UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG
School of Computer Science & Applied Mathematics

Introduction to Algorithms & Programming (COMS1018A/1022A) Assignment 1

Date Released: 12 March 2024, 10h15
Date Due: 8 April 2024, 23h59

1 Introduction

Like the labs, assignments are marked automatically online. This means that your program must produce *exactly* the expected output for it to be considered correct. For each task, submit the appropriate `.py` file to the Moodle marker. Successfully completing all six tasks will earn you 100%.

Again, please be aware of the policy surrounding plagiarism (see the course outline for a full description). Code will be scrutinised, and anyone found to have cheated (including the person from whom the code was copied) will immediately receive 0, and may be subject to disciplinary sanctions.

2 Background

Zlatan and Lionel have been best friends since birth. The two of them are completely inseparable, and even when they're apart, they're constantly staring at their phones sending messages to each other. Lionel occasionally gets annoyed with Zlatan who often sends a few too many Christiano Ronaldo memes. Apart from that, though, things couldn't be better.

One day however, Zlatan stumbled upon a whole bunch of conspiracy websites. The sites managed to convince him that not only is the earth flat, but that it's actually a good idea for a man to have both a ponytail *and* a goatee! But worst of all, Zlatan became totally convinced that NASA[1] are spying on his conversations with Lionel.

Lionel knows that this is completely ridiculous, but he's too busy making adverts for Lays chips, and doesn't really have the time to argue with Zlatan — he reckons that it's just easier to go along with Zlatan's delusions. He does want to make Zlatan feel better though, so he gets in touch with you and asks if you could make some program that could *encrypt* and *decrypt* their conversations and ensure that no-one is able to read any intercepted messages.

You'd rather not help out — you've got many matrices that need to be Gaussian eliminated — but then you remember that Lionel doesn't pay tax, and so probably has lots of money. He might even give you some! You can't get in touch with him, so you leave a note with his receptionist (some Brazilian with ridiculous hair called Neymar) saying that you'll take the job.

3 The Plan

Even though you've offered to help out, you're still very busy and so you can't allocate too much time to this task. Implementing a robust encryption system would take too long, so you decide to provide *security through obfuscation*. This means that if the people spying on you don't know

[1] Not the NSA. NASA. The space agency.

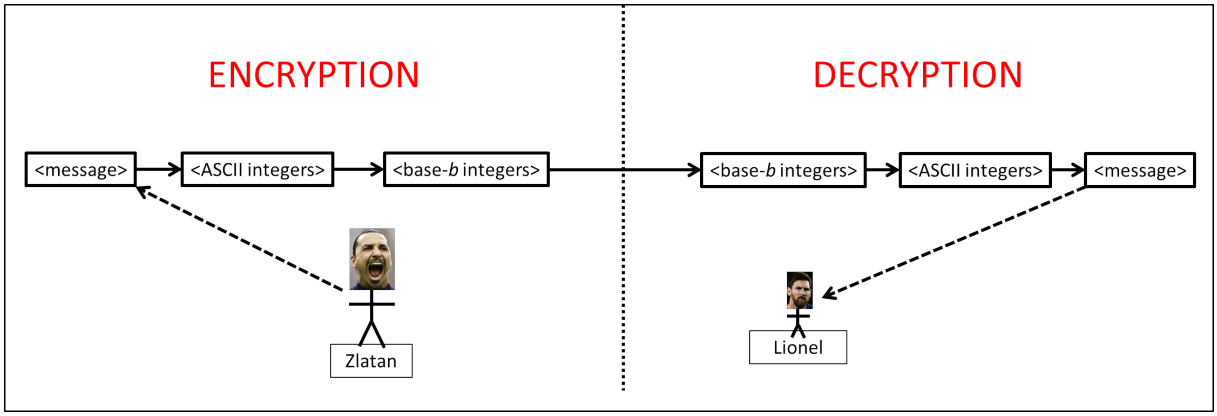

Figure 1: A diagram of what happens when Zlatan sends a message to Lionel.

how you've altered a message, they probably won't be able to recover the original message. Here's the method you come up with:

First, Zlatan and Lionel both agree on a certain secret number $b \in [2, 36]$. Let's assume Zlatan wants to send a message to Lionel. Each character in the string is converted into its ASCII representation. Next, each ASCII number is converted from its decimal representation to base-b. These numbers (which are now in base-b) are sent to Lionel. Lionel receives these numbers and converts them back to base-10 numbers. These numbers are converted back to the characters they represent in ASCII, and the original message can then read. A graphical representation of the process is drawn above.

You're not going to write this as one big system, so you break it down into a series of tasks.

4 The First Task (25 marks)

Write a program that reads in a line of text and converts each character into its ASCII representation. The line of text may contain letters, digits, punctuation **and spaces**. Your output should consist of a single line containing each character's ASCII number. Numbers should be separated by a single space space.

4.1 Input

Input consists of a single *line of text* (there may be spaces, so be careful). You can assume that the line is never empty.

4.2 Output

Your output should consist of a *single line* of integers, separated by spaces. Each integer is the ASCII representation of the characters of the given string.

4.3 Example Input-Output Pairs

Sample Input #1

```
Hi Leo. How are you?
```

Sample Output #1

```
72 105 32 76 101 111 46 32 72 111 119 32 97 114 101 32 121 111 117 63
```

Sample Input #2

```
Masterchef was great!
```

Sample Output #2

```
77 97 115 116 101 114 99 104 101 102 32 119 97 115 32 103 114 101 97 116 33
```

Sample Input #3

```
Dane Klate is soo good
```

Sample Output #3

```
68 97 110 101 32 75 108 97 116 101 32 105 115 32 115 111 111 32 103 111 111 100
```

---

5 The Second Task (25 marks)

Your next task is to do the opposite of the first task: write a program that accepts a number of integers, converts them to their ASCII characters, and outputs the resulting string.

5.1 Input

The input consists of a series of *integers*, one per line. You can assume that there will always be at least one number given. The end of the input will be signalled by the number -1, which simply tells you when to stop reading in, and **should not be processed.**

5.2 Output

Your output should consist of a *single line* of text that results when the given integers are converted into characters.

5.3 Example Input-Output Pairs

Sample Input #1

```
69
97
116
-1
```

Sample Output #1

```
Eat
```

Sample Input #2

```
80
114
97
121
-1
```

Sample Output #2

```
Pray
```

Sample Input #3

```
76
111
118
101
33
-1
```

Sample Output #3

```
Love!
```

6 The Third Task (15 marks)

Your next task is write a program that converts decimal integers into a given base.

6.1 Input

The first line of input is a number $2 \leq b \leq 36$, which represents the base we want to convert to. The rest of the input consists of a series of *non-negative integers*, one per line. You can assume that there will always be at least one number given. The end of the input will be signalled by the number -1, which simply tells you when to stop reading in, and **should not be processed.**

6.2 Output

For each decimal number, output its base-b representation. Use the capital letters for bases greater than 10.

6.3 Example Input-Output Pairs

Sample Input #1

```
2
42
1
-1
```

Sample Output #1

```
101010
1
```

Sample Input #2

```
16
48879
31
-1
```

Sample Output #2

```
BEEF
1F
```

Sample Input #3

```
36
29053366
36661
1432321437
-1
```

Sample Output #3

```
HAPPY
SAD
NORMAL
```

7 The Fourth Task (20 marks)

Now, write a program that does the opposite of the previous task: converts integers in a given base into decimal numbers.

7.1 Input

The first line of input is a number $2 \leq b \leq 36$, which represents the base we want to convert from. The rest of the input consists of a series of strings, one per line. Be aware that these cannot be treated simply as integers, because they may contain letters. If there are letters, they are always uppercase. You can assume that there will always be at least one number given. The end of the input will be signalled by -1, which simply tells you when to stop reading in, and **should not be processed.**

7.2 Output

For each base-b number, output its decimal representation.

7.3 Example Input-Output Pairs

Sample Input #1

```
8
42
1
-1
```

Sample Output #1

```
34
1
```

Sample Input #2

```
10
999
-1
```

Sample Output #2

```
999
```

Sample Input #3

```
25
42
BAE
MAMMA
-1
```

Sample Output #3

```
102
7139
8764310
```

---

8 The Fifth Task (10 marks)

It's finally time to put everything together! The input format will be slightly different here, but essentially you'll be combining the previous tasks to make a full system that is able to encrypt or decrypt any message.

8.1 Input

Input consists of three lines. The first line is a string that has either the value `encrypt` or `decrypt` and represents the mode. The next line contains a single integer $2 \leq b \leq 36$ which represents the base you'll be working with. The third line is the message (which is either a normal text message, or a series of base-b numbers separated by spaces, depending on the mode). You can assume the third line is never empty.

8.2 Output

Your output will be a single line that depends on the mode.

8.2.1 Encryption Mode

If the mode is `encrypt`, then the message line is a normal message that needs to be encrypted. Convert each character to its ASCII value, and then convert each of those numbers to their base-b representations. Output these base-b numbers on a single line, with each number separated by a single space.

8.2.2 Decryption Mode

If the mode is `decrypt`, then the message line is a series of base-b numbers, each separated by a single space. The message line does not end with a -1. Convert each number into its decimal representation, and then into its ASCII character. Output all of these characters on a single line.

8.3 Some Advice

Reading in a series of strings on a single line is not a trivial task. You will need to look up how to `split` a line into its individual entries.

8.4 Example Input-Output Pairs

Sample Input #1

```
encrypt
16
What's for lunch?
```

Sample Output #1

```
57 68 61 74 27 73 20 66 6F 72 20 6C 75 6E 63 68 3F
```

Sample Input #2

```
decrypt
36
2C 2P 3D 30 33 36 W 2B 3B 2X 2U 38 W 2X 37 W 2P 2S 33 36 2P 2Q 30 2T 18 W 3D 2T 37 1R
```

Sample Output #2

```
Taylor Swift is adorable, yes?
```

Sample Input #3

```
encrypt
16
COMS dominates CAM ;)
```

Sample Output #3

```
43 4F 4D 53 20 64 6F 6D 69 6E 61 74 65 73 20 43 41 4D 20 3B 29
```

9 The Sixth Task (5 marks)

This task requires you to extend the previous task slightly in order to handle a message that might be a paragraph.

9.1 Input

Input consists of at least 3 lines. The first line is a string that has either the value `encrypt` or `decrypt` and represents the mode. The next line contains a single integer $2 \leq b \leq 36$ which represents the base you'll be working with. The remaining lines are the message (either a normal text message, or a series of base-b numbers separated by spaces, depending on the mode). You can assume that there is at least one message line.

9.2 Output

Your output will be a single line that depends on the mode.

9.2.1 Encryption Mode

If the mode is `encrypt`, then the message lines represent a normal message that needs to be encrypted. Convert each character to its ASCII value (newlines should also be converted to their ASCII value), and then convert each of those numbers to their base-b representations. Output these base-b numbers on a single line, with each number separated by a single space.

9.2.2 Decryption Mode

If the mode is `decrypt`, then the message line is a single line consisting of any amount of base-b numbers, each separated by a single space. The message line does not end with a -1. Convert each number into its decimal representation, and then into its ASCII character. Output all of these characters. Because some of these numbers may be the newline character, your output may appear on multiple lines.

9.3 Example Input-Output Pairs

Sample Input #1

```
decrypt
16
48 65 6C 6C 6F 20 4C 65 6F 2E A 48 6F 77 20 61 72 65 20 79 6F 75 3F
```

Sample Output #1

```
Hello Leo.
How are you?
```

Sample Input #2

```
encrypt
31
I
am
Zlatan
```

Sample Output #2

```
2B A 34 3G A 2S 3F 34 3N 34 3H
```