Instituto Politécnico Nacional
La Técnica al Servicio de la Patria

Unidad Profesional Interdisciplinaria de Ingeniería Campus Tlaxcala UPIIT

Fundamentos de Programación

Esaú Eliezer Escobar Juárez

Ingeniería en Inteligencia Artificial (IIA)

Variables

- Datos que se mantienen en memoria
  - Literal: Valor concreto
  - Variable: puede cambiar de valor (variar)

```
edad = 19; // variable edad y literal 19
```

Las variables deben ser declaradas

¿Qué tipo de dato queremos usar?

Numéricos:

tipo entero: int

tipo decimal: float

Declaración de variables

```
Tipo nombre;
```

```
int cantidad;
float precio;
```

	Memoria	Rango
cantidad	2 bytes	-32768 a 32767
precio	4 bytes	[-3.4e-38 , -3.4e+38] [3.4e-38 , 3.4e+38]

Se reserva espacio suficiente en memoria

LAS VARIABLES SE TIENEN QUE INICIALIZAR

No se deben usar hasta que se les haya dado un valor.

¿Dónde declararlas?

- Antes del primer uso
- Normalmente al principio de la función

Variables

- Asignación de valores (operador =)

- `variable = expresión;`

- `cantidad = 12;  // int`
- `precio = 39.95; //float`
- `total = cantidad * precio; //Asigna 479.4`

- Concordancia de tipos: ~~`cantidad = 12.5`~~

A la izquierda del = siempre va una variable

Operadores

- Aritméticos
  - `+` Suma
  - `-` Resta
  - `*` Multiplicación
  - `/` División
  - `%` Módulo

- Binarios

Operación	Resultado
`3 + 4`	`7`
`2.56 - 3`	`-0.44`
`143 * 2`	`286`
`45.45 / 3`	`15.15`

Precedencia de los operadores

Operadores	Operación	Orden
* / %	Multiplicación, división y módulo	Se evalúan primero y de izquierda a derecha
+ -	Suma y resta	Se evalúan después y de izquierda a derecha

álgebra: $m = \frac{a+b+c+d+e}{5}$

Leng. C: m = (a+b+c+d+e)/5

Evaluar `y = 2 * 5 * 5 + 3 * 5 + 7`

1. `y= 2 * 5 * 5 + 3 * 5 + 7`
2. `y= 10 * 5 + 3 * 5 + 7`
3. `y= 50 + 3 * 5 + 7`
4. `y= 50 + 15 + 7`
5. `y= 65 + 7`
6. `y= 72`

Secuencias de escape

- Al utilizar la función `printf`

Secuencia de escape	Descripción
`\n`	Nueva línea. Coloca el cursor al principio de la siguiente línea.
`\t`	Tabulador horizontal. Mueve el cursor a la siguiente posición del tabulador.
`\a`	Alerta. Suena la campana del sistema.
`\\`	Inserta una diagonal invertida en una cadena.
`\”`	Inserta unas comillas en una cadena.

Ejemplo:

```
printf("Hola mundo\nBienvenidos");
```

Secuencias de escape

- Al utilizar la función `printf`

Control de formato	Descripción
`%d`	Especifica que se va a mostrar un valor entero.
`%f`	Especifica que se va a mostrar un valor con decimales.

Ejemplo:

```
printf("Cantidad: %d",cantidad);
```

Dev C++ 01

Entrada por teclado

```
scanf("%d", &variable);
```

- Transforma los caracteres introducidos en datos
- Cursor parpadeante, la entrada termina con intro.
- SIEMPRE tendrá referencia a una variable.

Toma 2 argumentos(valores)

1. Cadena de caracteres de control de formato que indica el tipo de dato. Usa % especificador de conversión.
2. Operador de dirección & y el nombre de una variable

Entrada por teclado

Lectura de valores enteros (int)

- Leer dígitos hasta encontrar un carácter que no lo sea

  `12abc↲` <- Asigna 12 a la variable

  El resto queda pendiente para la siguiente lectura

Lectura de valores reales (float)

- Se leen dígitos, el punto decimal y otros dígitos

  `39.95abc↲` <- Asigna 39.95 a la variable

  El resto queda pendiente.

Entrada por teclado

¿Qué pasa si el usuario se equivoca?

- El dato no será correcto.
- Aplicación profesional: código de comprobación y ayuda.
- Nosotros supondremos que los usuarios no se equivocan.
- Algunas veces haremos comprobaciones sencillas.

Para evitar errores, lee cada dato en una instrucción aparte.

Identificadores

- Es el nombre de un dato
  - Para variables y constantes
  - Deben ser descriptivos

- Ejemplo de variables

```
Cantidad  precio  total  base  altura area
```

- Ejemplo de constantes (primer letra en mayúsculas)

```
Pi  Euler
```

Utilice identificadores de menos de 32 caracteres.

¡No usar eñes, ni vocales acentuadas!

Identificadores

- Palabras reservadas

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Identificadores

- ¿Qué identificadores son válidos?

```
balance                 interesAnual
_base_imposible         años
EDAD12                  salario_1_mes
__edad                  cálculoNómina
Valor%00                AlgunValor
100caracteres           valor?
_12_meses               __valor
```

Dev C++ 02

Tipos de datos

- Tipos
  - Cada dato, de un tipo concreto
  - Cada tipo establece:
    - El conjunto (intervalo) de valores válidos
    - El conjunto de operaciones que se pueden realizar
- Expresiones con datos de distintos tipos (compatibles): Transformación automática de tipos (promoción de tipo)

Tipos de datos

- int

  Números enteros (sin decimales) 1363, -12, 49
- float

  Números reales 12.45, -3.1932, 1.16E+02
- double

  Números reales (mayores intervalo y precisión)
- char

  Caracteres 'a' , '{', '\t'
- bool

  Valores lógicos (verdadero/falso) true, false
- void

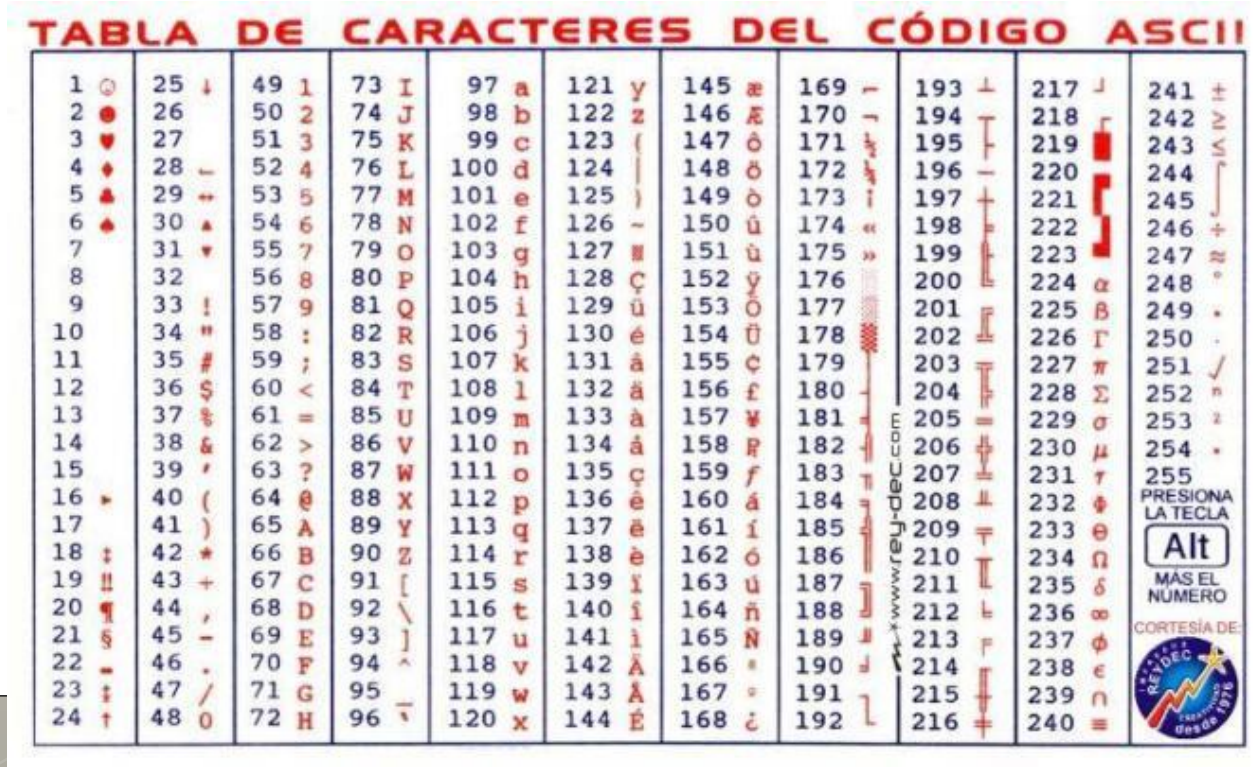
  Nada, ausencia de tipo, ausencia de dato (*funciones*)

char

- 1 byte
- Intervalo de valores: caracteres ASCII
- Literales:

```
'a'   '%'   '\t'
```

secuencias de escape son un solo carácter



bool

- Sólo dos valores
  - Verdadero (*true*)
  - Falso (false)

- Literales:

```
true    false
```

- Cualquier número distinto de cero es equivalente a `true`
- El cero es equivalente a `false`

Dev C++ 03

Modificadores de tipos

- signed / unsigned : con signo (por defecto) / sin signo
- short / long : menor / mayor intervalo de valores

Tipo	Intervalo
Int	-32768..32767
unsigned int	0 .. 65535
short int	-32768 .. 32768
unsigned short int	0 .. 65535
long int	-2147483648 .. 2147483647
unsigned long int	0 .. 4294967295
double	+|- 1.7e-308 .. 1.79e+308
long double	+|- 3.37E-4932 .. 1.18E+4932

Operaciones sobre tipos

Cada tipo determina las operaciones posibles

Tipos de datos numéricos (int, float y double):

- Asignación (=)
- Operadores aritméticos
- Operadores relacionales (menor, mayor, igual, ...)

Tipo de datos bool:

- Asignación (=)
- Operadores lógicos (Y, O, NO)

Tipos de datos char y string:

- Asignación (=)
- Operadores relacionales (menor, mayor, igual, ...)

Operaciones sobre tipos

Operaciones para tipos de datos numéricos

Operador	Operandos	Posición	`int`	`float/double`
-	1 (monario)	Prefijo	Cambio de signo	
+	2 (binario)	Infijo	Suma	
-	2 (binario)	Infijo	Resta	
*	2 (binario)	Infijo	Producto	
/	2 (binario)	Infijo	Div. Entera	Div, real
%	2 (binario)	Infijo	Módulo	No aplica
++	1 (monario)	Prefijo y postfijo	Incremento	
--	1 (monario)	Prefijo y postfijo	Decremento	

Operaciones aritméticos

- *Operadores monarios y operadores binarios*
  - Operadores monarios (*unarios*)
    - Cambio de signo (-):

      Delante de variable, constante o expresión entre paréntesis

      ```
      -saldo   -RATIO   -(3 * a - b)
      ```
    - Incremento/decremento (sólo variables) (prefijo/postfijo):

      ```
      ++interes --meses j++ // 1 más ó 1 menos
      ```
  - Operadores binarios
    - Operando izquierdo operador operando derecho

      ```
      2+3     a*RATIO     -a+b     (a%b)*(c/d)
      ```

Operaciones aritméticos

- *Operadores de incremento y decremento*

  Incremento/decremento de la variable numérica en una unidad

  - Prefijo: Antes de acceder

```
int i = 10, j;
j = ++i; // Incrementa antes de copiar
printf("%d - %d",i,j); //Muestra 11 - 11
```

i=i+1;
j=i;

  - Postfijo: Después de acceder

```
int i = 10, j;
j = i++; // Copia y después incrementa
printf("%d - %d",i,j); //Muestra 11 - 10
```

j=i;
i=i+1;

No mezcles ++ y - - con otros operadores

Precedencia

Precedencia	Operadores
Mayor prioridad	`++  - - (postfijos)`
	`++  - - (prefijos)` `- (cambio de signo`
	`* / %`
Menor prioridad	`+  -`

Dev C++ 04

Abreviaturas aritméticas

variable operador= op_derecho;

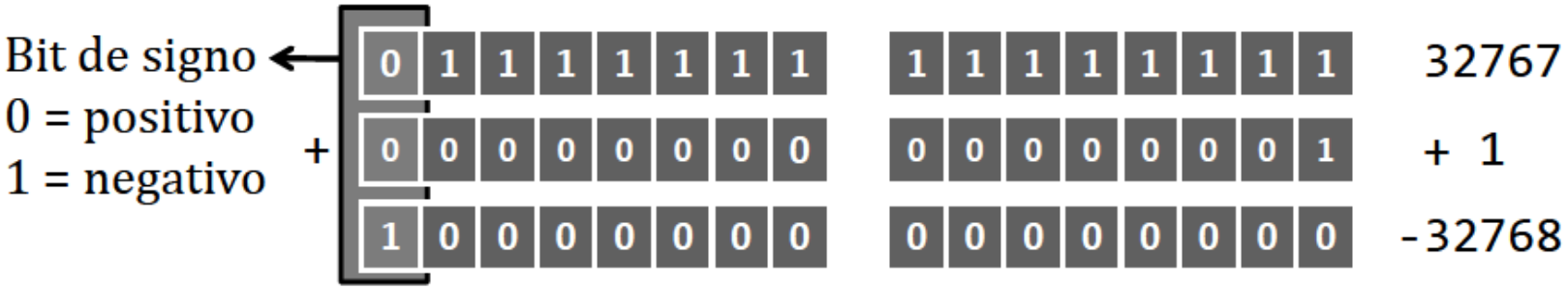
Asignación	Abreviatura
`a = a + 12;`	`a += 12;`
`a = a * 3;`	`a *= 3;`
`a = a - 5;`	`a -= 5;`
`a = a / 37;`	`a /= 37;`
`a = a % b;`	`a %= b;`

Igual precedencia que la asignación.

Desbordamiento

- ¿Valor siguiente al máximo?
  - Valor mayor del máximo (o menor del mínimo) del tipo.

```
short int i = 32767; // Valor máximo para short int
i++; // 32768 no cabe en un short int
printf("%d",i); // Muestra -32768
```



Constantes

- Declaración de constantes
  - Modificador de acceso const
    - Variables incializadas a las que no dejamos variar

```
const int Meses = 12;
const double Pi = 3.141592, Euler = 2.71828182;
```

- La constante no podrá volver a aparecer a la izquierda de un =

¿Por qué usar constantes con nombre?

- Aumentan la legibilidad del código

```
cambioPoblacion = (0.1758 - 0.1257) * poblacion;
cambioPoblacion = (RatioNacimientos - RatioMuertes) * poblacion;
```

- Facilitan la modificación del código

```
double compra1 = bruto1 * 18 / 100;
double compra2 = bruto2 * 18 / 100;
double total = compra1 + compra2;
printf("%f (IVA: 18%%\n",total);

const int IVA = 18;
double compra1 = bruto1 * IVA / 100;
double compra2 = bruto2 * IVA / 100;
double total = compra1 + compra2;
printf("%f (IVA: %d%%\n",total,IVA);
```

3 cambios

↑

Cambiar el IVA al 21%

↓

1 cambio

Funciones matemáticas

- Algunas…

abs(x)	Valor absoluto de x
pow(x, y)	x elevado a y
sqrt(x)	Raíz cuadrada de x
ceil(x)	Menor entero que es mayor o igual que x
floor(x)	Mayor entero que es menor o igual que x
exp(x)	e^x
log(x)	Ln x (logaritmo natural de x)
log10(x)	Logaritmo en base 10 de x
sin(x)	Seno de x
cos(x)	Coseno de x
tan(x)	Tangente de x
round(x)	Redondeo al entero más próximo
trunc(x)	Pérdida de la parte decimal (entero)

Biblioteca math

$$f(x,y) = 2x^5 + \frac{\sqrt{\frac{x^3}{y^2}}}{|x \times y|} - \cos(y)$$