TiposdeVariables

PROFESOR. JOSÉ BLANCAS

Palabras reservadas

Aunque antes se han dado una serie de restricciones sobre cuáles son los nombres válidos que se pueden dar en C++ a los identificadores, falta todavía por dar una: los siguientes nombres no son válidos como identificadores ya que tienen un significado especial en el lenguaje:

fuente:C++ keywords - cppreference.com

A - C	D - P	R - Z
alignas (C++11)	decltype (C++11)	reflexpr (reflection TS)
alignof (C++11)	default (1)	register (2)
and	delete (1)	reinterpret_cast
and_eq	do	requires (C++20)
asm	double	return
atomic_cancel (TM TS)	dynamic_cast	short
atomic_commit (TM TS)	else	signed
atomic_noexcept (TM TS)	enum (1)	sizeof (1)
auto (1) (2) (3) (4)	explicit	static
bitand	export (1) (3)	static_assert (C++11)
bitor	extern (1)	static_cast
bool	false	struct (1)
break	float	switch
case	for (1)	synchronized (TM TS)
catch	friend	template
char	goto	this (4)
char8_t (C++20)	if (2) (4)	thread_local (C++11)
char16_t (C++11)	inline (1)	throw
char32_t (C++11)	int	true
class (1)	long	try
compl	mutable (1)	typedef
concept (C++20)	namespace	typeid
const	new	typename
consteval (C++20)	noexcept (C++11)	union
constexpr (C++11)	not	unsigned
constinit (C++20)	not_eq	using (1)
const_cast	nullptr (C++11)	virtual
continue	operator (4)	void
co_await (C++20)	or	volatile
co_return (C++20)	or_eq	wchar_t
co_yield (C++20)	private (3)	while
	protected	xor
	public	xor_eq

4.2 Comentarios en el código

Comentarios :

Un comentario es texto cuyo contenido es, por defecto, completamente ignorado por el compilador. En C# hay dos formas de escribir comentarios.

1. La primera consiste en encerrar todo el texto que se desee (incluyendo varias líneas) comentar entre caracteres **/* y */** siguiendo la siguiente sintaxis:
/*<texto>*/ No es posible anidar comentarios de este tipo.

2. C++ ofrece una sintaxis alternativa más compacta para la escritura este tipo de comentarios en las que se considera como indicador del comienzo del comentario la pareja de caracteres **//** y como indicador de su final el fin de línea. Por tanto, la sintaxis que siguen estos comentarios es:
// <texto> Pueden anidarse sin ningún problema.

4.4 Estructuras de salida

Método: cout

Namespace: std

Salida, escribe los datos especificados, seguido de un retorno, a la salida estándar del lenguaje. La variante es si necesitas retorno usar << endl si no se quiere el retorno.

Formato:

```
cout << "string";
```

Ejemplos:

```
cout << "Esto es un string";
cout << "Hola Mundo" << endl;
```

3.1 Tipo de Variable

"Los tipos de variable se asocian a la identidad del valor de dicha variable a lo largo del problema estas identidades se dividen en cuatro tipos básicos: numérico, texto o alfabético, lógico, carácter"

Ejemplo de un tipo de variable:

Declarar variable 'variable_uno' como un número entero (integer ó int). /* El valor de esta variable solo podrá tomar valores enteros a lo largo de la ejecución del programa

Tipos de variables

1. Tipos de datos básicos (primitivos):
   a) Enteros: int, short, long (unsigned)
   b) Fraccionarios: double, float
   c) Lógicos: bool
   d) Texto o caracteres: string, char
2. Tipos de datos complejos:
   a) Arreglos: arrays, vectors
   b) Estructuras: struct
   c) Objetos: class, objects

Enteros

Representa un número entero, son valores simples, que representan un valor numérico. fuente:Data Type Ranges | Microsoft Learn

Type Name	Bytes	Other Names	Range of Values
`int`	4	`signed`	-2,147,483,648 to 2,147,483,647
`unsigned int`	4	`unsigned`	0 to 4,294,967,295
`__int8`	1	`char`	-128 to 127
`unsigned __int8`	1	`unsigned char`	0 to 255
`__int16`	2	`short`, `short int`, `signed short int`	-32,768 to 32,767
`unsigned __int16`	2	`unsigned short`, `unsigned short int`	0 to 65,535
`__int32`	4	`signed`, `signed int`, `int`	-2,147,483,648 to 2,147,483,647
`unsigned __int32`	4	`unsigned`, `unsigned int`	0 to 4,294,967,295
`__int64`	8	`long long`, `signed long long`	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
`unsigned __int64`	8	`unsigned long long`	0 to 18,446,744,073,709,551,615

¿Como se declaran?

Se debe usar un formato estándar:

<tipo de dato> nombre_variable;

```
char num_char = -129;
cout << "Ejemplo numero char : " << int(num_char) << endl;
int num_int = 5;
cout << "Ejemplo numero entero : " << num_int << endl;
```

¿Como se alimentan desde el programa?

Se debe usar un formato estándar:

nombre_variable = <valor>;

¿Como se alimentan desde la consola?

Solicitar un valor al usuario:

```
int edad;
cout << "Dame tu edad : ";
cin >> edad;
cout << endl;
cout << "Tu edad es : " << edad << endl;
```

cin : Es un método de la librería std, para leer un conjunto de caracters desde la consola.

NOTA: Por default, la consola acepta cualquier valor de entrada, es decir una cadena de caracteres, esto aplica a C++, Java, C, C#.

Como se imprimen en consola?

Imprimir en consola :

```
cout << "Tu edad es : " << edad << endl;
```

<< es el medio para concatenar o unir tanto texto fijo “ “, como variables e incluso dar retorno dentro de la misma línea.

```
double numeroPI = 3.141632;
cout << fixed;
cout << setprecision(4);
cout << "Numero PI : " << numeroPI << endl;
```

Si requieres imprimir decimales en un número con punto decimal, debe usarse una librería adicional llamada <iomanip>, declarar fixed y poner la precisión con la función setprecision(# de decimales), como se muestra arriba.

Números con decimal

Representa un número entero + su valor decimal, son valores simples, que representan un valor numérico.

C# supports the following predefined floating-point types:

C# type/keyword	Approximate range	Precision	Size	.NET type
`float`	$\pm 1.5 \times 10^{-45}$ to $\pm 3.4 \times 10^{38}$	~6-9 digits	4 bytes	System.Single
`double`	$\pm 5.0 \times 10^{-324}$ to $\pm 1.7 \times 10^{308}$	~15-17 digits	8 bytes	System.Double
`decimal`	$\pm 1.0 \times 10^{-28}$ to $\pm 7.9228 \times 10^{28}$	28-29 digits	16 bytes	System.Decimal

Como se declaran?

Se debe usar un formato estándar:

<tipo de dato> nombre_variable;

```
float varFloat;
double varDouble;
decimal varDecimal;
```

NOTA: Precisión Aritmética

Se refiere al número de dígitos utilizados para representar un número. Por ejemplo, el número 6.34 es menos preciso aritméticamente que el número 6.3423342

Como se alimentan desde el programa?

Se debe usar un formato estándar:

nombre_variable = <valor>;

```
varFloat = 6.342334290f;
varDouble = 6.342334290d;
varDecimal = 6.342334290m;
```

Como se alimentan desde la consola?

Solicitar un valor al usuario:

```
float numero_convertir;
cout << "Dame el numero a convertir : ";
cin >> numero_convertir;
cout << "Ejemplo numero entero : " << numero_convertir << en
```

NOTA: Por default, la consola acepta cualquier valor de entrada, es decir una cadena de caracteres, esto aplica a C#, C++, Java.

Como se imprimen en consola?

Imprimir en consola :

```
cout << "Ejemplo numero entero : " << numero_convertir << end
```

Recuerda que si requieres imprimir decimales en un número con punto decimal, debe usarse una librería adicional llamada <iomanip>, declarar fixed y poner la precisión con la función setprecision(# de decimales), como se muestra arriba.

Concepto Básico #10

Constante

"Un valor constante no puede ser alterado durante la ejecución del programa"

Ejemplo de una constante:

El valor de la constante 'PI' es 3.1416, siempre.

Como se declaran?

Se debe usar un formato estándar:

const <tipo de dato> nombre_variable = <valor>;

```
const double pi = 3.14163264;
const int velocidad_luz = 3000000; // km por segundo
```

NOTA:

La palabra previa al tipo de variable, es también una palabra reservada, adicionalmente se puede declarar pública o privada. Estas nunca se solicitan al usuario, y se imprimen exactamente igual que enteros o fraccionarios.

Textos (string)

Un “string” es un tipo de variable que contiene valores de texto, que se guarda como una secuencia ordenada de caracteres

Como se declaran?

Se debe usar un formato estándar:

<tipo de dato> nombre_variable;

```
string nombre;
string apellido_paterno;
string apellido_materno;
string numero_cuenta;
```

NOTA:

Un string es como se lee casi cualquier dato de entrada, no importa si son números, valores lógicos u letras, la manera de viajar de estos datos en internet es de la manera más simple, que es un texto de código tipo ASCII, HTML o Unicode, este dos últimos los más usados.

Como se alimentan desde el programa?

Se debe usar un formato estándar:

nombre_variable = <valor>;

```
nombre = "Jose Luis";
apellido_paterno = "Blancas";
apellido_materno = "Ruiz";
numero_cuenta = "1234567-8";
```

Nota:

Se usan dobles comillas, no se puede usar comilla sencilla, porque esta última esta reservada para un solo carácter.

Como se imprimen en consola?

Imprimir en consola :

```
cout << nombre << " " << apellido_paterno << " " << numero_cuenta;
```

NOTA: El uso de del símbolo << se usa para “unir” texto con valores de variables y poder dar mejor contexto al valor de las variables.

Como se alimentan desde la consola?

Solicitar un valor al usuario:

```
cout << "Dame el nombre ";
getline(cin, nombre);
cout << endl;
cout << "Dame el apellido paterno ";
cin >> apellido_paterno;
cout << endl;
cout << "Dame el apellido materno ";
cin >> apellido_materno;
cout << endl;
```

NOTA: Cuando necesitas capturar varias palabras, es decir, en este caso dos nombres, debes añadir la biblioteca <string> y usar la función getline como se muestra en este ejemplo.

Booleanos/Lógicos (bool)

La palabra clave de tipo bool es un alias para el tipo de estructura de .NET System.Boolean que representa un valor booleano que puede ser true o false.

Para realizar operaciones lógicas con valores del tipo bool, use operadores lógicos booleanos. El tipo bool es el tipo de resultado de los operadores de comparación e igualdad. Una expresión bool puede ser una expresión condicional de control en las instrucciones if, do, while y for, así como en el operador condicional ?:.

Como se declaran?

Se debe usar un formato estándar:

<tipo de dato> nombre_variable;

NOTA:

El valor por default de la variable será indefinida, por lo que se necesita asignar un valor manualmente el valor de la variable para convertirlo a verdadero, la mejor analogía sería como un semáforo de dos colores, verde y rojo.

Como se alimentan desde programa?

Solicitar un valor al usuario:

```
checado = true;
```

NOTA:

No se suele preguntar en consola por un valor lógico, por lo que no se solicita casi nunca el valor.

Como se imprimen en consola?

Imprimir en consola :

```
cout << checado << endl;
```

NOTA: Normalmente esto no se hace en los programas, ya que sirven más para control de flujo de datos.