You are given an array of up to four non-negative integers, each less than 256.

Your task is to pack these integers into one number M in the following way:

* The first element of the array occupies the first 8 bits of M;
* The second element occupies next 8 bits, and so on.

Return the obtained integer M.

Note: the phrase *"first bits of M"* refers to the *least significant bits* of M - the right-most bits of an integer. For further clarification see the following example.

**Example**

For a = [24, 85, 0], the output should be  
arrayPacking(a) = 21784.

An array [24, 85, 0] looks like [00011000, 01010101, 00000000] in binary.  
After packing these into one number we get 00000000 01010101 00011000 (spaces are placed for convenience), which equals to 21784.

**Input/Output**

* **[time limit] 3000ms (cs)**
* **[input] array.integer a**

*Constraints:*  
1 ≤ a.length ≤ 4,  
0 ≤ a[i] < 256.

* **[output] integer**

<https://codefights.com/arcade/code-arcade/corner-of-0s-and-1s/KeMqde6oqfF5wBXxf>

static string toBin(int n)

{

string bin = "";

while (n > 0)

{

bin = bin.Insert(0, (n % 2).ToString());

n /= 2;

}

return bin;

}

public static int toDec(String bin)

{

char[] temporal = bin.ToCharArray();

Array.Reverse(temporal);

int NumeroDecimal = 0;

for (int i = bin.Length - 1; i >= 0; i--)

{

NumeroDecimal += Convert.ToInt32(int.Parse(temporal[i].ToString()) \* Math.Pow(2, i));

}

return NumeroDecimal;

}

static int arrayPacking(int[] a)

{

string bin = "";

for (int i = a.Length - 1; i >= 0; i--)

{

string temp = toBin(a[i]);

string ceros = new string('0', 8 - temp.Length);

bin += ceros + temp;

}

return toDec(bin);

}