C++ tricks for competitive programming (for C++ 11)

We have discussed some tricks in below post. In this post, some more tricks are discussed.

Writing C/C++ code efficiently in Competitive programming

Although practice is the only way that ensures increased performance in programming contests but having some tricks up your sleeve ensures an upper edge and fast debugging.

1) Checking if the number is even or odd without using the % operator:
Although this trick is not much better than using % operator but is sometimes efficient (with large numbers). Use & operator:

```
if (num & 1)
   cout << "ODD";
else
   cout << "EVEN";
```

Example:
num = 5
Binary: “101 & 1” will be 001, so true

num = 4

Binary: “100 & 1” will be 000, so false.

2) Fast Multiplication or Division by 2

Multiplying by 2 means shifting all the bits to left and dividing by 2 means shifting to the right.

Example : 2 (Binary 10): shifting left 4 (Binary 100) and right 1 (Binary 1)

```
n = n << 1;   // Multiply n with 2
n = n >> 1;   // Divide n by 2
```

3) Swapping of 2 numbers using XOR:

This method is fast and doesn’t require the use of 3rd variable.

```
// A quick way to swap a and b
a ^= b;
b ^= a;
a ^= b;
```

4) Avoiding use of strlen():

```
//  Use of strlen() can be avoided by:
for (i=0; s[i]; i++)
{
}
// loop breaks when the character array ends.
```

5) Use of emplace_back() (Discussed here, here and here)

Instead of push_back() in STL emplace_back can be used because it is much faster and instead of allocating memory somewhere else and then appending it directly allocates memory in the container.

6) Inbuilt GCD function: C++ has inbuilt GCD function and there is no need to explicitly code it. Syntax: __gcd(x, y);

7) Using Inline functions: We can create inline functions and use them without having to code them up during the contest. Examples: (a) function for sieve, (b) function for palindrome.

8) Maximum size of the array: We must be knowing that the maximum size of array declared inside the main function is of the order of 10^6 but if you declare array globally then you can declare its size upto 10^7.

9) Calculating the most significant digit: To calculate the most significant digit of any number log can be directly used to calculate it.

```
Suppose the number is N then
Let double K = log10(N);
now K = K - floor(K);
int X = pow(10, K);
X will be the most significant digit.
```

10) Calculating the number of digits directly: To calculate number of digits in a number, instead of looping you can efficiently use log :

```
Number of digits in N =floor(log10(N)) + 1;
```

11) Efficient trick to know if a number is a power of 2 sing the normal technique of division the complexity comes out to be O(logN), but it can be solved using O(v) where v are the number of digits of number in binary form.

```
/* Function to check if x is power of 2*/
bool isPowerOfTwo (int x)
{
  /* First x in the below expression is
    for the case when x is 0 */
  return x && (!(x&(x-1)));
}
```

12) C++11 has in built algorithms for following:

```
      // are all of the elements positive?
      all_of(first, first+n, ispositive());

     // is there at least one positive element?
     any_of(first, first+n, ispositive());
```

```
// are none of the elements positive?
none_of(first, first+n, ispositive());
```

Please refer Array algorithms in C++ STL (all_of, any_of, none_of, copy_n and itoa) for details.

13) Copy Algorithm: used to copy elements from one container to another.

```
int source[5] = {0, 12, 34, 50, 80};
int target[5];
// copy 5 elements from source to target
copy_n(source, 5, target);
```

Please refer Array algorithms in C++ STL (all_of, any_of, none_of, copy_n and itoa) for details.

14) The Iota Algorithm The algorithm iota() creates a range of sequentially increasing values, as if by assigning an initial value to *first, then incrementing that value using prefix ++. In the following listing, iota() assigns the consecutive values {10, 11, 12, 13, 14} to the array arr, and {'a', 'b', 'c'} to the char array c[].

```
int a[5] = {0};
char c[3] = {0};

// changes a to {10, 11, 12, 13, 14}
iota(a, a+5, 10);
iota(c, c+3, 'a'); // {'a', 'b', 'c'}
```

Please refer Array algorithms in C++ STL (all_of, any_of, none_of, copy_n and itoa) for details.

15) Initialization in Binary form: In C++ 11 assignments can also be made in binary form.

```
// C++ code to demonstrate working of
// "binary" numbers
#include<iostream>
using namespace std;
int main()
```

```
{
    auto number = 0b011;
    cout << number;
    return 0;
}
```

Run on IDE

Output :

```
3
```

16) Use of "and" Though not a very productive one, this tip helps you to just use conditional operator and instead of typing &.

```
// C++ code to demonstrate working of "and"
#include<iostream>
using namespace std;
int main()
{
    int a = 10;
    if (a < 20 and a > 5)
      cout << "Yes";
    return 0;
}
```

Run on IDE

Output :

```
Yes
```

This article is contributed by **Yash Kodesia**. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.