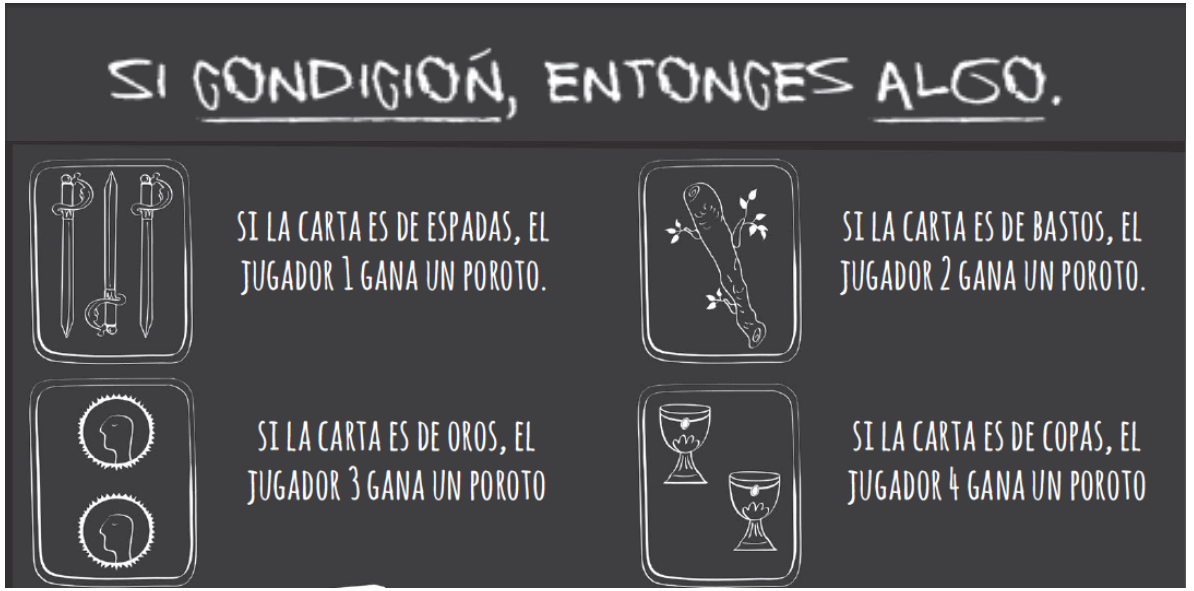
PROGRAMACIÓN ESTRUCTURADA

UNIDAD 3

Estructura de decisión

Estructuras de decisión y condicionales

- Permiten expresar que algo debe suceder bajo ciertas circunstancias.
- Una condición es un enunciado que, o bien es cierto, o bien es falso. Siempre posee un valor de verdad.
- Todos los lenguajes de programación tienen construcciones que nos permiten evaluar condiciones y realizar ciertas acciones dependiendo de si lo evaluado es verdadero o falso.
- Permite incorporar a nuestros programas la toma de decisiones

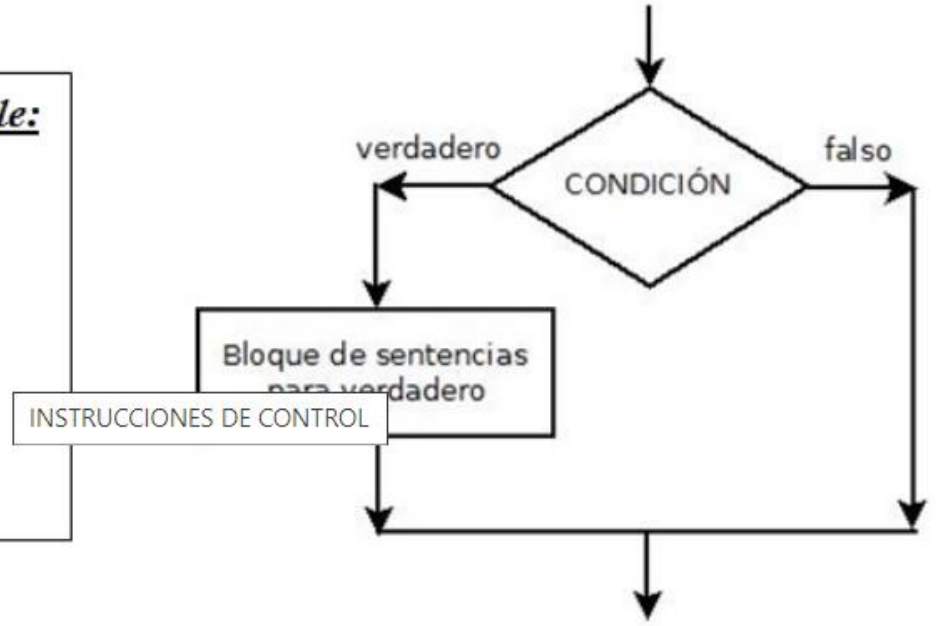


Alternativa simple

- Se evalúa una condición, ejecutándose un grupo de sentencias si el resultado es «verdadero», y no ejecutándose este grupo de sentencias si el resultado es «falso».

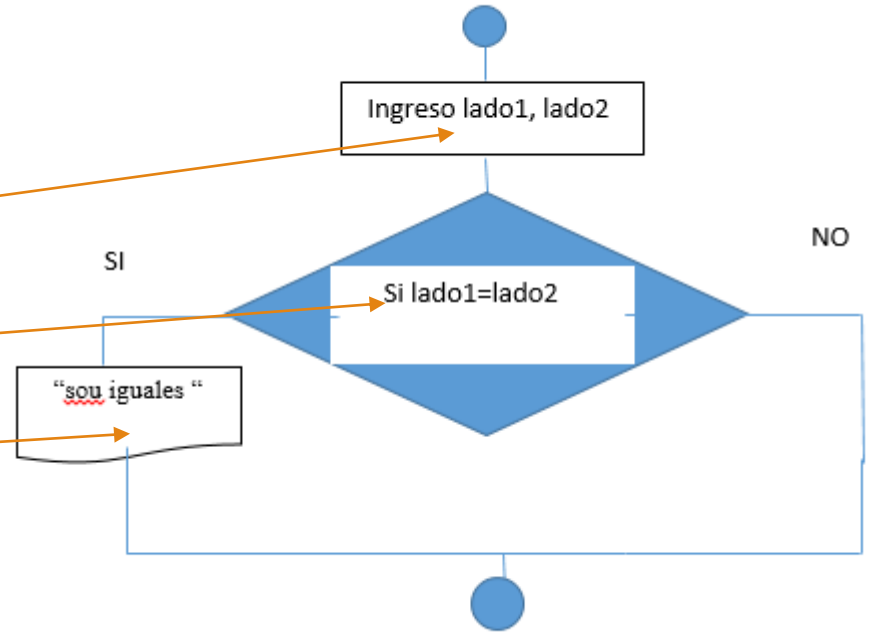
Pseudocódigo de la alternativa simple:

Si *condición*
sentencia 1
sentencia 2
. . .
sentencia n
FinSi



Alternativa simple

```
1   #include <stdio.h>
2   #include <conio.h>
3   #include<stdlib.h>
4   #include<iostream>
5
6   int main()
7   {
8   int lado1,lado2;
9   sytem("cls");
10  printf("ingrese el primer lado  ");
11  scanf("%d",&lado1);
12  printf("ingrese el segundo lado  ");
13  scanf("%d", &lado2);
14  if(lado1= =lado2)
15      {
16      printf("Los lados son iguales \n");
17      }
18  system("pause");
19  }
20
```

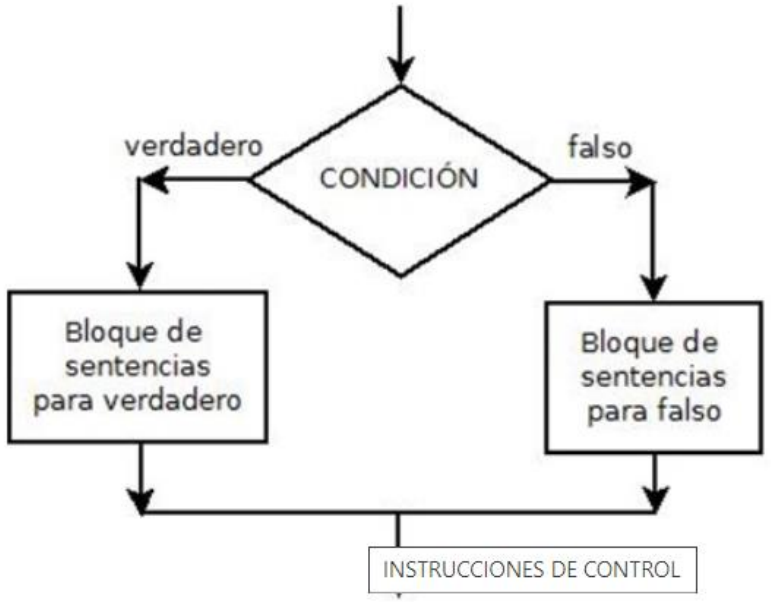


Alternativa doble

- Se evalúa la condición, ejecutándose un grupo de sentencias si el resultado es «verdadero», y ejecutándose otro grupo alternativo de sentencias si el resultado es «falso».

Pseudocódigo de la alternativa doble:

Si *condición*
    sentencia v1
    sentencia v2
    . . .
    sentencia vn
Sino
    sentencia f1
    sentencia f2
    . . .
    sentencia fm
FinSi



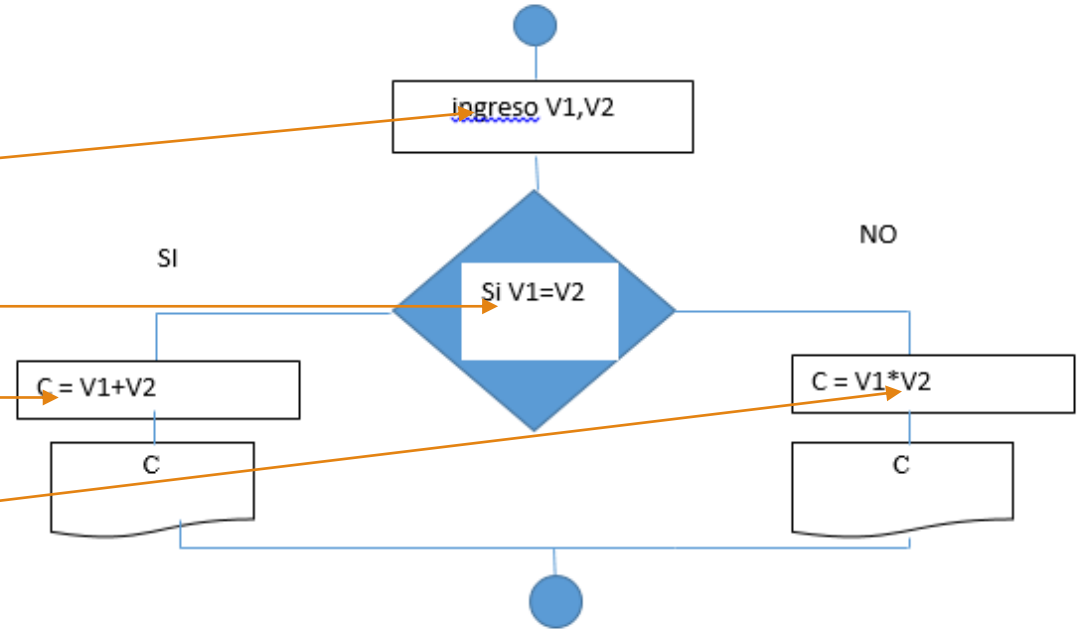
Alternativa doble

```c
#include <stdio.h>
#include <conio.h>
#include<stdlib.h>
#include<iostream>

int main()
{
int V1,V2,c;
system("cls");
printf("ingrese el primer valor \n");
scanf("%d", &V1);
printf("ingrese el segundo valor \n");
scanf("%d", &V2);
if(V1==V2)
    {
    c=V1+V2;
    printf("son iguales y la suma es %d \n",c);
    }

    else
    {
    c=V1*V2;
    printf("son distintos y el producto es %d \n",c);
    }
    system("pause");

}
```

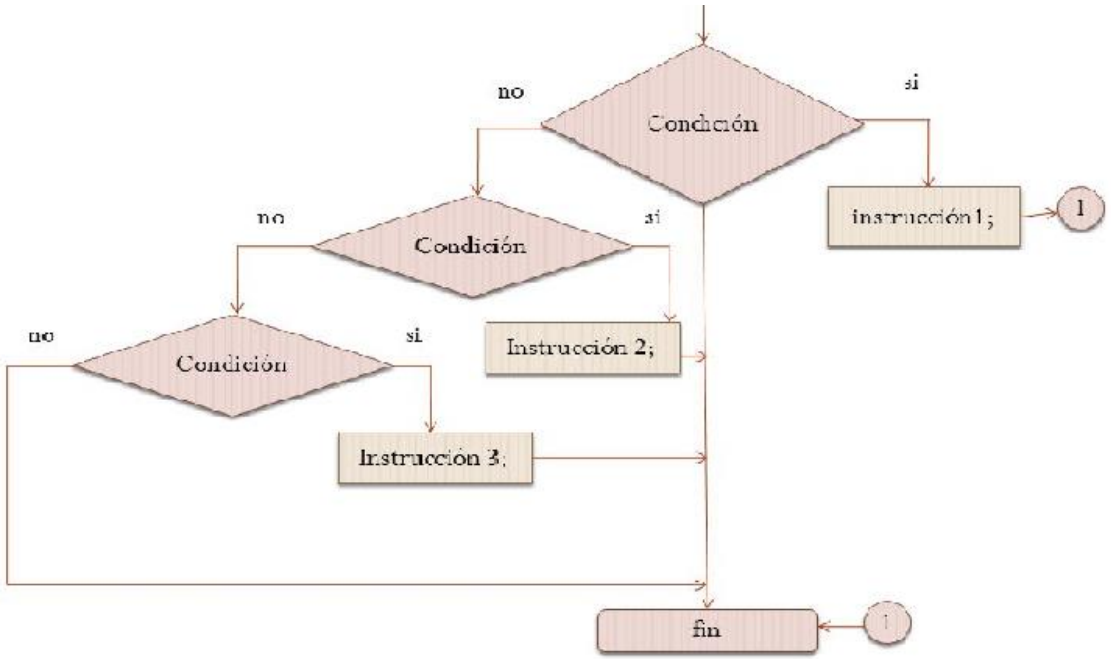


Alternativas anidadas

- También es posible utilizar la instrucción Si-Sino para diseñar estructuras de selecciones entre más de dos alternativas. Esto se consigue mediante las estructuras anidadas, donde tanto la rama Si como la Sino pueden contener a su vez otra instrucción Si-Sino, y así sucesivamente un número determinado de veces.

Ejemplo de estructuras Si-Sino anidadas:

```
Si condición1
    bloque de sentencias 1v
Sino
    Si condición 2
        bloque de sentencias 2v
    Sino
        Si condición 3
            bloque de sentencias 3v
        Sino
            bloque de sentencias 3f
        FinSi
    FinSi
FinSi
```



Alternativas anidadas

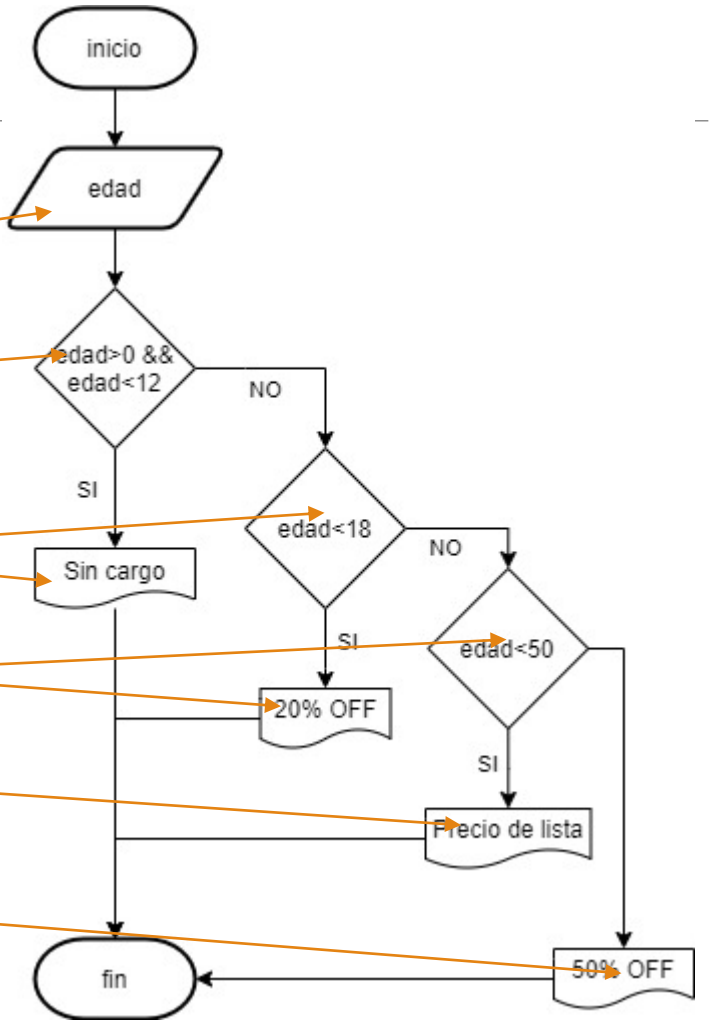
```c
int main(int argc, char** argv) {



    int edad;
    printf("Ingrese su edad: ");
    scanf("%d",&edad);


    if (edad>0 && edad <12){
        printf("Entrada sin cargo\n");
    }
    else
    {
        if (edad<18){
        printf("Entrada con 20%% de descuento\n");
        }
        else{
            if (edad<50){
                printf("Entrada precio de lista");
            }
            else
            {
                printf("Entrada 50%% de descuento");
            }
        }

    }
}
```

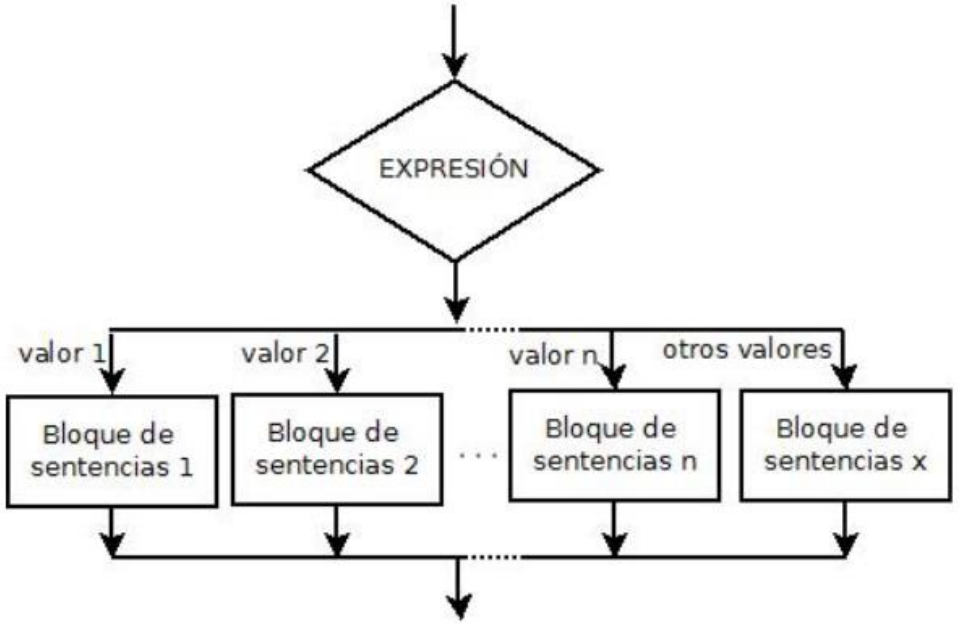


Alternativa múltiple

- En lugar de una condición, se evalúa una expresión con múltiples pero finitos resultados, ejecutándose en función del resultado de la expresión, un grupo de sentencias entre múltiples posibles

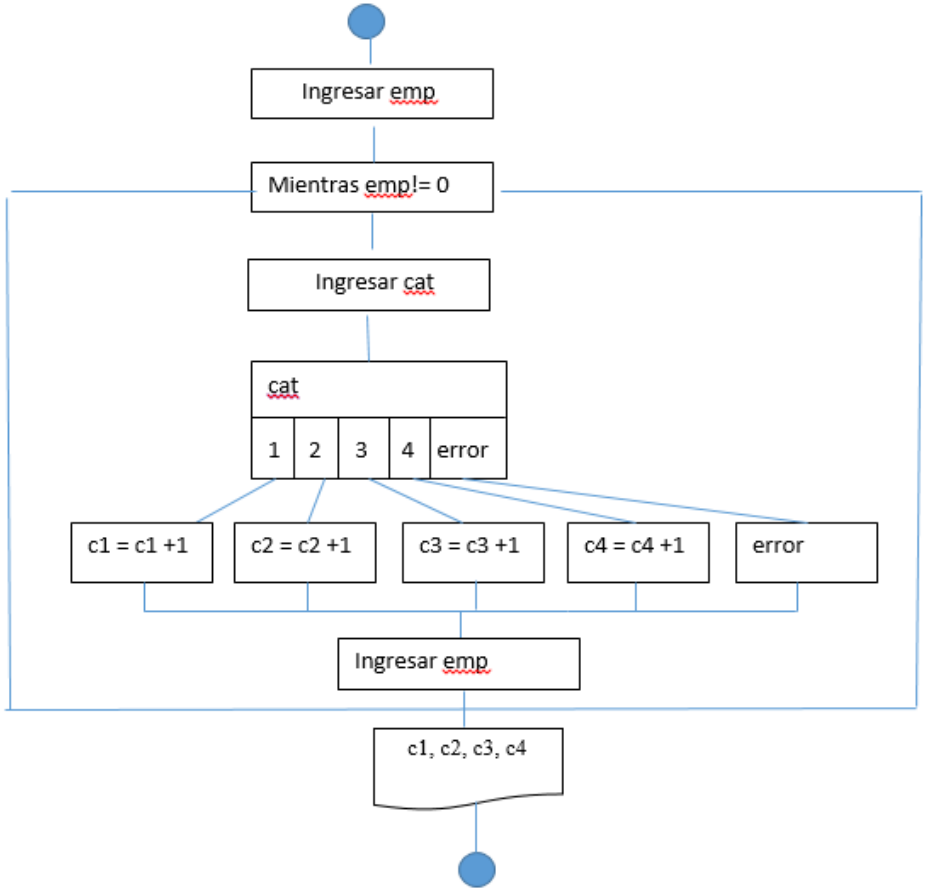
Pseudocódigo de la alternativa múltiple:

Según ***valor*** *expresión* ***hacer***
Valor_1*:*
bloque de sentencias 1
Valor_2*:*
bloque de sentencias 2
. . .
Valor_n*:*
bloque de sentencias n
Sino*:*
bloque de sentencias x
FinSegún_valor



Alternativa múltiple

```
int main ()
{
int emp,cat,c1,c2,c3,c4;
c1=c2=c3=c4=0;
system("cls");
printf("ingresar el nro de empleado   ");
scanf("%d",&emp);
while (emp!=0)
{
    printf("ingrese la categoría  ");
    scanf("%d",&cat);
    switch(cat)
        {
        case 1:  c1=c1+1;
        break;
        case 2:  c2=c2+1;
        break;
        case 3:  c3=c3+1;
        break;
        case 4:  c4=c4+1;
        break;
        default: printf("error de categoría");
        }
    printf("ingresar el nro de empleado   ");
    scanf("%d",&emp);
}
printf("el total de empleados de la categoría 1 es %d \n ",c1);
printf("el total de empleados de la categoría 2 es %d \n",c2);
printf("el total de empleados de la categoría 3 es %d \n",c3);
printf("el total de empleados de la categoría 4 es %d \n",c4);
system("pause");
}
```



Tablas de verdad

Tablas de verdad

- Una tabla de verdad es un diagrama que permite determinar claramente cuando una proposición compuesta es verdadera, falsa o variada.
- UNA PROPOSICIÓN es el producto lógico del acto por el cual se afirma o se niega algo de algo.
- Muestra el valor de verdad de una proposición compuesta, para cada combinación de verdad que se pueda asignar.

Tablas de verdad

- **Verdadero** El valor verdadero se representa con la letra V; si se emplea notación numérica se expresa con un uno: 1; en un circuito eléctrico, el circuito está cerrado.
- **Falso** El valor falso se representa con la letra F; si se emplea notación numérica se expresa con un cero: 0; en un circuito eléctrico, el circuito está abierto.
- **Negación** es el valor opuesto
- **Conjunción** (y) s un operador, que actúa sobre dos valores de verdad, típicamente los valores de verdad de dos proposiciones, devolviendo el valor de verdad verdadero cuando ambas proposiciones son verdaderas, y falso en cualquier otro caso
- **Disyunción** (o) s un operador que actúa sobre dos valores de verdad, típicamente los valores de verdad de dos proposiciones, devolviendo el valor de verdad verdadero cuando una de las proposiciones es verdadera, o cuando ambas lo son, y falso cuando ambas son falsas.

Tablas de verdad

- **NEGACIÓN**
  - Si la proposición ‘A’ es verdadera, su negación (es decir, ‘no A’) será necesariamente falsa. Si la proposición ‘A’ es falsa, entonces su negación será necesariamente verdadera

Negación

A	¬ A
V	F
F	V

Tablas de verdad

- **CONJUNCIÓN**
  - Para que la relación ‘A y B’ sea verdadera, la proposición simple ‘A’ y la proposición simple ‘B’, deben serlo también. Si una de estas proposiciones simples es falsa, entonces la relación conjuntiva es también necesariamente falsa.

A	B	$A \wedge B$
V	V	V
V	F	F
F	V	F
F	F	F

Tablas de verdad

- **DISYUNCIÓN**
  - Para que la relación proposicional ‘A o B’ sea verdadera, basta con que solo una de las proposiciones simples lo sea. De ser ambas proposiciones falsas, entonces la relación de conjunción también lo será

A	B	$A \vee B$
V	V	V
V	F	V
F	V	V
F	F	F