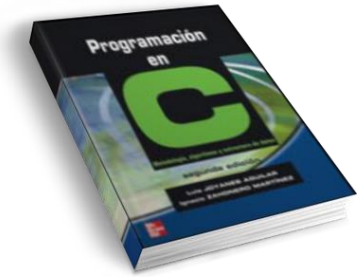
Programación en

Un poco de historia



- *C es un **lenguaje de propósito general**, es decir, se pueden desarrollar aplicaciones de diversas áreas.*
- *Dentro de sus **principales características** podemos mencionar que:*

- *Es un **lenguaje estructurado**, tiene una abundante cantidad de operadores y tipos de datos.*

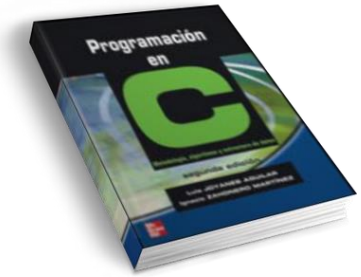
- *Es un **lenguaje de nivel medio**, pero se puede codificar a alto nivel, produce código objeto altamente optimizado, posee punteros y capacidad de aritmética de direcciones.*

- *C fue creado en los Laboratorios Bell de AT&T para correr originalmente en el sistema operativo Unix*

Ventajas de trabajar en C

- *Entre sus **múltiples ventajas** podemos mencionar que:*
- *C es un lenguaje muy **portable**, es decir, es independiente de la arquitectura de la máquina y con alguna o ninguna modificación un programa puede correr en una amplia variedad de computadores.*
- *Es relativamente flexible en la conversión de datos.*
- *Su **eficiencia y claridad** han hecho que el lenguaje ensamblador casi no haya sido utilizado en Unix.*
- *El compilador de C es pequeño y tiene un gran poderío debido a sus múltiples bibliotecas.*
- *Sin embargo, tiene sus desventajas y entre ellas se puede mencionar*

Como desventajas



Podemos mencionar:

- *La excesiva libertad en la escritura del código fuente hace que muchas veces se cometan **errores de programación**, que, por ser correctos sintácticamente no se detectan en tiempo de compilación.*

- *Carece de **instrucciones de entrada y salida**, de **manejo de strings** (cadenas de caracteres), quedando el trabajo en manos de las bibliotecas provocando con esto algunos problemas de portabilidad.*

Pero también

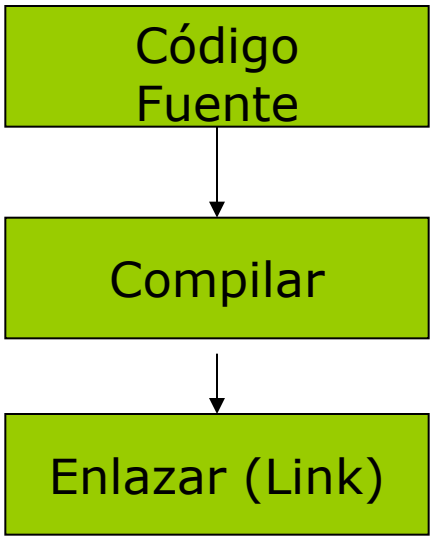
- *Hay múltiples ejemplos de aplicaciones construidas en C, como ser:*

- *construcción de **sistemas operativos**, procesadores de texto, administradores de bases de datos (por ejemplo Clipper).*

- *Programas para Windows y Windows 95, las APIs (Application Program Interface) de las Librerías de Enlace Dinámico (DLL) de **Windows están construidas mayoritariamente en C**.*

Forma General del Lenguaje C

- *Para **crear** un programa en C, se escribe el **código fuente** (programa), luego se **compila** y finalmente se **enlaza** con las bibliotecas (se hace un link, en nuestra jerga diríamos "se linkea").*



Para hacer un programa en C, lo primero que se debe hacer es crear el programa fuente (con extensión ".c")

Forma General del Lenguaje C

- *Todo lenguaje de programación posee **palabras claves**, estas son aquellas palabras que reserva el lenguaje para identificar **ciclos, estructuras** y en general cualquier cosa que sea **parte de instrucciones.***

- *Por ejemplo, las palabras **while**, **if**, **struct** son palabras claves en C.*

- *En C las **palabras claves** o **reservadas** deben escribirse en **minúsculas**, esto ya que C diferencia entre mayúsculas y minúsculas (a diferencia del lenguaje Pascal)*

Como escribir un programa en C

- *El **Lenguaje C se escribe** a partir de un programa principal y de funciones, el programa principal también es una **función** (función principal), y se denomina **main(),** esta debe estar siempre presente y es la primera en ser llamada.*
- *Para hacer un programa en C, lo primero que se debe hacer es **crear el programa** fuente (con extensión ".c") de la siguiente manera:*

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
  int c=0;
  int b=0;
  printf("Ingrese entero");
  scanf("%d",&c);
  b=c*c;
  printf("el cuadrado es %d ", b);
  system("PAUSE");
  return 0;
}
```

Estructura de un Programa en

- ***Cuando se escribe el programa se recomienda usar el siguiente formato:***

- **llamadas a bibliotecas**
- **declaración de funciones (prototipos de funciones)**
- **declaración de variables globales**
- **main()**
- **{**
- **declaración de variables locales**
- **sentencias**
- **}**
- **definición de funciones**

```
/* esto es un comentario */
#include<stdio.h>
#include<stdlib.h>
 float num;
float res, cubo(int x);
main()
{
printf("ingrese un
número  ");
scanf("%f",&num);
res=cubo(num);
printf("%f al cubo es:
%f",num,res);
system("PAUSE");
 return 0;
}
float cubo(float x)
{return x*x*x;}
```

Estructura de un Programa en

- *Por lo general las bibliotecas contienen funciones creadas por el programador o entregadas con el producto. Por ejemplo las funciones de entrada y salida vienen especificadas en la biblioteca de entradas y salidas estándar stdio.h (Standar Input Ouput Header).*
- **# include <stdio.h>**
- *Cada vez que se escribe una línea, esta debe llevar ";" al final, con lo cual el compilador entiende que llega al final de una instrucción o proposición.*
- *Antes de usar una variable dentro de un programa, es necesario indicarle al compilador que va a ser usada, para ello se declara:*
- **<tipo> <lista_de_variables>; Ejemplo: Int a;**
- *Como ya es sabido, una asignación es almacenar en una variable el resultado de una operación o proceso. En "C" una asignación se representa como:*
- **<variable> = <expresión>;**
- *Ejemplo:*
- **a=5; /* a la variable a se le asigna el valor 5 (constante)*/**
- **expo=max; /* a la variable expo se le asigna el contenido de la variable max*/**
- **expofin=ult+4; /* a expofin se le asigna el contenido de la variable ult sumada a la constante 4*/**

Identificadores

- *Estos son simplemente los **nombres** que pueden asumir **variables** o **constantes** dentro del programa.*
- *Existen algunas **restricciones** para los identificadores:*

- *No deben ser iguales a nombres de palabras reservadas o bibliotecas.*
- *Deben comenzar con una letra o el caracter '_'.*
- *No deben tener en medio el caracter espacio.*

Ejemplo de identificadores:
Contador
Conta
var1
variable3
Suma_total

- ***Comentarios***
- *Un comentario es una cadena de caracteres que no es tomada en cuenta por el compilador, esta va dentro de "/*" y "*/",*
- *Ejemplo:*
- /* esto es un comentario */

Entrada / Salida Estándar

- *La biblioteca (o archivo de cabecera) llamada **stdio.h** contiene todas las funciones de entrada y salida estándar, entre ellas las funciones **printf()** y **scanf()**. La **f** significa que es entrada o salida con formato.*
- ***printf()***
- *Esta es la función de salida por pantalla, su formato es el siguiente:*
- **printf("cadena_de_control: %",lista_de_argumentos)**
- **printf("a * b = %d \n", c);**
- *Los caracteres **barra invertida** (backslash) y **n** juntos, provocan un retorno de carro (return) en pantalla.*

Entrada / Salida Estándar

- ***scanf()***
- *Esta es la función para entrada estándar, se utiliza de un modo parecido a la anterior.*
- **scanf("%d",&numdato);**
- *%d indica que se lee con formato entero. En el ejemplo el caracter "&" se usa para indicar la dirección de memoria, es decir, "lee a un valor con formato entero y lo guarda en la dirección de memoria numdato".*

Operadores

- *Son caracteres especiales que tienen un significado específico o determinado, estos indican al compilador realizar operaciones* ***matemáticas o lógicas****.*

Operadores Matemáticos

- **- menos unuario**
- **- resta**
- **+ suma**
- *** producto**
- **/ división**
- **-- Decremento**
- **++ Incremento**
- **% módulo**
- **= asignación**

Operadores

Operadores de Relación

Estos se utilizan dentro de proposiciones del tipo ***while, if, for;*** *es decir, en estructuras de control .*

- **< : menor que**
- **> : mayor que**
- **<= : menor o igual que**
- **>= : mayor o igual que**
- **== : igual a**
- **!= : distinto de (no igual a**)

Operadores Lógicos

- ! : negación ------- not
- && : y lógico ------- and
- || : o lógico -------- or

Estructura de control

- La sentencia de control básica es if (<condicion>) then <sentencia/s> else <sentencia/s>. En ella se evalúa una expresión condicional y si se cumple, se ejecuta la sentencia s; si no, se ejecuta la sentencia t. La segunda parte de la condición, else <t>, es opcional.
- if (CONDICION) then /* Se pregunta "es dato2 igual a cero" */
- printf("\n LA DIVISION ES INDETERMINADA");
- else
- {
- printf("\n LA DIVISION ES: %f", div); /*se ejecuta si dato <> 0*/
- }