9 Having Funs on Summation

9.1 Function Pointer

One good feature of the pointer is that it can point even functions. A pointer referencing a function is called a function pointer. The following code shows how to use the function pointer:

```c
#include <stdio.h>

int add(int a, int b) {
    return a + b;
}

int sub(int a, int b) {
    return a - b;
}

typedef int (*fun)(int, int);

int main()
{
    fun f[] = { add, sub };
    int a = 10, b = 5, sz = sizeof f/sizeof *f;
    for (int i = 0; i < sz; i++)
        printf("%d\n", f[i](a, b));
    return 0;
}
```

Note that you don't have to use the address-of operator (`&`) on the names of the functions. Note also that the dereferencing operator (`*`) is used for calculating `sz`.

9.2 Programming Lab 9: funsum.c

We can think of two types of sum of positive integers: one is normal addition (sum type 1) and the other is the bitwise addition, say XOR (exclusive or) sum (sum type 0). C supports them with the operators + and ^, respectively.

For the bitwise operation, you should think of the bit representations of operands. For example, the bit-representation of 2 and 3 are 10 and 11, respectively. The XOR of two bits are 1 if they are different, and 0, otherwise. Therefore, computing the expression `2 ^ 3` makes `01`, which is 1 in decimal (`2 ^ 3 == 1`).

For example, given the numbers 1, 2, 3, and 4, applying sum type 1 results in 10, but applying sum type 0 results in 4.

Write a program summing up the nonnegative integers depending on the type of sum. The input consists of a single line containing n integers separated by space ($0 < n < 1,000$). The type of the sum function is determined by the parity of the first number: select type 0 if it is even and type 1, otherwise. Your program should print a line containing the decimal integer for the sum in standard output.

Additional requirements for bonus points

- Use function pointers to solve this problem.
- Do not use `typedef`.

Input	**Output**
5 4 3 2 1	15
2 3 0 1 1 4	5