

Types of Errors, Operators, if...else

---



C Programming

Trainer : Smita Kadam

Email ID : [smita@sunbeaminfo.com](mailto:smita@sunbeaminfo.com)



# Using Width for printing data

- `int num = 12;`
- `printf("%4d",num);`

output : \_\_ 12

\_\_\_\_\_

- `int num = 12;`
- `printf("%-4d",num);`

output : 1 2 \_\_

\_\_\_\_\_

- `float fval= 12.48;`
- `printf("%6.2f",fval);`

output : \_ 1 2 . 4 8

\_\_\_\_\_



# Types of errors

## Compile Time Errors

Mainly check of syntax and semantics

## Run Time Errors

Occurs on execution of program e.g. Divide by Zero

## Linker Errors

Occurs when linker tries to search a resource but fails

## Logical Errors

Occurs when mentioned logic do not provide expected output.



# Operators In C

## Arithmetical Operators

+      -      /      \*      %

## Logical Operators

&&      ||      !

## Relational Operators

>      <      >=      <=      ==      !=

## Bitwise Operators

&      |      ^      <<      >>      ~

## Unary Operators

+      -      \*      &      sizeof      ++      --      .      ->

## Shorthand Operators

+=      -=      /=      \*=      %=

&=      |=      ^=      <<=      >>=      ~=

## Conditional Operators

?      :

## (Ternary Operators)

## Special Operators

[]      ( )      ,

## Assignment

=



# Operator Precedence and Associativity

OPERATOR	TYPE	ASSOCIIVITY
() [] . ->		left-to-right
++ -- +- ! ~ (type) * & sizeof	Unary Operator	right-to-left
* / %	Arithmetic Operator	left-to-right
+ -	Arithmetic Operator	left-to-right
<< >>	Shift Operator	left-to-right
< <= > >=	Relational Operator	left-to-right
== !=	Relational Operator	left-to-right
&	Bitwise AND Operator	left-to-right
^	Bitwise EX-OR Operator	left-to-right
	Bitwise OR Operator	left-to-right
&&	Logical AND Operator	left-to-right
	Logical OR Operator	left-to-right
? :	Ternary Conditional Operator	right-to-left
= += -= *= /= %= &= ^=  = <<= >>=	Assignment Operator	right-to-left
,	Comma	left-to-right



## Decision Control – If ..else

Syntax:

```
if (<expression>)  
{  
    <statements>  
} // // executes when expression results true
```

Any expression which results non zero value is considered as true where as zero is considered as false.



## Decision Control – If ..else

Syntax:

```
if (<expression>)  
{  
    <statements>  
} // executes when expression results true  
else  
{  
    <statements>  
} // executes when expression results false
```



# Decision Control – Nested if..

Syntax:

```
if (<expression>)  
{  
    <statements>  
    if (<expression>)  
    {  
        <statements>  
    } // executes when expression results true  
} // executes when expression results true  
else  
{  
    <statements>  
} // executes when expression results false
```





# Decision Control – If ..else if ....

## Syntax:

```
if (<expression>)                                //1.
{
    <statements>
} // executes when expression results true
else if (<expression>)                            //2.
{
    <statements>
} // executes when expression 1 results false
else if (<expression>)                            //3.
{
    <statements>
} // executes when expression 1,2 results false
else                                             //4.
{
    <statements>
} // executes when expression 1,2,3 results false
```



## Conditional Operators      ?      :

### Syntax:

<expression> ? <true> : <false>;

### Points to note:

1. Follows right to left associativity rule
2. Can not use jump statement in true or false part





Thank you!

