MP2: J ibwf b tfdsfu *(I have a secret)*
Due: Tuesday, Sept. 23, 2014 by 6:00pm

Jouspevdujpo

Pof pg uif nptu jnqpsubou ufdiopmphjft uibu bmmpx uif joufsofu up gvodujpo bt ju epft upebz jt uif bcjmjuz up dpnnvojdbuf tfdvsfmz. Fwfszuijoh gspn tipqqjoh, cboljoh, boe vojwfstjuz hsbefts bsf qspufdufe gspn bozpof fmtf cvu zpv sfbejoh uif jogpsnbujpo uispvhi b ufdiopmphz lopxo bt IUUQT.

Jotufbe pg xpsljoh xjui IUUQT, b ufdiopmphz uibu jowpmwft fyusfnfmz ifbwz-evuz nbui boe uppl zfbst gps b ufbn up eftjho, xf xjmm fybnjof b tjnjmbs bmhpsjuin uibu xjmm tujmm nblf pvs dpnnvojdbujpot npsf "tfdvsf".

…oh, whoops! Allow me to start over…

Introduction

One of the most important technologies that allow the internet to function as it does today is the ability to communicate securely. Everything from shopping, banking, and university grades are protected from anyone else but you reading the information through a technology known as HTTPS.

Instead of working with HTTPS, a technology that involves extremely heavy-duty math and took years for a team to design, we will examine a similar algorithm that will still make our communications more "secure".

…was that better?

The preceding two blocks of text both contain the exact same text, except the first block has been **encrypted** to prevent someone who might have found this paper lying around to easily read it. Specifically, the encryption method we used is called the **Caesar Cipher** and it is done by simply shifting the letters in the alphabet by a certain number of characters. In our case, we only shifted the characters by one.

Knowing how it was encrypted, we can set up a translation table that will allow us to **decrypt** the original message:

Encrypted:	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
Decrypted:	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Using the translation table, we can start decrypting. The first word in the encrypted text was "Pof".

- Looking at "P" in the encrypted row, it translates to "O".
- Looking at "o" in the encrypted row, it translates to "n".
- Looking at "f" in the encrypted row, it translates to "e".
- …therefore, the "Pof" ➔ "One".

In this MP, you will be writing a **Caesar Cipher** to both encrypt and decrypt text. When you have finished this MP, you will be able to encrypt a string and send that message to a friend who will only be able to read it if they can decrypt your message!

Remember: This MP is a **solo assignment**. You should review the academic integrity policy on the course website if you have any concerns about what is reasonable to do when completing this MP.

Getting Started

Similar to MP1 and the labs, we have prepared a base set of files for you work from. You can find these files on the CS 105 website, under the "Assignments" tab, and under the "MP2" link. The file you will download is a zipped file and needs to be **extracted**.

Overview and Review of Lab #3

In order to encrypt or decrypt the String, you must visit each and every character, either shift or un-shift that character, and add the shifted or un-shifted character to a new string.

This process is very similar to your *Lab #3: A MessAge in a ZillIoN Glass bottles* where you looked at each character and checked if it was a capital (uppercase) letter. Instead of checking if each character is an uppercase letter, you should encrypt (or decrypt) the letter and add that to the String you will be returning. You should refer to Lab #3 to review the **`s.charAt()`** and **`s.charCodeAt()`** functions.

Shifting Characters within the String

During Week #3 of lecture and as part of Lab #3, you have seen the **ASCII chart** – the numeric representation that the computer uses for each character that makes up a string. The ASCII chart for the uppercase and lowercase letters is reprinted here:

60 +					70 +										80 +										90
5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
90 +			100 +										110 +										120 +		
7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2
a	b	c	d	e	f	g	h	I	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z

In order to **shift** a character, similar to the encryption shown at the beginning of this document, you will need to find the ASCII value of your character and add to it the amount of characters you wish to shift it by. For example: encrypting the character "M" with a **`shift`** of 1 will in the character "N"; encrypting the character "Q" (ASCII value 81) with a **`shift`** of 5 will result in 81 + 5 = 86, which is "V".

As part of this MP, we have already included all of this logic to shift a character. Inside your **`mp2.js`** file, you should review the **`shiftCharacter()`** code that we have provided to you. You will need to call this function to do the encryption of the input.

Your Task

Also inside your **`mp2.js`** file, you will find that you have a empty **`crypt()`** function that takes in two parameters:

- **`originalString`**, a string that the user wants to encrypt or decrypt
- **`shift`**, the amount the user wants to shift each character in the string

To complete this MP, you must complete the **`crypt()`** function. This function must return the original string shifted by the amount requested. We have already programmed up the logic to have the user type the string in, so all you need to focus on is the shifting of the strings.

Testing and Submitting Your Program

Since the **`crypt()`** function can handle both positive and negative shifts, you can use this to both encrypt and decrypt a message. In order to test to determine if your program works, try some of the following inputs:

- "ABC" with a shift of 1 ➔ BCD
- "Hello" with a shift of 3 ➔ Khoor
- "CS 105" with a shift of 10 ➔ MC 105
- "Vrznjhz" with a shift of -10 ➔ _____

Scoring and Submission

Similar to MP1, this MP will be graded on an "all or nothing basis". To submit MP2, follow the link on the MP2 page on the CS 105 website for the submission page.