**H** | 🏠 **Domains** | 🛡 **Contests** | 🏅 **Rank** | 🏆 **Leaderboard** | 💼 **Jobs** | 🔍 | 💬 📢 **H** lcmarkinson ⌄

All Domains > Algorithms > Bit Manipulation > 2's complement

| Badge Progress (Details) | Points: 259.41 Rank: 90625 |

# 2's complement

**H** **by HackerRank**

| Problem | Submissions | Leaderboard | Discussions | Editorial 🔒 |

Understanding *2's complement* representation is fundamental to learning about Computer Science. Let's say you wrote down the $2$'s complement for each $32$-bit integer in the inclusive range from $A$ to $B$; how many $1$'s would you write down in all?

**Input Format**

The first line contains $T$, the number of test cases.
The $T$ subsequent lines each contain two space-separated integers, $A$ and $B$, respectively.

**Constraints**

- $T \leq 1000$

- $-2^{31} \leq A \leq B \leq 2^{31} - 1$

**Output Format**

On a new line for each of the $T$ test cases, print the number of $1$'s in the $32$-bit $2$'s complement representation for integers in the inclusive range from $A$ to $B$.

**Sample Input**

```
3
-2 0
-3 4
-1 4
```

**Sample Output**

```
63
99
37
```

**Explanation**

*Test Case 0*:
$-2$ contains $31$ ones followed by a zero.
$-1$ contains $32$ ones.
$0$ contains $0$ ones.
$31 + 32 + 0 = 63$, so we print $63$ on a new line.

*Test Case 1*:
$31 + 31 + 32 + 0 + 1 + 1 + 2 + 1 = 99$, so we print $99$ on a new line.

*Test Case 2*:
$32 + 0 + 1 + 1 + 2 + 1 = 37$, so we print $37$ on a new line.

in 🐦 f

**Submissions:** 2798

**Max Score:** 70

**Difficulty:** Advanced

**Rate This Challenge:**
☆☆☆☆☆

More

---

**Current Buffer** (saved locally, editable)  ⑂ ⟳          C++  ⌄   ⤢ ⚙

```cpp
1  #include <cmath>
2  #include <cstdio>
3  #include <vector>
4  #include <iostream>
5  #include <algorithm>
6  using namespace std;
7
8
9  int main() {
10     /* Enter your code here. Read input from STDIN. Print output to STDOUT */
11     return 0;
12 }
13
```

Line: 1 Col: 1

⬆ Upload Code as File    ☐ **Test against custom input**      Run Code    Submit Code

Join us on IRC at #hackerrank on freenode for hugs or bugs.