programa de BECAS
DGTIC
de FORMACIÓN en TIC

Seguridad Informática

Lenguaje C y llamadas al Sistema.

Fernando C.

Temario.

- Programación en C 101.
  - Historia del lenguaje de programación C.
  - Niveles de abstracción.
  - Estructura básica de un programa en C.
  - Conceptos, tipos de datos, variables, declaraciones, etc.
  - Estructuras de control.
    - Estructuras de selección.
    - Estructuras iterativas.
  - Funciones.
  - Valores de retorno.
  - Operadores a bajo nivel.
  - Estructuras, campos de bits, uniones y otras estructuras.
- GDB
  - Fundamentos de lenguaje ensamblador.
- Arreglos y strings.
- Apuntadores y memoria dinámica.
- Proceso de compilación.
- Preprocesador y compilación condicional.
- Makefiles.
- Automatización de la compilación.
- Manejo de archivos.
  - Flujos.
  - Descriptores de archivos.
- Uso de bibliotecas externas.
  - Matemáticas con C.
  - Networking con C.
  - Cifrado con C.
  - Bases de datos con C.
  - Interfaces graficas con C.
- Interacción de C con el sistema operativo.
- Clasificación de llamadas al sistema.
- Implementación de llamadas al sistema.
- Módulos de kernel.
- “Programación segura”.

Duración del curso.

- Inicio 23/Enero
- Fin 1/Febrero (Examen)

- Horario. 14:00 – 18:00
- Descanso: 25 a 30 min.
- Extraclase: 18:00 – 20:00

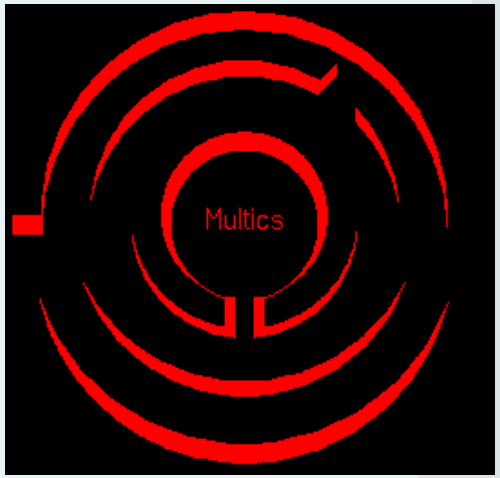
- Repositorio de github a:
  - 6665726e616e646f@gmail.com

- Evaluación:
  - %
  - %
  - %

Programación en C 101

Historia.

- Multics. 1964-1970 (MIT, General Electric && Bell Labs)

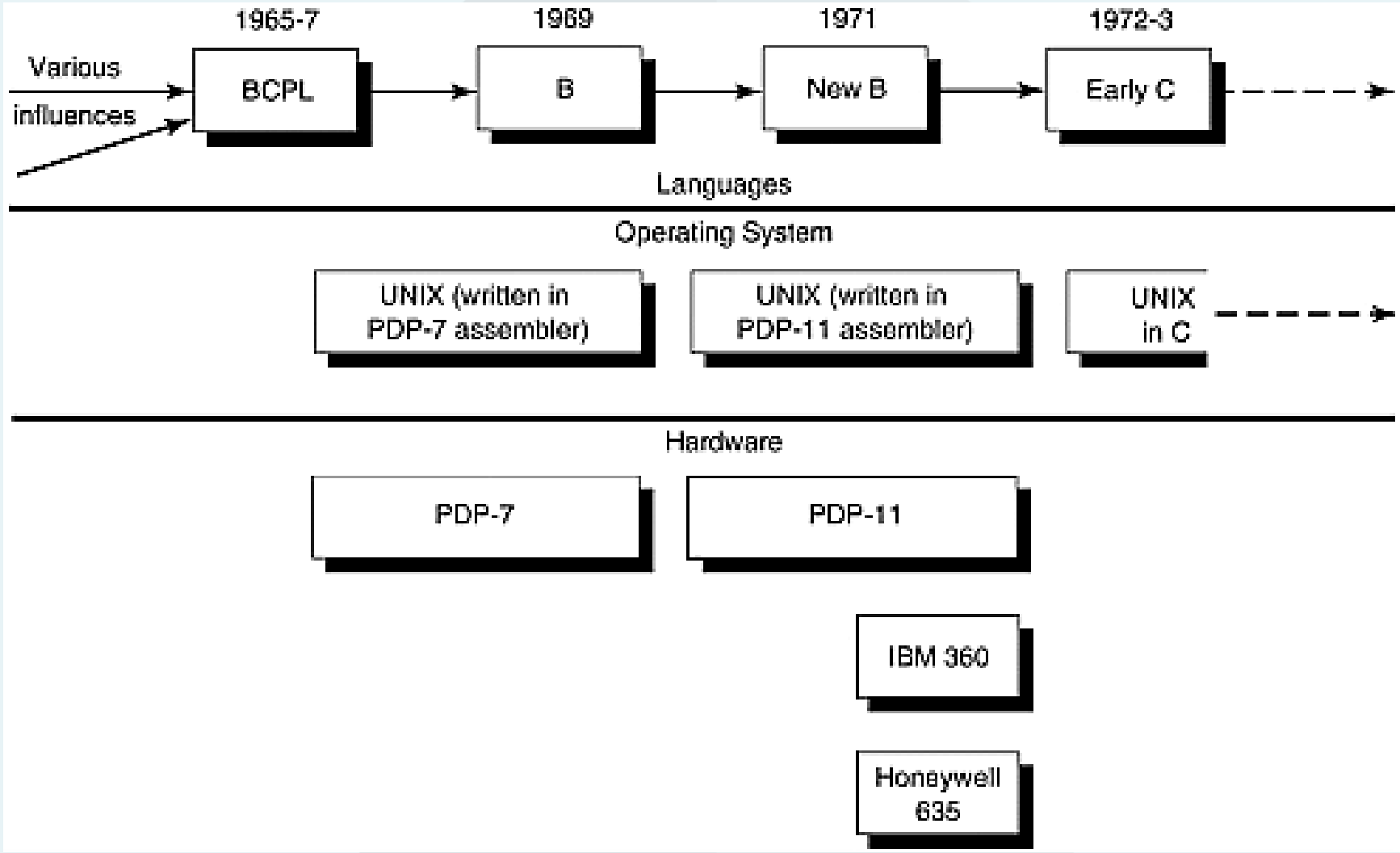


- El fracaso de MULTICS derivo en conocimiento, que sirvió para el desarrollo de UNIX.
- ¿Has escuchado hablar sobre Ken Thompson?

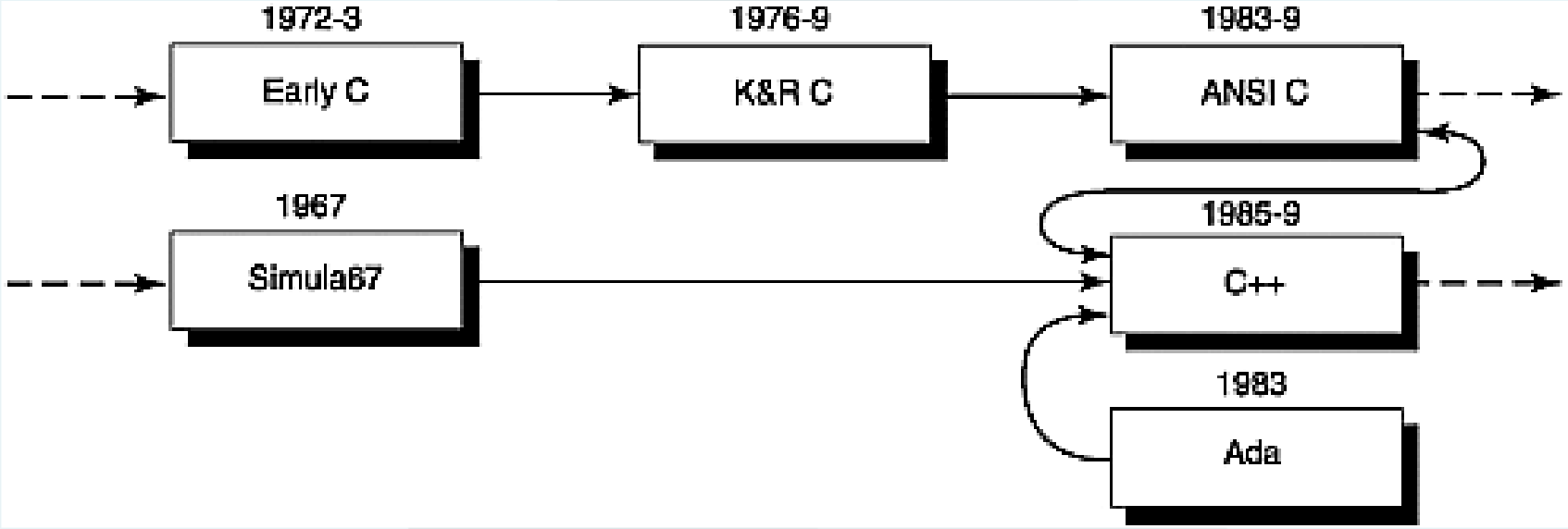
Ritchie && Thompson

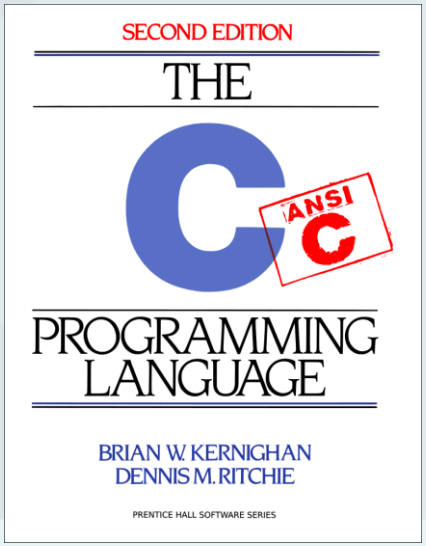
C fue una mejora de Ritchie al lenguaje de programación B, desarrollado por Thompson.

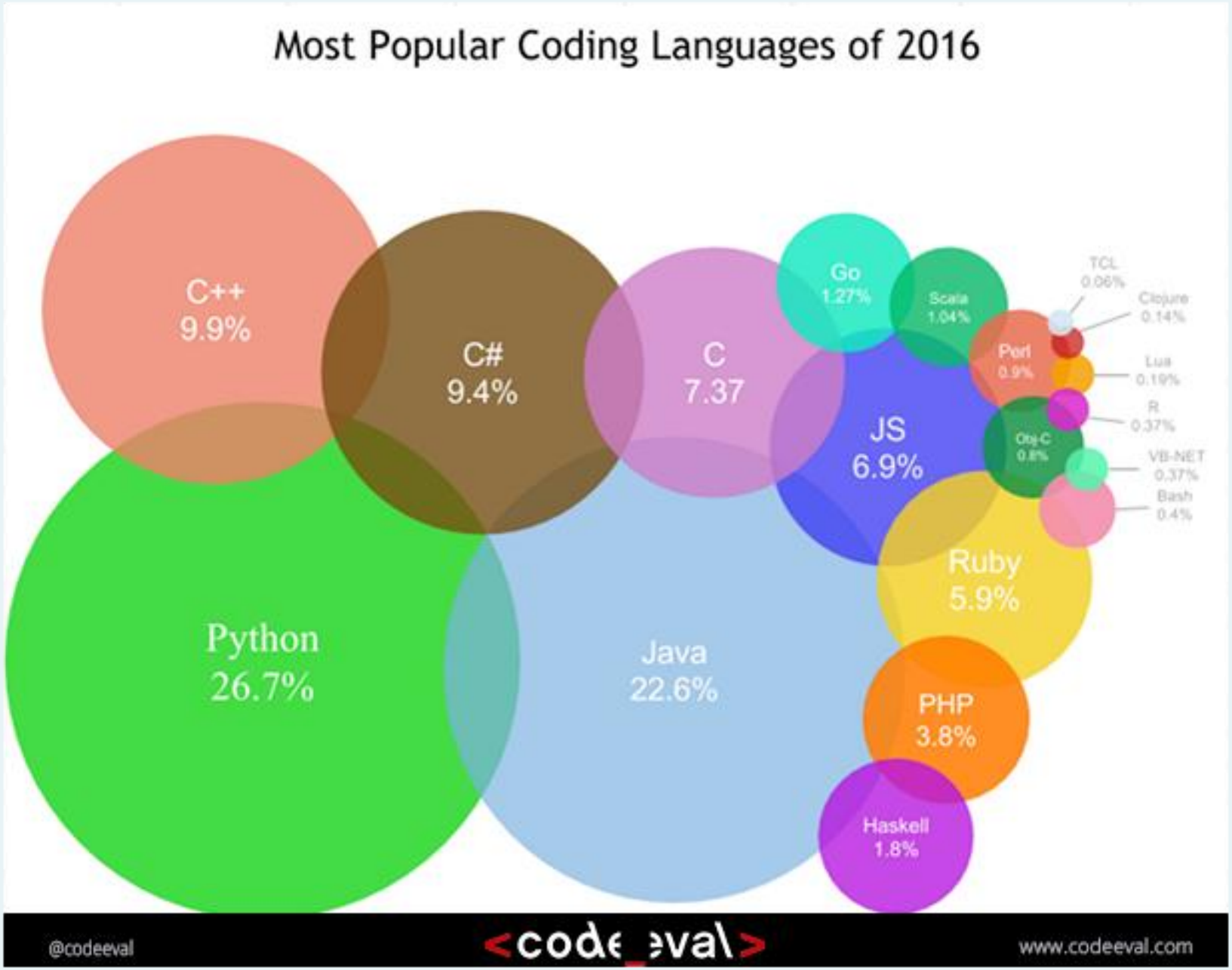
¿De donde viene...?



Evolución y legado.






Most Popular Coding Languages of 2016
C++
9.9%
C#
9.4%
C
7.37
Go
1.27%
Scala
1.04%
TCL
0.06%
Clojure
0.14%
Perl
0.9%
Lua
0.19%
R
0.37%
Obj-C
0.8%
VB.NET
0.37%
Bash
0.4%
JS
6.9%
Ruby
5.9%
Python
26.7%
Java
22.6%
PHP
3.8%
Haskell
1.8%
@codeeval
<code_eval>
www.codeeval.com

¿Por qué es tan trascendental?

- C creció con UNIX, UNIX creció con C.
  - Durante los 70s Bell Labs se vio envuelto en escándalos de practicas monopólicas.
  - Solución: Liberar UNIX al publico.
  - Berkeley Software Distribution. ☺
  - GNU
  - GNU/Linux…
- N cantidad de lenguajes de programación han sido desarrollados sobre C, sus interpretes son compilados a través de cc o gcc, y la creación de nuevas extensiones para dichos lenguajes son desarrolladas en C. Ejemplo, python.

Estándares/dialectos.

- C de K & R
- ANSI C (Estándar en Windows.)
- C89
- C99
- C11
- POSIX
- C++(?)

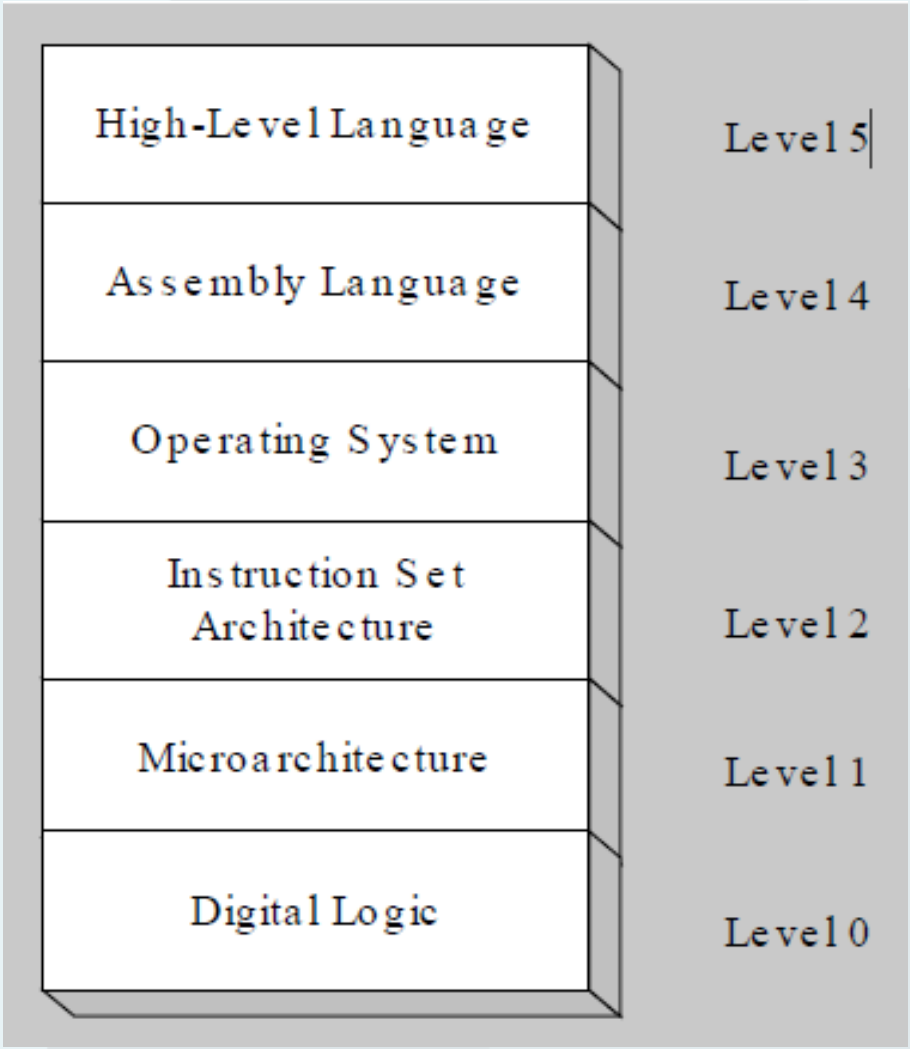
Para este curso:

- GNU/Linux o BSD. (Sustituir apt-get por el gestor de paquetes en cuestión)
  - apt-get install build-essential (todo lo básico referente a C)
    - gcc(GNU compiler collection), compatible con todos los estándares y dialectos de C ☺
  - apt-get install git (Desarrollado también por Linus Torvalds, en C, no C++)
- Si quieres usar Windows, posiblemente esperes con ansias el curso de POO, donde jugaras con .net e interactuaras un poco con el kernel de Windows, pero si en verdad quieres usar Windows para este curso, instala cygwin u Orwell Dev C++.
- Si usas BSD(MacOS) descarga los ports, y entonces instala a mano las herramientas que utilicemos.

¿Es importante C en ciberseguridad?

- China and Russia have thousand of well trained cyber terrorists and we are just sitting ducks.
  - George Ledin.

Niveles de abstracción.



Entonces.

- C es un lenguaje de alto nivel, que interactúa con el sistema operativo a muy bajo nivel. La versatilidad de C consiste en la facilidad que tiene para manejar la memoria, esto hace que frecuentemente se le denomine como un lenguaje de programación de sistemas, aunque esa denominación es muy genérica para un lenguaje tan útil.

Algunos conceptos.

- Código fuente.
- Compilador.
- Lenguaje ensamblador.
- Código objeto.
- Instrucciones de maquina.
- Arquitectura de la CPU.
- Ejecutable/ binario.

Características generales de los lenguajes de programación.

- Tipado.
  - Dinámico. (Comprobación de tipos en tiempo de ejecución.)
  - Estático. (Comprobación de tipos en la compilación.)
- Compilado o interpretado.
- Paradigma.
  - Estructurado.
  - Orientado a objetos.
  - Funcional.

- En C:
  - Tipado estático.
  - Lenguaje compilado.
  - Paradigma estructurado.

Algo clásico.

```
#include<stdio.h> //header (preprocesador)
char *cadena = “Hola, Mundo\n”; //Variables globales.
int main()  //Funcion main()
{
        printf(“%s”,cadena); //Cuerpo de la función main()
}
```

En línea de comandos:

```
gcc –Wall -g hola.c –o hola
```

Mas conceptos.

- Variable.
- Dato.
- Declaración.
- Función.
- Tipo de retorno.
- Constante.
- Operador.
- Palabras reservadas.

Tipos de datos.

Variable Type	Keyword	Bytes Required	Range	Format
Character (signed)	Char	1	-128 to +127	%c
Integer (signed)	Int	2	-32768 to +32767	%d
Float (signed)	Float	4	-3.4e38 to +3.4e38	%f
Double	Double	8	-1.7e308 to + 1.7e308	%lf
Long integer (signed)	Long	4	2,147,483,648 to 2.147.438.647	%ld
Character (unsigned)	Unsigned char	1	0 to 255	%c
Integer (unsigned)	Unsigned int	2	0 to 65535	%u
Unsigned long integer	unsigned long	4	0 to 4,294,967,295	%lu
Long double	Long double	10	-1.7e932 to +1.7e932	%Lf

Practica 2. LIMITES

- Ejecuta el siguiente comando.
  - find / -name limits.h

  Como retorno obtendrás la ubicación del header limits.h, analízalo rápidamente y ubica los nombres de las variables que definen máximo y mínimo para cada tipo de dato.

  Tomando como base tu programa de hola mundo, importa este archivo .h y utiliza estas variables.

  Imprime dichos valores con la función printf teniendo en cuenta el formato.

Practica 3. Fin del mundo.

- El tiempo en la mayoría de los sistemas derivados de UNIX, se mide en segundos transcurridos desde el 1 de enero de 1970, este dato no se ha cambiado desde entonces, la sincronización de los servidores que mueven al mundo se basan en este contador.
- El problema es que esta variable tiene un limite, y cuando se alcance este limite algo interesante sucederá…
- ¿Puedes hacer un programa que diga la fecha exacta de este suceso?
- Pistas:
  - La variable se almacena en un tipo de dato int de 32 bits.
  - time.h define el tipo de dato time_t y la función ctime(), la cual regresa una cadena con formato de fecha.

Operadores y su precedencia.

- man 7 operator
- Live session :D

Estructuras de control

- No son propias de C, sino del paradigma estructurado.
- Son la base de muchas soluciones, y lo que se suele enseñar en un curso de programación básica.
- En C:
  - Estructuras de selección.
    - If
    - Switch
  - Estructuras iterativas
    - While
    - Do while
    - For

GOTO

- Edsger Dijkstra ☺
  - GOTO se utiliza ampliamente en lenguaje ensamblador para implementar saltos, ya sean estos condicionales o no condicionales, GOTO era empleado en C ampliamente. Sin embargo un documento escrito por Dijsktra, dio pie a muchas malintepretaciones por su titulo, en dicho documento se exponía una idea simple, un programador sin experiencia puede equivocarse mas fácilmente empleando GOTO que estructuras de control, y se puede lograr lo mismo con GOTO que con estructuras de control básicas, sin embargo poco a poco GOTO fue considerado como dañino y cayo en desuso, aunque aun es soportado por cualquier compilador de C.
  - Aquí el documento.

Ejemplo IF

```
#include<stdio.h>
int main()
{
        if(10>9) //Condición, si se cumple, se ejecuta el código del bloque.
        {
                printf("SI\n");
        }
}
```

Dentro de cualquier expresión condicional, cualquier valor diferente de 0, será considerado verdadero.

Ejemplo SWITCH

```
#include<stdio.h>
int main()
{
        int k;
        scanf("%d",&k);
        switch(k)
        {
                case 1:
                        printf("UNO\n");
                        break;
                default:
                        printf("No es uno\n");
        }
}
```

La palabra reservada break "rompe" la ejecución de la estructura de control y sale de ella.

Operador ternario.

- Funcional en muchos lenguajes de programación, se dice que hace el código ilegible, pero es bello.

- (condición)?(acción verdadero): (acción falso)

- Ejemplo:

```
int main()
{
        int a,b;
        scanf(“%d %d”,&a,&b);
        printf(”%d es mayor”,(a>b?a:b));
}
```

while

- Las estructuras iterativas, repiten una acción hasta que se cumpla con una condición especifica, o mientras se cumpla con una condición especifica, por lo que lo que va entre paréntesis, debe arrojar un valor true o false.

```
int main()
{
        int k;
        scanf("%d",&k);
        while(k)
                printf("%d\n",k--);
}
```

Do while

- Do while, revisa la condición al final de la ejecución de lo que se encuentra dentro del bloque de código.

```
int main()
{
        do
        {
                printf(“NON DVCOR, DVCO!\n”);
        }
        while(0>1);
}
```

For

- For agrupa la declaración de la variable, la condición y el incremento de la variable en su cuerpo.

```
int main()
{
        for(int i=1;i<=30;i++)
                printf(!(i&1)?"%d\n":"",i);
}
```

Practica 4. Recordando programación.

- Realiza los siguientes programas, guárdalos todos en el mismo directorio. Coméntalos.
- Esta practica vale por 2.

1. Realiza un programa que imprima los números de 0 a n, n debe ser ingresado por el usuario.
2. Realiza un programa que imprima los números nones de 1 a n, n debe ser ingresado por el usuario.
3. Realiza un programa que imprima los números primeros n números primos, n debe ser especificado por el usuario.

4. Realiza un programa que tome como parámetro un carácter, hecho esto, regresa su equivalente hexadecimal en la terminal, el formato es tu decisión.

5. Realiza un programa de autenticación, es decir que acepte un “password”, de ser correcto, imprimir “OK” de lo contrario imprimir, “GTFOH!”

6. Realiza un programa que calcule área de ciertas figuras geométricas, solamente, triangulo, rectángulo, cuadrado, dependiendo de la selección del usuario, se pedirán, base, altura o solo lado.

Funciones.

- Divide et conquera.
  - Las funciones es donde radica el poder de C.

- En matemáticas la notación de una función f(x) = y, donde y varias en base a lo que la función reciba como variable x.
- En programación la notación es similar.
  - función(argumentos) = valor de retorno o acción.

- main() es una función, los paréntesis lo delatan.
- Puede recibir argumentos por línea de comandos gracias a su estructura de funcion.

Estructura de un programa con funciones.

```
#include<stdio.h>
int suma(int,int); //Prototipo de función.
int main()
{
        printf(“%d”,suma(1,1));
}
Int suma(int a, int b) //Declaracion de funcion
{
        return a+b;
}
```

Paso por valor y por referencia.

- Cuando se declara una función, y esta recibe argumentos con los cuales trabajar, existe la opción de pasar estos argumentos por valor y por referencia.
- Paso por valor se refiere a crear una copia del objeto en cuestión, mientras que referencia se refiere a trabajar con el objeto directamente, para esto se emplea la dirección en memoria.
- Los arreglos, se tratan directamente como paso por referencia, por lo que cualquier modificación dentro de una función tendrá impacto en el objeto real.

“Scope” o ámbito.

```
#include<stdio.h>
void nada();
int i = 0;
Int main()
{
        int i = 6;
        nada();
        printf(“%d”,i);
        nada();
}
void nada()
{
        printf(“%d”,i);
        i++;
}
```

Practica 5

- Implementa de manera recursiva el calculo de factorial.
- 5! = 5x4x3x2….
- Por favor, no copies y pegues de internet.

- Recursividad no es mas que una función que se llama a si misma hasta que se cumpla una condición de paro.

```
int main()
{
        imprime(3);
}
void imprime(int i)
{
        imprime(i--);
        if(i==0)
                return;
        printf("%d", i);
}
```