Reverse Integer

Given a 32-bit signed integer, reverse digits of an integer.

Example 1:

```
Input: 123
Output: 321
```

Example 2:

```
Input: -123
Output: -321
```

Example 3:

```
Input: 120
Output: 21
```

Note:
Assume we are dealing with an environment which could only store integers within the 32-bit signed integer range: [231, 231 1]. For the purpose of this problem, assume that your function returns 0 when the reversed integer overflows.

Solution 1 in C++

```
#include <iostream>
#include <math.h>


using namespace std;

class Solution {
public:
    int reverse(int x) {
        int y=x;
        long ret=0;
        long valid_range = pow(2,31);
        while(y)
        {
            ret *= 10;
            ret += (y %10);
            cout << ret << " - " << y << endl;
            y /= 10;
        }
        if ((ret<valid_range*(-1)) || (ret>(valid_range-1))) return 0;
        return ret;
    }
};

int main(void)
{
    Solution s;
    int testcase;

    testcase = 123;
    cout << testcase << " -> " << s.reverse(testcase) << endl;

    testcase = -123;
    cout << testcase << " -> " << s.reverse(testcase) << endl;

    testcase = 120;
    cout << testcase << " -> " << s.reverse(testcase) << endl;

    testcase = 1534236469;
    cout << testcase << " -> " << s.reverse(testcase) << endl;

    return 0;
}
```

Solution 2 in C++

```
class Solution {
public:
    int reverse(int x) {
        int rev = 0;
        while (x != 0) {
            int pop = x % 10;
            x /= 10;
            if (rev > INT_MAX/10 || (rev == INT_MAX / 10 && pop > 7)) return 0;
            if (rev < INT_MIN/10 || (rev == INT_MIN / 10 && pop < -8)) return 0;
            rev = rev * 10 + pop;
        }
        return rev;
    }
};
```

Solution 3 in Java

```
class Solution {
    public int reverse(int x) {
        int rev = 0;
        while (x != 0) {
            int pop = x % 10;
            x /= 10;
            if (rev > Integer.MAX_VALUE/10 || (rev == Integer.MAX_VALUE / 10 && pop > 7)) return 0;
            if (rev < Integer.MIN_VALUE/10 || (rev == Integer.MIN_VALUE / 10 && pop < -8)) return 0;
            rev = rev * 10 + pop;
        }
        return rev;
    }
}
```