Capítulo 1
Introducción a la programación

1.1. Un ejemplo

Con el fin de exponer una noción de lo que es programar veamos el siguiente ejemplo, suponga que un familiar suyo estuvo de viaje, visitó Japón, y le trajo de presente un robot, que solamente atiende a los dos siguientes tipos de ordenes:

- *avanzar* X centímetros
- *girar* X grados.

Una secuencia de ellas es posible dárselas al robot, para que recorra un camino determinado. Si queremos indicarle al robot (la carita feliz de color turquesa) que aparece en la figura 1.1 que se desplace hasta donde está el objetivo debemos de algún modo *"decirle"* lo que debe hacer, si suponemos que cada rectángulo de la cuadricula tiene diez centímetros de lado, las ordenes le dariamos a nuestro alegre amigo para alcanzar el objetivo podrian ser algo como:

Código 1 Ejemplo de instrucciones para llegar al objetivo.

```
1 avanzar 70cm.
2 girar 90 grados a la izquierda.
3 avanzar 250cm.
4 avanzar 80 cm.
```

Aunque ahora es posible darle algunas instrucciones a las máquinas mediante la voz, por ahora se las daremos a la antigua, escribiendolas,

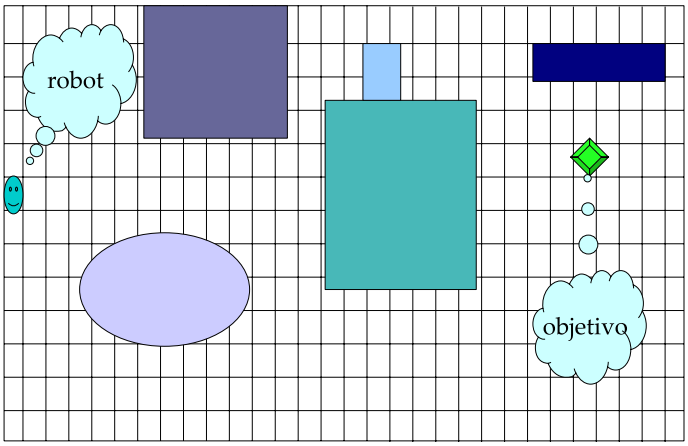

Figura 1.1: Un robot.

para hacerlo, debe existir algún tipo de teclado que nos permita digitarlas. La ordenes se graban para que se ejecuten una a una.

Si el robot toma las instrucciones dadas anteriormente relizará un recorrido como el mostrado en la figura 1.2. Lo que se acaba de hacer es *programar*, la programación de sistemas reales no difiere mucho de lo aquí mostrado, bueno, posiblemente se tengan a la mano más instrucciones y un sistema que no se llame "smile".

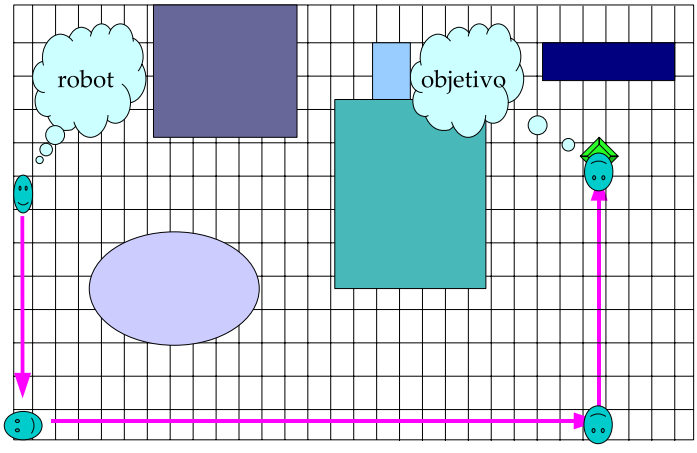

Figura 1.2: Ejecución de intrucciones.

1.2. Sistemas programables, algoritmos y programas

Hoy en día, las computadoras, ya sean de propósito general o especifico están por todas partes, teléfonos, electrodomésticos, aviones, etc; y realizan tareas de todo tipo, desde reproducir vídeo hasta controlar trayectorias de disparo de tanques, todas esas máquinas de cómputo requieren, como cualquier máquina, que se enciendan y sean controladas para realizar una realizar una tarea específica, la diferencia entre una computadora y un tractor (sin computadora de abordo) es que al tractor lo controla una persona y a la computadora lo que denominamos un *programa,* también llamado *software.*

Las computadoras son un ejemplo de *sistemas básados en programa almacenado*, todos estos sistemas poseen un procesador central, cuya actividad de una forma simple puede resumirse a:

1. Obtener una instrucción.
2. Determinar que instrucción es.
3. Ejecutar la instrucción
4. Ir al paso número 1

El conjunto de instrucciones que se desea que el sistema ejecute se almacena en algun tipo de memoria, RAM o ROM, dependiendo del sistema, por ejemplo muchos de los microcontroladores el programa se almacena en ROM, mientras que en las computadoras los programas son cargados a memoria RAM por el sistema operativo para su ejecución. En la figura 1.3 se muestra un ejemplo de estructura de un sistema basado en procesador.

Todo programa comienza con idea, algo que se quiere hacer, generalmente ese algo resulta como solución a un problema específico, la solución de un problema requiere el diseño de un *algoritmo.*

Algoritmo Palabra que proviene del nombre de un matemático y astrónomo árabe *Al-Khôwarizmi* del sigo IX, que escribio un tratado sobre la manipulación de números y ecuaciones llamado *Kitab al-jabr w'almugabala.* Un algoritmo es una secuencia ordenada de pasos, no ambiguos, expresados en lenguaje natural que conducen a la solución de un problema dado.

Los algoritmos deben cumplir con algunas características:

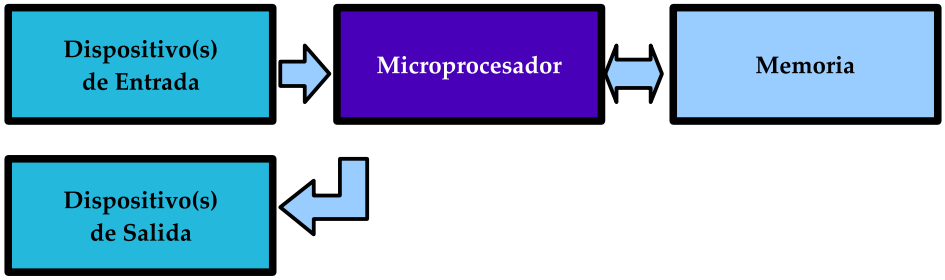

Figura 1.3: Sistema basado en procesador.

- *Preciso.* Indica el orden de realización de cada uno de los pasos.
- *Definido*. Si a un algoritmo se le suministra varias veces los mismos datos los resultados deben ser los mismos.
- *Finito.* El algoritmo debe terminar en algún momento.

Los algoritmos son soluciones abstractas a problemas, ellos generalmente son codificados en un lenguaje de programación y luego traducidos para que una computadora los pueda ejecutar y solucionar entonces un problema real. Los algoritmos son independientes del lenguaje de programación y de la máquina que lo ejecute, una analogia de la vida real [joyanes1989], la receta de un plato de cocina puede expresarse en ingles, frances o español e indicará la misma preparación independientemente del cocinero.

Lenguaje de programación Son conjuntos de instruccines con que se pueden escribir los algoritmos para que un sistema lo ejecute. Existen múltiples tipos de lenguajes de programación:

- *Lenguaje de máquina.* Es un lenguaje compuesto por códigos binarios que un sistema puede ejecutar directamente, los programas ejecutables son precisamente secuencias de instrucciones en lenguaje de máquina, un ejemplo de intrucciones en lenguaje de máquina es:

```
0011 0000 0001
0101 0001 0011
```

  Las anteriores intrucciones le indican a un procesador que sume dos datos y que luego multipliquen ese resultado por otro. Las instrucciones en lenguaje de máquina estan compuestas de un código que

identifica la instrucción (*opcode*) y uno o varios operandos (o referencias a los mismos). Depende del tipo de procesador sobre la cual se esté programando, Intel, Motorola, Atmel, etc, cada uno de ellos tiene códigos de operación diferentes.

- *Lenguajes ensambladores.* Escribir programas funcionales en lenguaje de máquina es una tarea que pocas personas desean hacer, pues es muy propenso a errores y tedioso, por ello a alguien se le ocurrío asociar símbolos o mnemonicos a las instrucciones que una máquina podia realizar, por ejemplo en algun lenguaje ensamblador las instrucciones en lenguaje de máquina antes mencionadas quedarían:

```
add [0] [1]
mul [1] [3]
```

  Para convertir los programas en lenguaje ensamblador a código de máquina se usa un programa llamado *ensamblador.*

- *Lenguajes de alto nivel.* Son lenguajes que tienen conjuntos de instruccines similares a las palabras del idioma ingles (o algún otro) que son más fáciles de entender por los seres humanos. En C, las instrucciones para hacer lo que hicimos en lenguaje de máquina y ensamblador serían:

```
res=(a+b)*c;
```

  Existen muchos lenguajes que se pueden llamar de alto nivel, Basic, Pascal, Fortran. Lenguaje C, comparado con el lenguaje ensamblador es de alto nivel, pero tiene la característica de permitir acceder a muchos de los recursos de hardware disponibles en un sistema, por lo que mucho lo clasifican como lenguaje de nivel medio. Para convertir los programas escritos en un lenguaje de alto nivel en codigo de máquina se tienen las siguientes opciones:

  - *Intérpretes.* En los lenguajes interpretados, cuando se ejecuta un programa cada instrucción se traduce a lenguaje de máquina y a continuación se ejecuta, ejemplos de lenguajes interpretados son matlab, python, smalltalk.

- *Compiladores.* Traducen completamente los programa fuente a lenguaje de máquina, ejemplos de lenguajes compilados son, C/C++, Pascal, Fortran, COBOL.
- *Híbridos.* En los últimos años aparece una especie de lenguajes que combinan las dos opciones anteriores, existe un compilador que traduce el código fuente en un código intermedio que es ejecutado por un programa especial denominado máquina virtual. Ejemplos de esto son Java y los lenguajes de la plataforma .Net de Microsoft.

1.2.1. El proceso de traducción

Como se había mencionado antes, cuando se tiene un algoritmo listo, es posible programarlo en algún lenguaje de programación con el fin de ejecutarlo en una computadora, para ello, es necesario traducirlo a código de máquina, el proceso de generación de un programa puede dividirse en los siguientes pasos:

1. Se escribe el algoritmo en algún lenguaje de programación en un editor de texto y se guarda en un archivo.
2. Se utiliza un compilador para generar el ejecutable, en este paso si existen errores de sintaxis el compilador genera un mensaje de aviso.

En la figura 1.4 se muestra esquematicamente el proceso de traducción de un programa escrito en un lenguaje de alto nive a código entendible por una máquina.

1.2.2. Un poco de historia de los lenguajes de programación

Según lo que se dice, fue Charles Babage a mediados del sigo XIX el que creo el primer lenguaje de programación, dicho lenguaje era para una máquina diseñada por él, llamada la máquina analítica, como compañera de Babage estaba Ada Byron considerada como la primera persona que escribio un programa. La máquina de Babage solo se construyó alrededor de un siglo después. La primera computadora la ENIAC se programaba directamente modificando el hardware de la misma mediante cables, luego se usaron las ideas de Babage y se relizaban programas utilizando tarjetas perforadas.

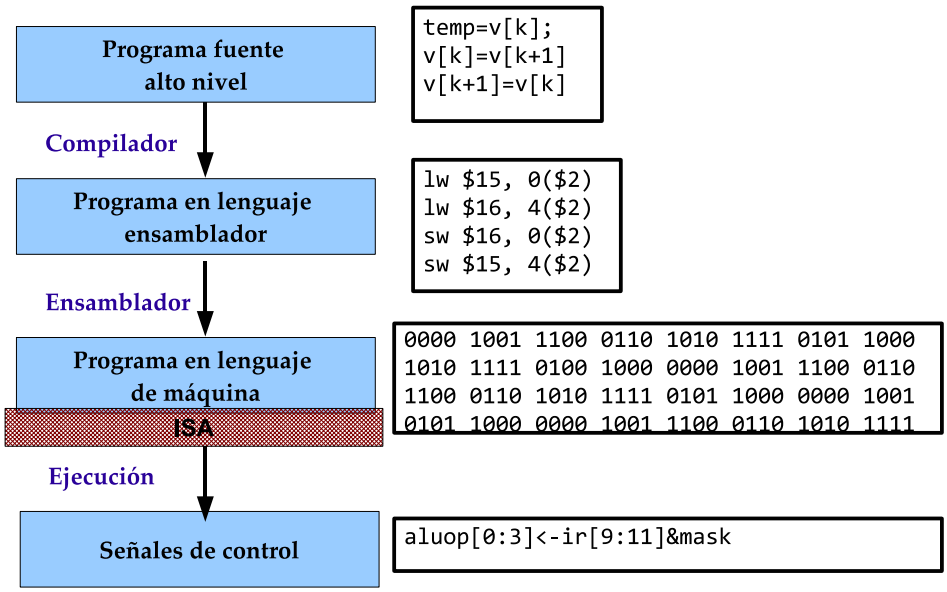

Figura 1.4: Traducción de un programa.

En la decada de los 50 surgen los primeros ensambladores simbólicos, no era necesario indicar explicitamente las direcciones de memoria, si no que podiamos usar variables para referirnos a ellas.

En los 60 aparece la *programación estructurada*, comienza el uso de instrucciones de control de flujo de alto nivel (if, else while, etc, que empiezan a remplazar el uso del goto), funciones y procedimientos, algunos de los lenguajes que aparecieron en esa epoca son: Fortran IV, COBOL, Lisp, Algol 60, Basic, Simula-67 (precursor de lo que fue más adelante la programación orientada a objetos) entre otros.

En los 70 surge el diseño orientado a los *tipos abstratos de datos* y con ello lenguajes que permitian implementar aplicaciones así diseñadas, Pascal, Prolog, lenguaje B, lenguaje C (nuestro querido amigo). Modula-2.

En los 80 entra en escena la *programación orientada a objetos* y el languaje Smalltalk-80, C++, Object-Pascal, Oberon, Ada (no orientado a objetos del todo)

En los años noventas la revolución de las interfaces de usuario junto con la yá en carrera programación orientada a objetos hacen que los lenguajes y entornos de programación permitan el desarrollo de aplicaciones *controladas por eventos*, y aparecen lenguajes-entornos como Visual-

Basic, Delphi y el lenguaje Java.

En el año 2000 Microsoft lanza la plataforma .Net con varios lenguajes en ella, entre los que se encuentra C#, un lenguaje orientado a objetos y componentes del que hablaremos alguna páginas adelante.

1.2.3. Fases del desarrollo de sofware

Construir cualquier cosa requiere de un cierto proceso, un orden, no debemos pintar una pared antes de levantarla, no podemos, despúes de terminar una casa que fue planeada para ser de un piso, convertila en un edificio, el proceso de crear un programa no es la excepción. Existen muchos textos en los cuales se exponen las fases del desarrollo de software, utilizando como referencia a [Booch1996] es posible indicar que la solución de problemas mediante programas de computadora (creación de un producto de software) puede dividirse en las siguientes fases:

- Análisis.
- Diseño.
- Implementación.
- Pruebas.
- Implantación.
- Mantenimiento.

Análisis. Aquí se asegura de que el problema esté completamente especificado y que se comprende completamente. Me parece muy util citar la siguiente espcificación de un problema [Bronson2000]:

> *Escriba un programa que proporcione la información que necesitamos acerca de los círculos. Termínelo para mañana.*
>
> -La gerencia.

Espero que ninguno de los lectores haya dejado de leer para sentarse en su computadora para trabajar en su lenguaje favorito. Una revisión rápida a la especificación del anterior problema nos indica que es vaga, pues no

se expresa de manera precisa qué es lo que se necesita a cerca de los círculos. Digamos que usted recibe la solicitud y mañana le lleva al gerente un programa que calcula el área un círculo dado su radio, pero el el gerente le dice *"no! yo lo que quería era un programa que calculara el perímetro"* no le vaya a decir esto a su jefe, pero el problema fue que no le especificaron los requirimientos del programa. En últimas el análisis proporciona un modelo de la forma en la que un sistema se comporta.

Diseño. Si el programa lo que tiene que hacer es una sola cosa en esta fase se diseña el algoritmo que realiza el proceso, Si es un programa de mayor tamaño aquí se crea la arquitectura a usar para la implementación y se establecen las políticas tácticas comunes.

Implementación. Se realiza la implementación del programa en un lenguaje o lenguajes de programación.

Pruebas. Se corroborar que cumple con los requisitos.

Implantación. Llevar el sistema a los clientes.

Mantenimiento. Los programas del mundo real generalmente deben evolucionar con el tiempo con el fin de no volverse obsoletos con el tiempo, por ello es necesario actualizarlos cada cierto tiempo.

1.2.4. Qué se necesita para programar

La programación no es solo conocer un lenguaje y escribir aplicaciones, existen varios tipos de conocimiento necesarios que hacen que programar no sea solamente un arte sino que sea algo sistematico, dentro de los conocimientos necesarios en la formación en esta área, podemos citar [Villalobos2005]:

- Modelaje. Análisis y especificación de problemas.
- Algorítmica. Diseño de algoritmos.
- Tecnología y programación. Lenguajes de programación, modelaje etc.
- Herramientas de programación. Editores, compiladores, depuradores, gestores de proyectos, etc.

- Procesos de software. División del proceso de programar en etapas claras, ciclo de vida del programa, formatos, entregables, estandares de documentación y codificación, técnicas de pruebas.
- Técnicas de programación y metodologías. Estrategias y guías que ayudan a crear un programa. Como se hacen las cosas.
- Elementos estructurales y arquitecturales. Estructura de la aplicación resultante, en terminos del problema y elementos del mundo del problema. Funciones, objetos, componentes, servicios, modelos, etc. La forma de la solución, responsabilidades y protocolos de comunicaciónes.

Se recomienda hacer que la formación en programación ataque los elementos citados de manera uniforme, pero esto es más fácil decirlo que hacerlo.

1.3. Algoritmos

-TODO, tipos de dato, variables, expresiones, funciones.

1.4. Ejercicios

1. Escriba un algoritmo para preparar una taza de café.
2. Escriba un algoritmo para realizar una llamada telefónica desde un teléfono público.
3. Suponga que las instrucciones del robot *smile* en lengueje de máquina son:

opcode	*operando*	*significado*
0000 0001	no tiene	rotar 90 grados a la derecha
0000 0010	no tiene	rotar 90 a la izquierda
0000 0011	xxxx xxxx (cm)	avanzar X centímetros

Cuadro 1.1: Instrucciones de smile.

Escriba el progarama que aparece en 1 en lenguaje de máquina usando la tabla anterior, necesita convertir los valores de centímetros a binario.

4. Inventese una instrucción para tomar un objeto y escriba un programa ir por el objetivo, tomarlo y regresar a la posición de origen.

5. Dado que la instrucción avanzar tiene un operando que es de 8 bits de longitud, el máximo número de centímetros que se puede avanzar es de 255 (2^8), Escriba un programa para avanzar 325 centímetros en línea recta.

6. Escriba un algoritmo para intercambiar el contenido de dos tazas de cafe. suponga que tiene una tercera taza disponible para realizar el proceso. Cada taza debe enjuagarse antes de recibir contenido nuevo.

7. Escriba un algoritmo para encotrar el número más pequeño de un conjunto de tres.

8. Escriba un algoritmo para encontrar el número de veces que aparece le letra *a* en una oración.