Expressions in C

Expression

- Expression returns a value

1 + 2

A + B

I * J + K

- Assignment is also an expression

Expression returns the value of the LHS after assignment is done

A = B;

J = i + (k = c)

Is valid code !

Side effect of this code is k =c

- Conditional expression is also an expression

True/False in C

- C does not have “boolean” data types
- Which means no “True/False” values in C
- Number 0 is treated as false and any non-zero value is treated as True in C
- In the code

If(i == 0) { … }

- The condition *i == 0* evaluates to 0 or 1

```
int i = 5, k;
    k = (i == 6);
 /*  evaluate to 0 or 1 for all relational operators*/
    printf("%d\n", k);
    if(5)
        printf("I am in if");
```

ANY ERROR?

True/False in C

- In the code

```
if ( i = 5) { ... }
```

i = 5 is an expression, which assigns 5 to i, then returns the value of i, that is 5, and 5 is true, so the expression is always true

Evaluation of Expressions

C does not specify an order of evaluation for operands in an expression.

For example : In the expression

$$X = f() + g()$$

It is not specified whether f() is called first or g()

Similarly in the expression

$$X = (a + b) * (c – d)$$

It is not specified whether (a + b) is done first or (c – d)

In the expression

$$A = B + C$$

Whether B is “Fetched” first or “C” is fetched is also not specified !

QUIZ QUESTION

printf(“%d%d%d”, i++, i++, i++);

printf() is a function with 4 arguments

orders of evaluation of these arguments in NOT DEFINED

so of the given options: COMPILER DEPENDENT was most correct option.

Side Effects

If a function code or an expression affects the calling function or other other parts of expression or other parts of code then it is called side effect.

For example

```
int j;
int f(int x )
{   j = 20;
   return x + 20;
}
```

This function, apart from returning x + 20 , also modified global j which is a side effect

Side effects and Expressions

Consider this code

```
i = 5;
i = i++;
```

Expecting i to be 6?

```
i = 5;
i = i++ + i--;
```

Expecting i to be 5 or 7 or 6?

The result is undefined !

DO NOT WRITE THIS TYPE OF CODE. IT HAS NO MEANING.

In C, result is undefined when evaluation of one operand affects the evaluation of another operand (**this is side effect)**

Boolean Short Circuit Evaluation

The order of evaluation for logical expressions is Left to Right

For example

If (i == 5 && j < 7)

First i == 5 is evaluated, and then j < 7

Short Circuit evaluation

In the above expression, if i == 5 was false, then (due to &&) the result is guaranteed false, so j < 7 is not evaluated !

If i == 5 was true then j < 7 is evaluated.

Do not assume that all parts of the relational expression will run!