**Course:** CpSc 212: Algorithms and Data Structures Spring 2014
**Lab 01: Array Manipulation**

# 1 Jumping Batman

Batman is jumping from one building to another to save Gotham from the joker. He takes a high jump every time he jumps from a lower building to a higher building and a low jump when jumping from a higher building to a lower building. Given the heights of buildings, please compute the total number of low and high jumps he will have to take.

## 1.1 Input

The heights of each building is given as an integer in meters above or below the sea level. Each building is seperated by a space. There will be at most $1,000,000$ buildings in the input.

## 1.2 Output

Output two integers, the total number of low jumps followed by the total number of high jumps.

Sample I/O

| Test case# | Input | Output |
|---|---|---|
| 1. | 2 3 1 4 5 | 1 3 |
| 2. | 1 | 0 |
| 3. | 1 1 1 | 0 |
| 4. | 2 3 3 2 | 1 1 |

# 2 The $3n + 1$ Problem

Consider the following algorithm.

1. Input $n$.
2. Print $n$.
3. If $n == 1$, STOP.
4. if $n$ is even: $n \leftarrow n/2$
   else: $n \leftarrow 3n + 1$
5. goto step 2.

Given the input 99, the following sequence of numbers will be printed. 99, 298, 149, 448, 224, 112, 56, 28, 14, 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1 and the length of this sequence, also known as the *cycle length* is 26.

It is widely conjectured (known as the Collatz conjecture) that this sequence terminates. That is for any input $n$, this sequence terminates with printing 1. This has been verified for atleast $n \leq 1,000,000$.

Between two numbers $i$ and $j$, what is the number with the maximum cycle length between $i$ and $j$.

### 2.1 Input

The input consists of two integers $i$ and $j$, separated by a space. Note that $1 \leq i, j \leq 1000$.

### 2.2 Output

The output consists of two numbers: the number with maximum cycle length between $i$ and $j$, and its cycle length.

Sample I/O

| Test case# | Input | Output |
|---|---|---|
| 1. | 4 10 | 9 20 |
| 2. | 100 200 | 171 125 |
| 3. | 1000 1001 | 1001 143 |
| 4. | 999 999 | 999 50 |

### 2.3 Additional Notes and Extra Credit

You can use an array to store previous computations. Consider the Collatz sequence for 5 and 6.

5 16 8 4 2 1
6 3 10 5 16 8 4 2 1

Note that the above two sequences have a lot of numbers in common. Therefore, when we have computed the cycle length for 5, we can store this result in an array and re use when computing the cycle length for 6. If you decide to use this technique, you may need to allocate enough memory so as to hold the cycle length of the largest number (approx. $251,000$) you encounter during your computations.

For extra credit (3 points), your program should work for $1 \leq i, j \leq 1000000$.

## 3 Compiling and running

Save your files with the following names.

| Exercise# | Filename |
|---|---|
| 1. | lab01_batman.cpp |
| 2. | lab01_3n.cpp |

To compile your file, use the following command.

```
g++ file.cpp -o file
```

For example, you will compile the first program as:

```
g++ lab01_batman.cpp -o lab01_batman
```

Recall that the "-o lab_batman" part of the command line tells the compiler what to name your executable file, in case you want to give it a different name. You can then execute your code by running:

```
./lab_batman
```

You can also enter input in a file named *in* and redirect your input using the following command:

```
./lab_batman < in
```

## 4 Testing

You are required to test for all cases that meet the specifications above. Please thoroughly test your program with all boundary cases. You will receive full credit only if the program works for all cases.

## 5 Submission

Please submit your solutions to handin.cs.clemson.edu. Please log into handin and set it up before submitting. The due date for this assignment is Friday January 17th at 11:59pm.

## 6 Grading

Each program is graded on a scale of zero to ten points. No points will be awarded to a program that does not compile. You will recieve 1 point for successful compilation and 1 point for successful execution (no seg faults and infinite loops). 3 points is awarded for correctness and efficiency each and 2 points awarded for good code: readability, comments, simplicity and elegance, etc.