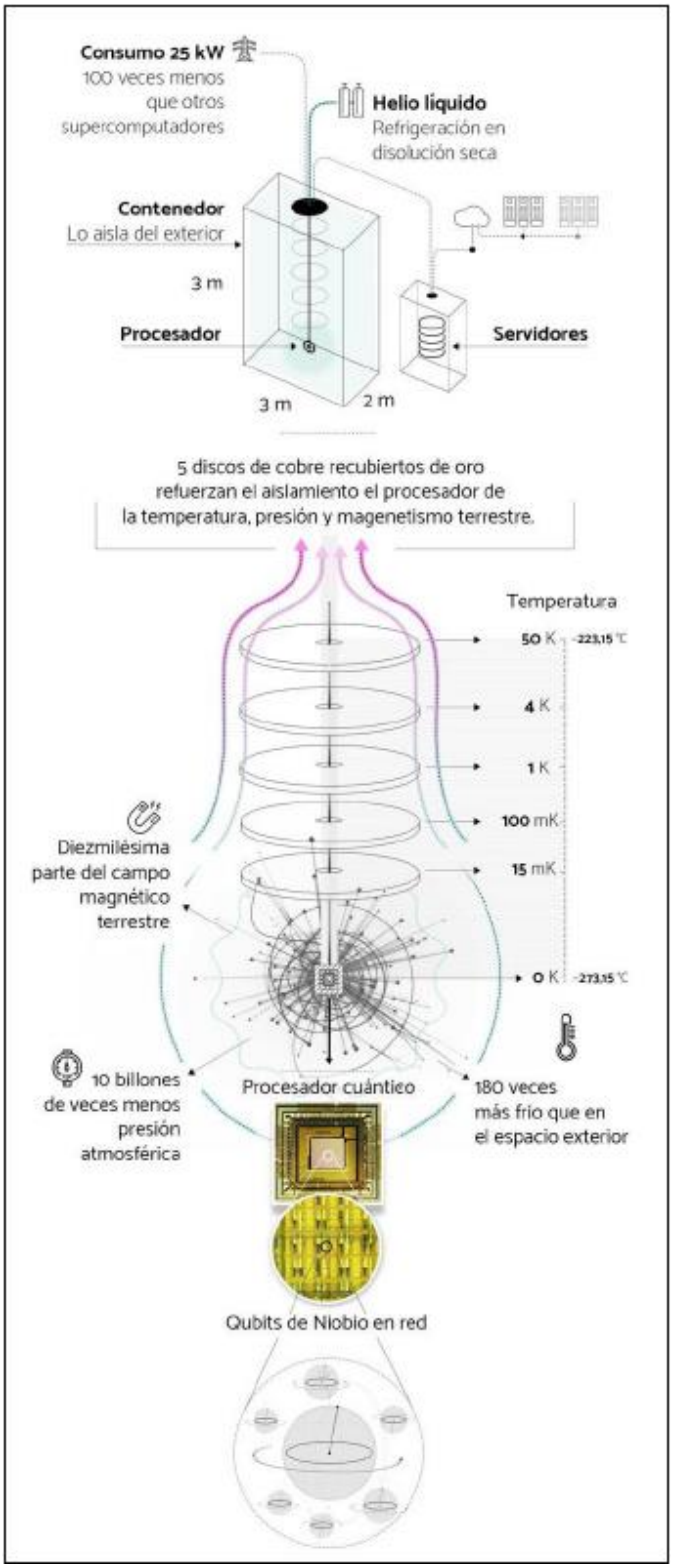

Figura 7. Representación estructural del ordenador cuántico. Fuente: (Ariza, Aguilar, & Mogollon, 2017)

En la definición de algoritmo cuántico se han incluido dos restricciones. La primera afecta al estado inicial, que siempre será el mismo: □1 = |0□. La segunda consiste en que las puertas y las medidas cuánticas no se pueden alternar. En primer lugar, se aplica una secuencia de puertas cuánticas, y a continuación una secuencia de medidas

> cuánticas. Estas restricciones simplifican los algoritmos cuánticos y no afectan al modelo de computación, pues todos los algoritmos cuánticos se pueden convertir en algoritmos cuánticos equivalentes que respetan estas restricciones. (Bonillo, 2013, pág. 74)

Uno de los algoritmos clásico en la computación cuántica es el algoritmo de Grover. Este es un algoritmo de búsqueda en una base de datos desordenadas (Portugal & Marquezino, 2019). Para ello se formula primero el problema:

> Sea N una potencia de 2, es decir, $N = 2^n$ para algún número entero *n*. Supongamos que *f*: {0,…, N-1*+}→{0, 1} es una función booleana tal que *f*(x) = 1, sí y solo sí x = x_0 para algún valor fijo x_0. Entonces
>
> $$f(x) = \begin{cases} 1, & si\ x = x_0 \\ 0, & \text{caso contrario} \end{cases}$$
>
> Se supone que x_0 no se conoce, por lo que se debe desarrollar un algoritmo que evalúe *f* el número mínimo de veces. (Portugal & Marquezino, 2019, pág. 29)

Para resolver este algoritmo se utiliza la puerta de Toffoli, generalizada por x_0. Esta se denominará Fx_0, que se define como $Fx_0|x\rangle\,|i\rangle = |x\rangle\,|i \oplus f(x)\rangle$, donde *x* es una cadena de n bits y *i* es un bit.

> El conjunto de los *n* primeros qubits se denomina primer registrador y el último qubit segundo registrador. Se tiene en cuenta que si se toma *i*=0, la ecuación anterior se reduce a
>
> $$Fx_0|x\rangle\,|0\rangle = \begin{cases} |x\rangle|1\rangle, & si\ x = x_0 \\ |x\rangle|0\rangle, & caso\ contrario \end{cases}$$
>
> El resultado del cálculo de *f*(x) se almacena en el segundo registrador, mientras que el valor del primer registrador no se modifica. (Portugal & Marquezino, 2019, pág. 30)

De forma general, la representación del algoritmo de Grover se puede representar de manera gráfica tal como se detalla en la Figura 7.

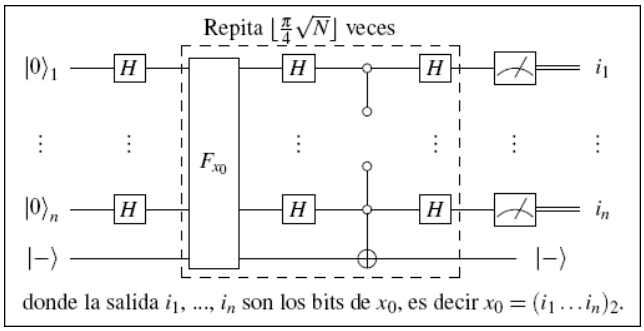

Figura 7. Representación gráfica general del algoritmo de Grover. Fuente: (Portugal & Marquezino, 2019)

También, existe otro algoritmo que puede determinar una programación cuántica, el cual es el Deutsch-Jozsa. En la Figura 8 se puede observar el circuito de Deutsch-Jozsa para n qubits.

Este algoritmo evalúa si una función *f*: *f*(*xn*...*x*1, *x*0), cuya entrada son ceros y unos, y su salida es cero o uno, tiene la propiedad de ser constante o balanceada. Se ha asegurado previamente que esta función *f*, tiene esta propiedad. Una función es constante si devuelve siempre cero o uno para cualquier entrada. Por el contrario, es balanceada si sus salidas son la mitad uno y la otra mitad cero. (Chiñas-Fuentes & Hernández-Mota, 2018, págs. 31-32)

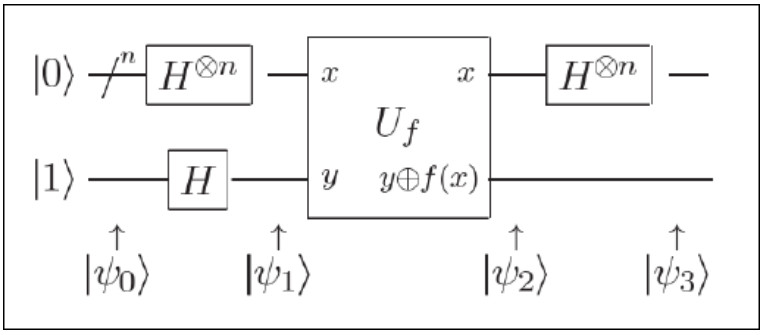

Figura 8. Representación gráfica general del algoritmo de Deutsch-Jozsa. Fuente: (Chiñas-Fuentes & Hernández-Mota, 2018)

Referencias bibliográficas

Aguirre Galindo, L., & Pellejero Ortega, J. (2020). *Computación cuántica: pruebas de mutación.* Trabajo fin de grado para optar al título en Matemática e Ingeniería Informática de la Universidad Complutense de Madrid. Obtenido de https://eprints.ucm.es/61617/1/AGUIRRE_GALINDO_Computacion_Cuantica_Pruebas_de_Mutacion_4398577_174863%20%281%29.pdf

Ariza, H. F., Aguilar, N. H., & Mogollon, K. V. (2017). *ORDENADOR CUANTICO ARQUITECTURA E INFORMACION.* Ingeniería de Sistemas, Universidad Industrial de Santander. Obtenido de http://wiki.sc3.uis.edu.co/images/5/54/ArtiG3.pdf

Bonillo, V. M. (2013). *Principios fundamentales de computación cuántica.* Texto de apoyo, Departamento de Computación de la Universidad de La Coruña. Obtenido de http://www.academia.edu/download/59425432/Principos_de_computacion_cuantica20190528-121332-14ziwch.pdf

Challenger-Pérez, I., Díaz-Ricardo, Y., & Becerra-García, R. A. (2014). El lenguaje de programación Python. *Ciencias Holguín, 20(2)*, 1-13.

Chiñas-Fuentes, K., & Hernández-Mota, D. (2018). *Algoritmos cuánticos implementados en una computadora de diamante C12-C13.* madrid, España: Trabajo de grado para optar al título de Ingeniero en Informática de la Universidad Complutense de Madrid. Obtenido de https://eprints.ucm.es/61918/1/SANCHEZ_VELEZ_Diseno_de_algoritmos_y_circuitos _cuanticos_4398577_287942181.pdf

Díaz Tejera, K. I., Fierro Martín, E., & Muñoz Pentón, M. A. (2018). La enseñanza de la programación: una experiencia en la formación de profesores de informática. *Educación, 27(53)*, 73-91. Obtenido de http://www.scielo.org.pe/pdf/educ/v27n53/a05v27n53.pdf

Díaz, J. (2006). Enseñando programación con C++: una propuesta didáctica. *Revista de Informática Educativa y Medios Audiovisuales, 3(7)*, 12-21. Obtenido de http://laboratorios.fi.uba.ar/lie/Revista/Articulos/030307/A2Jun2006.pdf

Díaz, L. M. (2009). *COMPUTACIÓN CUÁNTICA.* Revista Electrónica No. 12, Facultad de Ingeniería - Universidad Rafael Landívar. Obtenido de https://www.researchgate.net/profile/Leonel_Morales/publication/229025363_COMPUTACION_CUANTICA/links/0fcfd50fd65c93af6b000000.pdf

Dorado, V., & Lander, J. (2018). *Simulación de algunos algoritmos de computación cuántica.* Trabajo de fin de grado para optar al título de Físico de la Universidad del País Vasco. Obtenido de https://addi.ehu.es/bitstream/handle/10810/26458/TFG_Vallejo_Dorado_Jon_Lander.pdf?sequence=1&isAllowed=y

FCASUA. (2006). *Lenguajes de programación.* Recuperado el 08 de Octubre de 2020, de Unidad I: http://fcasua.contad.unam.mx/apuntes/interiores/docs/98/4/informatica_4.pdf

FUNDAMENTOS DE PROGRAMACIÓN. (2006). *Introducción a la ciencia de la computación y a la programación.* Recuperado el 08 de Octubre de 2020, de https://www.mheducation.es/bcv/guide/capitulo/844814645X.pdf

Gómez, N., Gómez, A., Gómez, A., & Villabona, D. (2019). *Arquitectura de la computación cuántica: La información es transmitida más rápido que la velocidad de la luz.* Bucaramanga, Colombia: Facultad de Fisicomecánicas, Ingeniería de Sistemas, Universidad industrial de Santander. Obtenido de http://wiki.sc3.uis.edu.co/images/8/8f/TF4.pdf

Juganaru Mathieu, M. (25 de Mayo de 2015). *Introducción a la programación.* Recuperado el 8 de Octubre de 2020, de Grupo Editorial Patria, S.A. de C.V.: https://editorialpatria.com.mx/pdffiles/9786074384154.pdf

Kleinman Ruiz, I. E. (2020). *Computación cuántica: Aplicaciones prácticas que la computación clásica no puede solucionar.* Madrid, España: Doble Grado en Ingeniería informática y Administración de empresas de la Universidad Carlos III de Madrid. Obtenido de https://e-archivo.uc3m.es/bitstream/handle/10016/29750/TFG_Ignacio_Erik_Kleinman_Ruiz.pdf?sequence=1&isAllowed=y

Lierta, A. C., & Kerstjens, A. P. (2020). Computación cuántica en la nube: un laboratorio en tu portátil. *Revista Española de Física, 34(1)*, 25-29. Obtenido de http://www.revistadefisica.es/index.php/ref/article/view/2577/2113

Luján-Mora, S. (2002). *Programación de aplicaciones web: historia, principios básicos y clientes web.* Alicante: Editorial Club Universitario. Obtenido de https://rua.ua.es/dspace/bitstream/10045/16995/1/sergio_lujan-programacion_de_aplicaciones_web.pdf

Magaz Romero, S. (2020). *Aprendizaje Máquina y Computación Cuántica.* Coruña, España: Trabajo de grado para optar al título de Ingeniero en Informática de la Universidad de La Coruña. Obtenido de https://ruc.udc.es/dspace/bitstream/handle/2183/26180/S.Magaz_Romero_2020_Aprendizaje_maquina_y_computacion_cuantica.pdf?sequence=3&isAllowed=y

Mazaeda Echevarría, R., de la Fuente López, E., González Sánchez, J. L., & Moya de la Torre, E. (2016). *DOCENCIA EN INFORMÁTICA INDUSTRIAL: LENGUAJES DE PROGRAMACIÓN.* Obtenido de https://uvadoc.uva.es/bitstream/handle/10324/17500/PID_15_156_Anexo1.pdf?sequence=1&isAllowed=y

Moroni, N., & Señas, P. (2005). *Estrategias para la enseñanza de la programación.* In I Jornadas de Educación en Informática y TICs en Argentina. Obtenido de http://sedici.unlp.edu.ar/bitstream/handle/10915/18901/Documento_completo.pdf?sequence=1&isAllowed=y

Palella Stracuzzi, S., & Martins Pestana, F. (2010). *Metodología de la investigación cuantitativa.* Caracas, Venezuela: FEDUPEL, Fondo Editorial de la Universidad Pedagógica Experimental Libertador.

Portugal, R., & Marquezino, F. L. (2019). *Introduccion a la Programacion de Computadores Cuanticos.* Capítulo 1. Obtenido de https://www.researchgate.net/profile/Frank_Acasiete/publication/341341621_Capitulo_1_Introduccion_a_la_Programacion_de_Computadores_Cuanticos_1/links/5ebb5e7a92851c11a865006f/Capitulo-1-Introduccion-a-la-Programacion-de-Computadores-Cuanticos-1.pdf

Reyes Alvarez, M. F. (2016). *Computación Cuántica Un Nuevo Reto para la Seguridad Informática.* XVI Jornada Internacional de Seguridad Informática. Obtenido de https://acis.org.co/archivos/JornadaSeguridad/Memorias/9.1.pdf

Roman Arenaza, R. E. (2019). *Lenguajes de programación Javascript Java y Javascript. Características. Norma de escritura. Variables y operadores lógicos. Mensajes. Ejercicios. Estructuras condicionales. Funciones y objetos. Aplicaciones.* Lima, Perú: Trabajo de grado para optar al título de Licenciado en Educación especialidad Informática de la Universidad Nacional de Educación. Obtenido de http://repositorio.une.edu.pe/bitstream/handle/UNE/3026/MONOGRAF%c3%8dA%20-%20ROMAN%20ARENAZA.pdf?sequence=1&isAllowed=y

Rossignoli, R. (2018). Computación Cuántica: ¿el futuro de los procesadores? *Revista Institucional de la Facultad de Informática | UNLP*, 17-19. Obtenido de http://sedici.unlp.edu.ar/bitstream/handle/10915/71774/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y

Rúa Vargas, J. P., & Branch Bedoya, J. W. (2009). Estado del arte de la computación cuántica. *Revista Avances en Sistemas e Informática, 6(2)*, 235-248. Obtenido de https://repositorio.unal.edu.co/bitstream/handle/unal/33388/20379-68840-1-PB.pdf?sequence=1&isAllowed=y

Ruiz, I. K., Fuster, G. G., & Ramírez, F. J. (s.f.). *Computación cuántica: Aplicaciones prácticas que la computación clásica no puede solucionar.* Obtenido de https://www.researchgate.net/profile/Ignacio_Kleinman/publication/337758188_Computacion_cuantica_Aplicaciones_practicas_que_la_computacion_clasica_no_puede_solucionar/links/5de8ae094585159aa462d589/Computacion-cuantica-Aplicaciones-practicas-que-la-comput

Sanz Pérez, M. A. (2019). *Estudio sobre computación cuántica con aplicación a particionado de grafos.* Trabajo d egrado para optar al título de Ingeniería en Tecnologías Industriales de la Universidad de Sevilla. Obtenido de https://idus.us.es/bitstream/handle/11441/94195/TFG-2556-SANZ%20PEREZ.pdf?sequence=1&isAllowed=y