Bitwise Operators Lab Assignments

1. Writre a program to Set, clear and toggle a particular bit using bit wise operator?

```
#include <stdio.h>

int main() {
   int n = 10, pos = 1;
   printf("Set: %d\n", n | (1 << pos));
   printf("Clear: %d\n", n & ~(1 << pos));
   printf("Toggle: %d\n", n ^ (1 << pos));
   return 0;
}
```

Output:
Set: 10
Clear: 8
Toggle: 8

2. WAP whether a number is ODD or EVEN using bitwise.

```
#include <stdio.h>

int main() {
   int n = 4;
   (n & 1) ? printf("Odd\n") : printf("Even\n");
   return 0;
}
```

Output:(for n =4)
Even

3. Write a printbinary(int , int) function consists of 2 integer variables. First one is the value of the variable and the second one is the size of the variable.
Example:

```
char x=5;
printbinary(x,sizeof(x));

#include <stdio.h>

void printbinary(int value, int size) {
   for (int i = size * 8 - 1; i >= 0; i--)
```

```
        printf("%d", (value >> i) & 1);
    printf("\n");
}

int main() {
    char x = 5;
    printbinary(x, sizeof(x));
    return 0;
}
```
Output: 00000101

4. WAP to count the bits set (bit value 1) in an integer? Find out and compare different possibilities?

```
int countBits2(int n) {
    int count = 0;
    while (n) {
        n &= (n - 1);
        count++;
    }
    return count;
}
```
Output:(for input 13):
Set bit :3

5. WAP whether a number is a power of 2 or not?

```
#include <stdio.h>

int main() {
    int n = 8;
    if (n > 0 && (n & (n - 1)) == 0)
        printf("Power of 2\n");
    else
        printf("Not a power of 2\n");
    return 0;
}
```
Output:(for n=8): power of 2

6. WAP implements subtraction functionality without using SUB('-') Operator.

```
#include <stdio.h>

int main() {
   int a = 10, b = 4;
   int result = a + (~b + 1);
   printf("Result: %d\n", result);
   return 0;
}
```

Output:
Result:6

7. WAP implements addition functionality without using ADD('+') Operator.

```
#include <stdio.h>

int add(int a, int b) {
   while (b != 0) {
      int carry = a & b;
      a = a ^ b;
      b = carry << 1;
   }
   return a;
}

int main() {
   int x = 7, y = 5;
   printf("Sum: %d\n", add(x, y));
   return 0;
}
```

Output:sum:12

8. WAP implements XOR functionality without using XOR(^) operator.

```
#include <stdio.h>

int main() {
```

```
    int a = 5, b = 3;
    int result = (a | b) & (~a | ~b);
    printf("XOR: %d\n", result);
    return 0;
}
```
Output:XOR:6

9. WAP to implement the sizeof operation using the bitwise operator.

```
#include <stdio.h>

#define SIZEOF(type) ((char *)(&((type *)0)[1]) - (char *)((type *)0))

int main() {
    printf("Size of int: %d\n", SIZEOF(int));
    printf("Size of char: %d\n", SIZEOF(char));
    printf("Size of float: %d\n", SIZEOF(float));
    printf("Size of double: %d\n", SIZEOF(double));
    return 0;
}
```

OUTPUT:
Size of int: 4
Size of char: 1
Size of float: 4
Size of double: 8

10. WAP to convert Little endian integer to Big endian integer

```
#include <stdio.h>

unsigned int convertToBigEndian(unsigned int num) {
    return ((num >> 24) & 0xFF) |
           ((num << 8) & 0xFF0000) |
           ((num >> 8) & 0xFF00) |
           ((num << 24) & 0xFF000000);
}

int main() {
```

```
    unsigned int num = 0x12345678;
    printf("Big Endian: 0x%X\n", convert To Big  Indian(num));
    return 0;
}
```
OUTPUT: Big indian: 0x78563412

11. WAP Swap any two numbers using bitwise operators. How does it work?

```
#include <stdio.h>

int main() {
    int a = 5, b = 10;
    a = a ^ b;
    b = a ^ b;
    a = a ^ b;
    printf("After swapping: a = %d, b = %d\n", a, b);
    return 0;
}
```
Output:
After swapping: a = 10, b = 5