# HW1: Data Representation

## Students

1. Nguyễn Gia Huy - 22127154
2. Trần Thành Long - 22127250

## Main idea

- Our program has been written in C++.
- To solve this problem, we use the boolean array consisting of 8 elements to store the binary form of the input number. An element of the array is true if the corresponding bit of the input number is 1, otherwise, it is false.

### Sign-Magnitude form

To convert a decimal number to the sign-magnitude form, we just need to check if the input number is negative or not. If the input number is negative, we set the first element of the array to true, otherwise, we set it to false. Then, we convert the absolute value of the input number to the binary form and store it in the rest of the array. The range of the sign-magnitude form is from -127 to 127. Number 0 has 2 representations: 00000000 and 10000000.

### One’s complement form

To convert a decimal number to the one’s complement form, we just need to check if the input number is negative or not. If the input number is negative, we set the first element of the array to true, otherwise, we set it to false. Then, we convert the absolute value of the input number to the binary form and store it in the rest of the array. Finally, we invert all the elements of the array except the first element. The range of the one’s complement form is from -127 to 127. Number 0 has 2 representations: 00000000 and 11111111.

### Two’s complement form

To convert a decimal number to the two’s complement form, we just need to check if the input number is negative or not. If the input number is negative, we set the first element of the array to true, otherwise, we set it to false. Then, we convert the absolute value of the input number to the binary form and store it in the rest of the array. Finally, we add 1 to the array. The range of the two's complement form is from -128 to 127. Number 0 has only 1 representation: 00000000.

## Implementation

In our program, we have 2 classes: `BinaryPattern` and `Solution`.

- `BinaryPattern` is used to store the binary form of the input number.
- `Solution` is used to solve the problem. it will check if the input number is valid or not, then it will convert the input number to the sign-magnitude form, one’s complement form, and two’s complement form, including checking for special cases; otherwise, it will print an error message **overflow** or **invalid**.

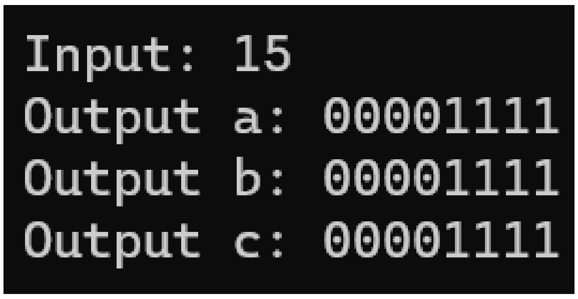
## Source code

Google Drive

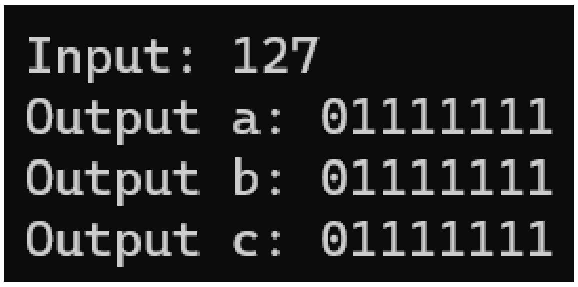
## Screenshots

### 1. Valid inputs

- Positive number

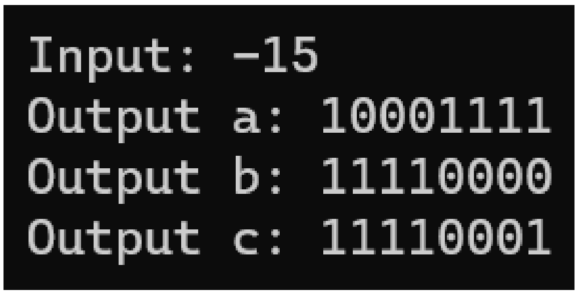


```
Input: 15
Output a: 00001111
Output b: 00001111
Output c: 00001111
```

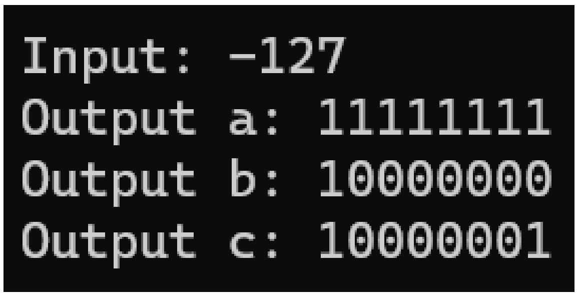


```
Input: 127
Output a: 01111111
Output b: 01111111
Output c: 01111111
```

- Negative number

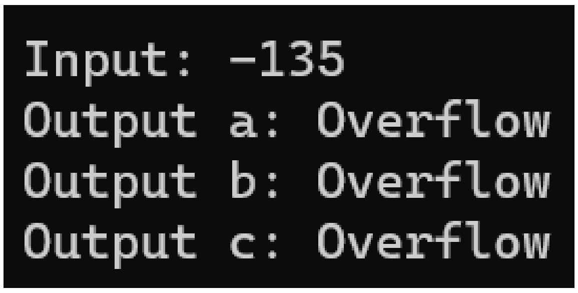


```
Input: -15
Output a: 10001111
Output b: 11110000
Output c: 11110001
```

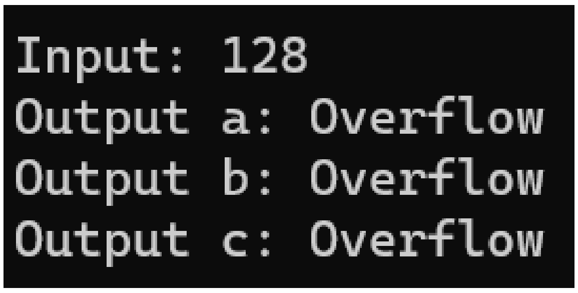


```
Input: -127
Output a: 11111111
Output b: 10000000
Output c: 10000001
```

## 2. Invalid inputs

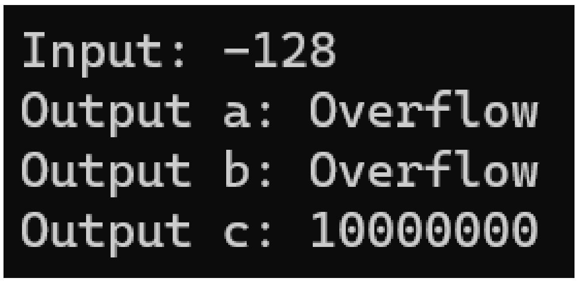


```
Input: -135
Output a: Overflow
Output b: Overflow
Output c: Overflow
```

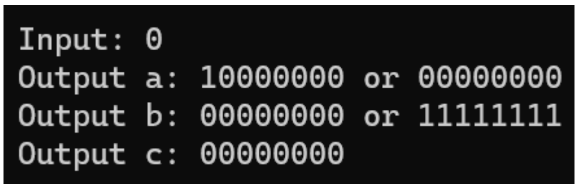


```
Input: 128
Output a: Overflow
Output b: Overflow
Output c: Overflow
```

## 3. Special cases



```
Input: -128
Output a: Overflow
Output b: Overflow
Output c: 10000000
```



```
Input: 0
Output a: 10000000 or 00000000
Output b: 00000000 or 11111111
Output c: 00000000
```