Variables y constantes

- Permiten almacenar y representar valores.
- Se pueden utilizar en cualquier ámbito (función, método, estructura)
- Su tiempo de vida depende del ámbito donde fueron declarados.
- Las letras mayúsculas y minúsculas son diferentes.

Variables

- El valor almacenado puede ser mutable o inmutable
- Permite la *inferencia del tipo de dato*, es decir el compilador deducirá el tipo de dato a partir del valor asignado.

Declarar una variable

```
    let persona_string = "Rodrigo Mendoza";  // variable tipo string
    let edad_integer = 32;
    let calificacion_float = 14.3;           // variable tipo float
    let es_mayor_edad = true;           // variable tipo boolean
    let icono_char = 'Ⓐ';                    // variable tipo caracter unicode
    println!("nombre: {}",persona_string);
    println!("edad: {}",edad_integer);
    println!("calificación: {}",calificacion_float);
    println!("mayor de edad: {}",es_mayor_edad);
    println!("icono: {}",icono_char);
```

- Para declarar una variable utilice el prefijo **let**.

Reglas para nombrar una variable

- Debe estar compuesto de letras, guión bajo o dígitos
- Debe comenzar con una letra o un guión bajo.

Inmutabilidad de variables

```
let x = 5;
println!("El valor de x es: {}", x);
x = 6; //--> ERROR, la variable inmutable no puede ser asignada 2 veces
println!("El valor de x es: {}", x);
```

- La **inmutabilidad** no permite el cambio del valor de una variable, es decir no se puede cambiar su valor una vez que su valor está vinculado al nombre de la variable.
- Las variables por defecto son inmutables.

Ventajas

- Incrementa la seguridad y concurrencia
- Son recomendables para estructuras de datos pequeñas porque ofrecen mayor claridad

Desventajas

- Reduce el rendimiento en estructuras de datos grandes, porque cada vez que cambie su valor se debe crear una nueva instancia y asignarle un nuevo valor

Mutabilidad de variables

```
let mut x = 5;
println!("El valor de x es: {}", x);
x = 6;
println!("El valor de x es: {}", x);
```

- La **mutabilidad** permite el cambio de valor de una variable.
- Para hacer a una variable mutable, prefije el nombre de la variable con la palabra clave **mut**.

Ventajas

- Mejora el rendimiento en estructuras de datos grandes, porque cada vez que se cambie su valor no se debe crear una nueva instancia solo se debe reasignar con un nuevo valor

Desventajas

- Incrementa la complejidad en estructuras de datos pequeñas

Shadowing de variables

```
let saludo=8014 4 7005;
let saludo="Hola a todos";
println!("saludo: {}",saludo);
```

- El **shadowing** permite reutilizar nombres de variables declaradas anteriormente, es decir la nueva variable oculta a la anterior, *pero internamente se crea una nueva variable*.
- Es utilizado para convertir valores entre diferentes tipos.

Constantes

- El valor almacenado es inmutable
- Es obligatorio declarar el tipo de dato, no se permite la *inferencia del tipo de dato*.

Declarar una constante

```
    const LIMITE_USUARIOS:i32 =100;
    const PI:f32=3.14;

    println!("limite de usuarios: {}",LIMITE_USUARIOS);
    println!("valor de PI: {}",PI);
```

- Para declarar una constante utilice el prefijo **const** y declare el tipo de dato.

Reglas para nombrar una constante

- Todos los caracteres alfabéticos de la constante deben estar en mayúsculas.
- Debe estar compuesto de letras, guión bajo o dígitos.
- Debe comenzar con una letra o un guión bajo.