

Introducción a los lenguajes de programación

Profesor: Victor Valenzuela

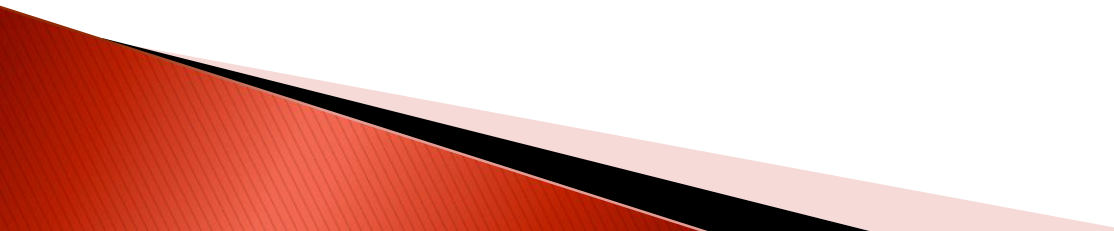
Tipos de lenguajes

- ▶ Existen 2 clases de lenguajes existentes:
 - Lenguajes compilados (fortran, c, pascal, c++, cobol...)
 - Lenguajes interpretados (basic, python, perl, ruby, php)
- ▶ Hay casos especiales como java, que tiene un poco de ambos tipos. Además su caso es especial por que usa una Maquina virtual.

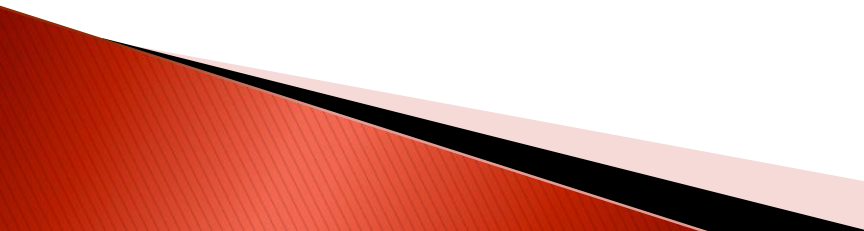
Maquina virtual de Java

- ▶ La máquina virtual de java es una **máquina virtual de proceso nativo**, es decir, ejecutable en una plataforma específica, capaz de interpretar y ejecutar instrucciones expresadas en un código binario especial (el Java bytecode), el cual es generado por el compilador del lenguaje Java.

Nota:

- ▶ Lenguaje ensamblador: Lenguaje de bajo nivel que trabaja de la manera más directa posible la máquina.
 - ▶ Lenguaje de máquinas: el lenguaje que utiliza la máquina para realizar instrucciones.
- 

Lenguajes a comparar...

- ▶ Perl: lenguaje interpretado de propósito general, orientado a objetos, muy sencillo en su sintaxis y utilización (similar a c) , y muy usado en web.
 - ▶ Python: lenguaje interpretado de alto nivel, orientado a objetos, basado en la limpieza de sintaxis. **Es indentado, sino no funciona.**
 - ▶ C: lenguaje compilado de medio nivel, de programación estructurada, orientado a la programación de sistemas operativos.
- 

Tips de ejecución

- ▶ C:
 - Para compilar: `gcc -o hola hola.c`
- ▶ Python y Perl:
 - Buscar la ubicación del interprete:
 - `whereis perl`
 - `whereis python`
 - Identificación del interprete en el archivo:
 - `#!/usr/bin/perl`
 - `#!/usr/bin/python`
- ▶ Ejecución en todos los lenguajes
 - Si es lenguaje compilado, el resultado será un archivo ejecutable.
 - Para lenguaje interpretado, se tiene que convertir el archivo fuente a ejecutable.
 - En cualquier caso, se ejecuta con: `./nombre_de_archivo`
 - Las extensiones de los archivos fuente serán: `c`, `py` y `pl`
- ▶ Uso de comando `file` en Linux...

Variables

- ▶ Espacios reservados en la memoria que pueden cambiar de contenido a lo largo de la ejecución de un programa. Una variable corresponde a un área reservada en la memoria principal del ordenador pudiendo ser de **longitud**:
 - Fija.– Cuando el tamaño de la misma no variará a lo largo de la ejecución del programa. Todas las variables tienen longitud fija, salvo algunas excepciones —.
 - Variable.– Cuando el tamaño de la misma puede variar a lo largo de la ejecución. Típicamente colecciones de datos (arrays) o las cadenas.

Tipos de datos

Tipo / lenguaje	C	Perl	Python
Entero	Int entero = 5	\$entero = 5	entero= 5
Coma Flotante	Float flotante = 3.24	\$flotante = 3.24	flotante= 3.24
Caracter	char caracter = 's'	\$caracter = "s"	caracter= 's'
Cadena	char cadena[] = "cadena" <i><u>final de la cadena tiene (\0)</u></i>	\$cadena = "cadena"	cadena='cadena'
Arreglo (Array)	Init: Int array[10] Uso: array[n]=5	Init: @array Uso y llenado: @array = ("yo", 23,"tu",3.45) Uso: \$array[n] = 5	Init: arrayn = zeros(3, Int) (3 elementos tipo int) Uso y llenado: arrayn = array([3,4,5]) Uso: arrayn[n] = 5 <i><u>arrayn = [23,"cosa"] (Lista)</u></i>

Estructuras de ciclo: for

Python: no funciona como los for en otros lenguajes, ya que pasa por los elementos de un elemento iterable (lista o cadena)

```
lista = ["a", "b", "c"]  
for i in lista:  
    print i  
# Iteramos sobre una  
lista, que es iterable.  
Se puede usar range  
para que funcione  
igual
```

C y Perl: son iguales, sólo cambia la forma en que se usan las variables

```
for ($count = 10;  
$count >= 1;  
$count--)  
{ print "$count "; }
```

#se usa printf en c

Estructura de ciclo: while

Python:

```
numero = 0
while numero < 10:
    numero += 1
    print numero,
```

C y Perl:

```
while ($count <
10)
{print "$count";
$count ++; }
```

Condicionales

Python:

```
verdadero = True
if verdadero:
    print "Verdadero"
else:
    print "Falso"
```

C y Perl:

```
$verdadero = 1;
if ($verdadero)
{
    print "Verdadero"
}
else
{
    print "Falso"
}
```

#en Perl elsif() permite
poner más
condiciones posibles
a verificar

Input e impresión Perl

```
print "What is your name?\n";  
$name = <>;  
chomp($name);  
print "Your name is ", $name, "\n";
```

Input e impresión C (caso de un string)

```
#include <stdio.h>
```

```
int main( ) {
```

```
    char str[100];
```

```
    printf( "Enter a value :");
```

```
    gets( str );
```

```
    printf( "\nYou entered: ");
```

```
    puts( str );
```

```
    return 0;
```

```
}
```



Input e impresión python

```
print('Enter your name:')  
x = input()  
print('Hello, ' + x)
```

Ejercicios de prueba:

1. Imprima por pantalla los n° s primos hasta un numero pedido por teclado.
2. Pida n° s por teclado hasta ingresar un n° par (momento en que termina el programa).
3. Muestre por pantalla el factorial de un n° pedido por teclado. n factorial se define en principio como el producto de todos los números enteros positivos desde 1 hasta n .