Switch statement in C

When you want to solve multiple option type problems, for example: Menu like program, where one value is associated with each option and you need to choose only one at a time, then, `switch` statement is used.

Switch statement is a control statement that allows us to choose only one choice among the many given choices. The expression in `switch` evaluates to return an integral value, which is then compared to the values present in different cases. It executes that block of code which matches the case value. If there is no match, then **default** block is executed(if present). The general form of `switch` statement is,

```
switch(expression)

{

    case value-1:

        block-1;

        break;

    case value-2:

        block-2;

        break;

    case value-3:

        block-3;

        break;

    case value-4:

        block-4;

            break;

    default:

            default-block;

        break;

}
```

Rules for using `switch` statement

1. The expression (after switch keyword) must yield an **integer** value i.e the expression should be an integer or a variable or an expression that evaluates to an integer.
2. The case **label** values must be unique.
3. The case label must end with a colon(:)
4. The next line, after the **case** statement, can be any valid C statement.

Points to Remember

1. We don't use those expressions to evaluate switch case, which may return floating point values or strings or characters.
2. `break` statements are used to **exit** the switch block. It isn't necessary to use `break` after each block, but if you do not use it, then all the consecutive blocks of code will get executed after the matching block.

```
int i = 1;
switch(i)
{
    case 1:
        printf("A");        // No break
    case 2:
        printf("B");        // No break
     case 3:
         printf("C");
         break;
}
```

```
A B C
```

The output was supposed to be only **A** because only the first case matches, but as there is no `break` statement after that block, the next blocks are executed too, until it a `break` statement in encountered or the execution reaches the end of the `switch` block.

13. **default** case is executed when none of the mentioned case matches the `switch` expression. The default case can be placed anywhere in the `switch` case. Even if we don't include the default case, `switch` statement works.
14. Nesting of `switch` statements are allowed, which means you can have `switch` statements inside another `switch` block. However, nested `switch` statements should be avoided as it makes the program more complex and less readable.

Example of `switch` statement

```
#include<stdio.h>

void main( )

{

    int a, b, c, choice;

    while(choice != 3)

    {

        /* Printing the available options */

        printf("\n 1. Press 1 for addition");

        printf("\n 2. Press 2 for subtraction");

        printf("\n Enter your choice");

        /* Taking users input */

        scanf("%d", &choice);


        switch(choice)

        {

            case 1:
```

```
                printf("Enter 2 numbers");
                scanf("%d%d", &a, &b);
                c = a + b;
                printf("%d", c);
                break;
            case 2:
                printf("Enter 2 numbers");
                scanf("%d%d", &a, &b);
                c = a - b;
                printf("%d", c);
                break;
            default:
                printf("you have passed a wrong key");
                printf("\n press any key to continue");
        }
    }
}
```

Difference between `switch` and `if`

- `if` statements can evaluate `float` conditions. `switch` statements cannot evaluate `float` conditions.
- `if` statement can evaluate relational operators. `switch` statement cannot evaluate relational operators i.e they are not allowed in `switch` statement.