string str = new string('1', position);

int mask = Convert.ToInt32(str, 2);

int rightBits = number & mask; - Така си права права маска за да запазя десните битове.

case "flip":

number ^= 1 << position;

case "insert":

number = number >> position;

number = number << (position + 1);

long mask2 = ((long)1 << position);

number = number | mask2;

number = number | rightBits;

case "remove":

number = number >> (position + 1);

number = number << position;

number = number | rightBits;

static int CountBitOne(int number)

{

int counter = 0;

while (number != 0)

{

int diff = number - 1;

number = number & diff;

counter++;

}

return counter; - Правя си отделен метод за броене на 1ци битове в числото.

static ulong GetBitAtPosition(ulong number, int position)

ulong nRightP = number >> position;

ulong bit = nRightP & 1;

return bit;

static ulong SetBitToZero(ulong number, int position)

ulong mask = (ulong)(~(1 << position));

ulong result = number & mask;

return result;

static ulong SetBitToOne(ulong number, int position)

ulong mask = (ulong)(1 << position);

ulong result = number | mask;

return result; - Това са три отделни метода за вземане на бит от позиция, превръщане на бита в нула и обратното.