03-05-2024

Sizeof():- Unary operator

SYNTAX:- sizeof(type) Unary it gives size in bytes.

EX:-
```
main()
{
printf("%d", sizeof(int));
printf("%d", sizeof(float));
int x;
x = sizeof(char);
printf("%d", x);
}
```
O/P:-

TYPE CASTING:- Converting 1 dtype data type to another data type

TYPE CASTING

IMPLICIT

↓

-> During assignment operation

-> comparing using relational operators

Ex:- 3 == 3.0

-> Sending arguments to function

Ex:- add(2);

add(float a)

{

}

EXPLICIT

SYNTAX:-

(type) operand (unary)

```
main()
{
int a=5, b=2.0;
float c;
c = (float) a / b;
printf("%f", c);
}
```

COMMA & SEMI COLON OPERATORS :-

Comma operator indicates a part of statement is completed, whereas semicolon indicates statement is completed.

C-TOKENS :- A smallest literal in a c language is a 'c' Token.

C Tokens

Key Words	Identifiers	Constants	Operators	String	Special Symbols
32	[Variables & functions]	23	+, -, *, <	"good"	{
int	main	4.5		"1234hai"	}
float	printf	'a'			
char	scanf				
if	sum, avg				
else					
while					
for					

EXAMPLES FOR SIZE OF():-

* int main()

{

printf("size of char is %d byte\n", sizeof(char));

printf("size of int is %d\n", sizeof(int));

printf("size of float is %d\n", sizeof(float));

printf("size of double is %d\n", sizeof(double));

printf("size of long double is %d\n", sizeof(long double));

printf("size of short int is %d bytes\n", sizeof(short int));

printf("size of long int is %d bytes\n", sizeof(long int));

}

O/P:-

Size of char is 1 byte

Size of int is 4 bytes

size of short int is 2 bytes

Size of long int is 4 bytes

size of float is 4 bytes

size of double is 8 bytes

size of long double is 16 bytes

EXAMPLES FOR IMPLICIT TYPE CASTING =>

-> int main()

```
{ int x=10;
  char y='a';
  x = x+y;
  float z = x+1.0;
  printf("x = %d\n z = %f", x, z);
}
```

O/P :-
x = 107
z = 108.000000

-> int main()

```
{ int x=10;
  double y;
  y = x;
  printf("Integer Value : %d\n", x);
  printf("Double value after implicit type casting : %f\n", y);
}
```

O/P :-
Integer Value : 10
Double Value ... : 10.0000

EXAMPLES FOR EXPLICIT TYPE CASTING =>

-> int main()

```
{ double x=3.14; // num_double = x
  int y; // num_int = y
  y = (int) x;
  printf("Double Value : %f\n", x);
  printf("Integer value after explicit type casting :
          %d\n", y);
}
```

O/P :- Double Value : 3.140000
Integer Value ---- casting : 3

IMPLICIT TYPE CASTING:-

-> int main()

{

int num_int = 65;

char char_value;

char_value = num_int;

printf("Integer value: %d\n", num_int);

printf("Character value after implicit type casting: %c\n",

char_value);

}

O/P:- Integer value: 65

Character value ..... : A

-> int main()

{

int num_int;

printf("Enter num_int: ");

scanf("%d", &num_int);

char char_value;

char_value = num_int;

printf("Integer value entered from user: %d\n", num_int);

printf("Character value after implicit type casting is: %c\n", char_value);

}

O/P:- Enter num_int: 122

Integer value entered from user: 122

Character value after implicit type casting : z

-> int main()

{ int num_int;

printf("Enter num_int: ");

scanf("%d", &num_int);

double num_double;

printf("Enter num_double: ");

scanf("%lf", &num_double);

double result = num_int + num_double;

printf("Result of addition after implicit type casting: %f\n",
result);

}

O/P:- Enter num_int: 10

Enter num_double: 3.5

Result of addition after implicit type casting: 13.5000

EXPLICIT TYPE CASTING :-

-> int main()

{ double num_double;

printf("Enter num_double:");

scanf("%lf", &num_double);

int num_int;

printf("Enter num_int:");

scanf("%d", &num_int);

double result = num_double / (double) num_int;

printf("Result for explicit type casting with arithmetic operations:
%f\n", result);

}

O/P:- Enter num_double: 3.5

Enter num_int: 2

Result ----- Operations: 1.7500

C TOKEN - KEYWORDS =>

* auto
* break
* case
* char
* const
* continue
* default
* do
* double
* else
* enum
* extern
* float
* for
* goto
* if
* int
* long
* register
* return
* short
* signed
* sizeof
* static
* struct
* switch
* typedef
* Union
* Unsigned
* Void
* Volatile
* while

C TOKEN - IDENTIFIERS =>

* main
* printf
* scanf

C TOKEN - SPECIAL SYMBOLS =>

* Brakets [ ]
* Parentheses ( )
* Braces { }
* Comma ,
* colon :
* Semicolon ;
* Asterisk *
* Assignment operator (=)
* Pre-processor #
* Period (.)
* Tilde ~

C TOKEN - OPERATORS =>

-> Unary operators

-> Binary operators:

* Arithmetic operators
* Relational operators
* Logical operators
* Assignment operators
* Bitwise operator

-> Ternary operator: The one operator that requires 3 operands to act upon. Ternary operator is also called as conditional operator (?).