NETWORK SECURITY
LAB 1 REPORT

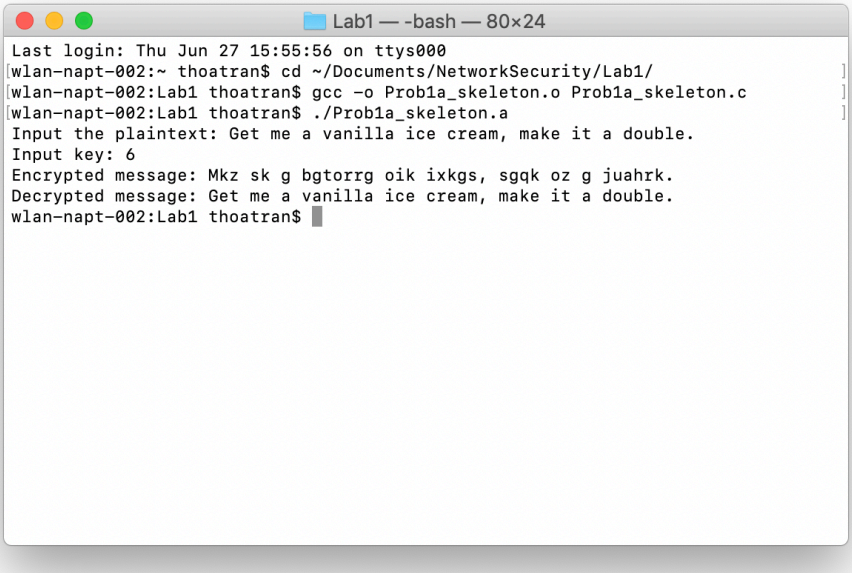
Name: Tran Thi Thoa
ID: s1242006

Problem 1:Implement C functions for encryption/decryption with the Caeser cipher as well as attacks.

(c) Compile and run the file Prob1a_skeleton.c

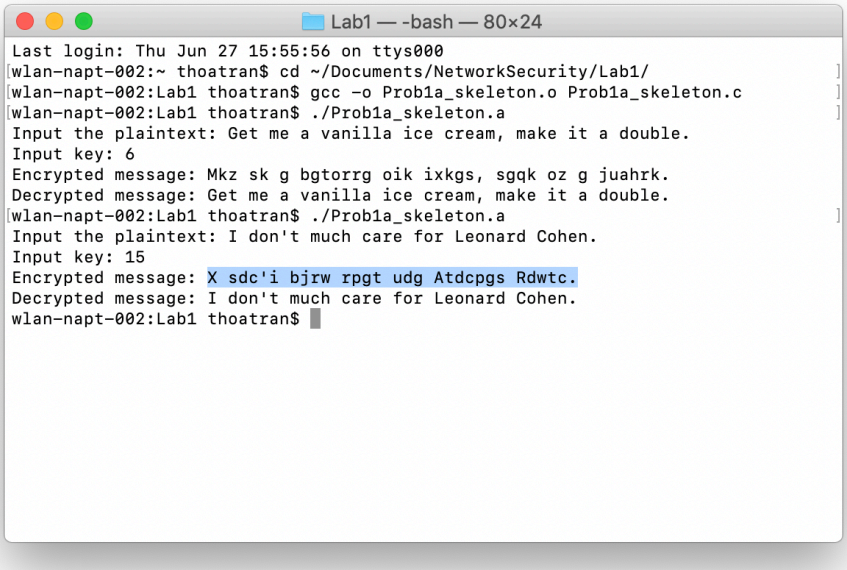
Using the functions in parts on the following (key, plaintext), we have:

+ k = 6, plaintext = "Get me a vanilla ice cream, make it a double."

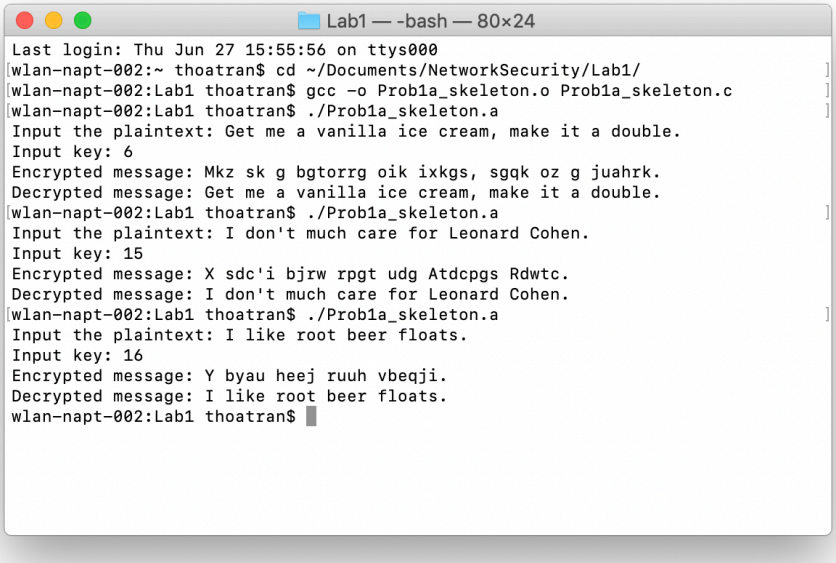


So, the encrypted text will be "Mkz sk g bgtorrg oik ixkgs, sgqk oz g juahrk."

+ k = 15 plaintext = "I don't much care for Leonard Cohen."

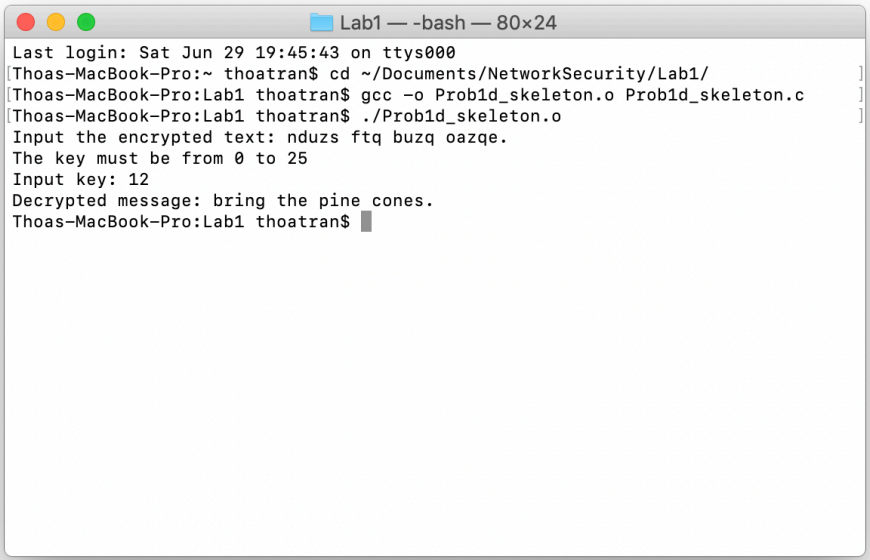


So, the encrypted text will be “X sdc'i bjrw rpgt udg Atdcpgs Rdwtc.”
+ k = 16 plaintext = "I like root beer floats.”



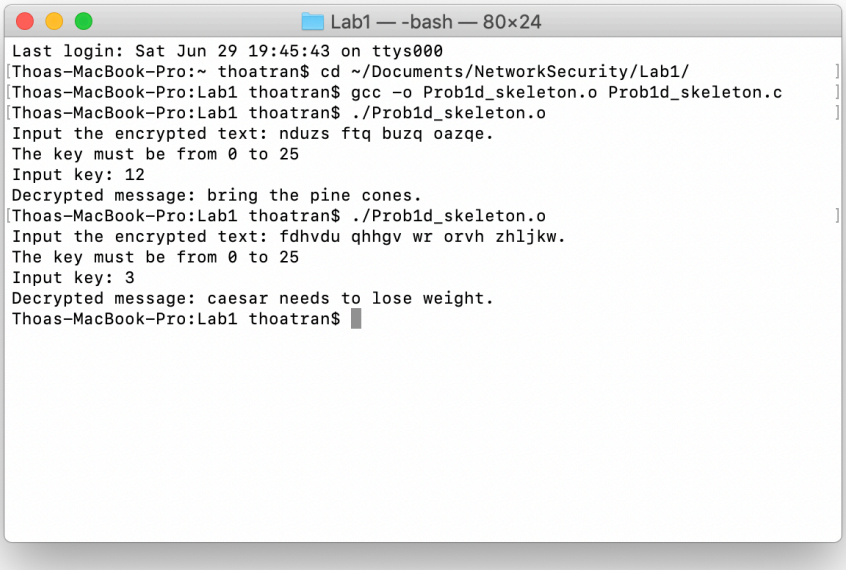
So, the encrypted text will be : “Y byau heej ruuh vbeqji.”

(d) Compile and run the file Prob1d_skeleton.c, having the same function with the program in file Prob1a_skeleton.c, but differences in the input.
+k = 12 ciphertext = 'nduzs ftq buzq oazqe.’



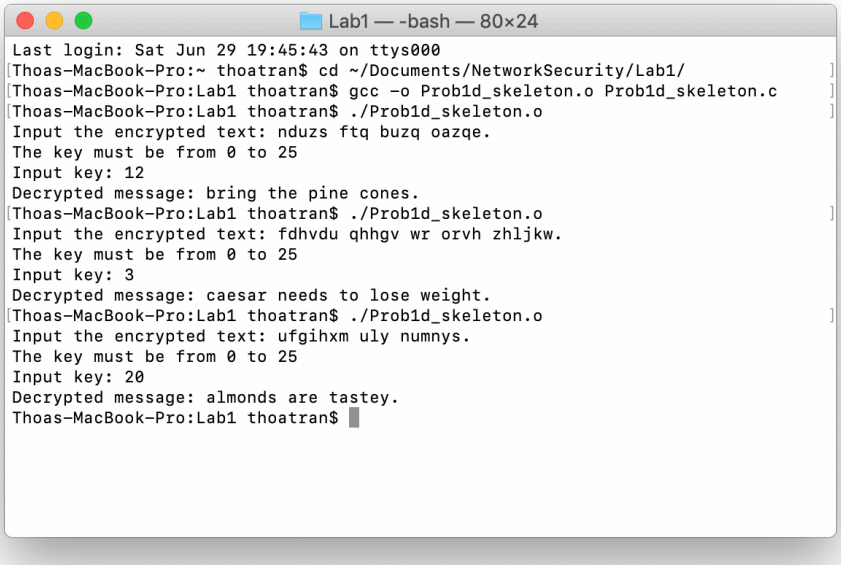
So, the decrypted text is: 'bring the pine cones.’

+ k = 3 ciphertext = "fdhvdu qhhgv wr orvh zhljkw.”



So, the decrypted text will be: “caesar needs to lose weight.”

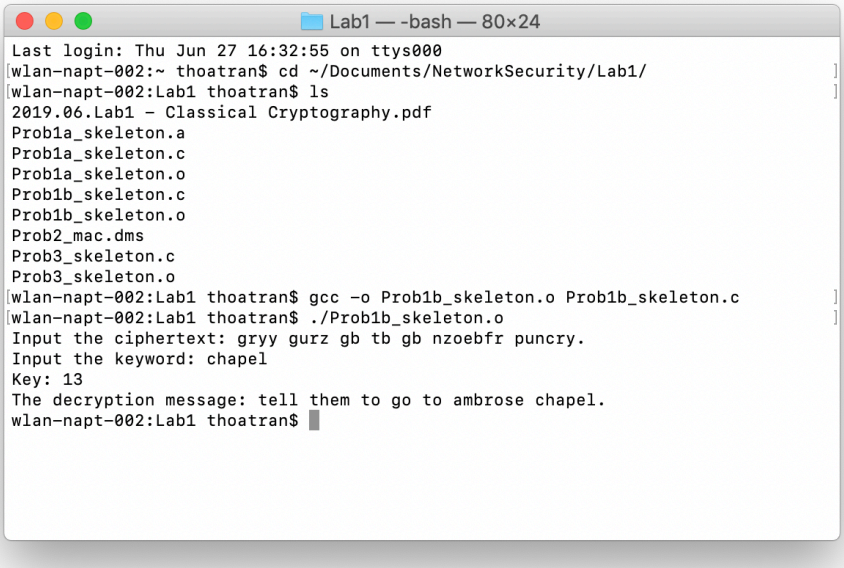
+ k = 20 ciphertext = "ufgihxm uly numnys."



So, the decrypted text will be: “almonds are tastey.”
(e) Run the file Prob1b_skeleton.c by the following command lines:

gcc -o Prob1b_skeleton.o Prob1b_skeleton.c
./Prob1b_skeleton.o

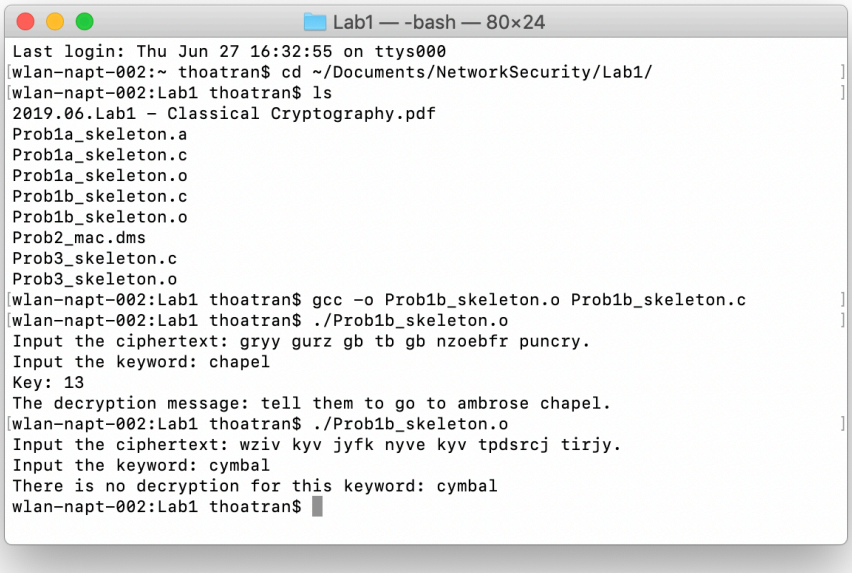
Show the output of the attack function (part b)
+ ciphertext = 'gryy gurz gb tb gb nzoebfr puncry.' keyword = ‘chapel'



So, the decrypted text for the keyword ‘chapel’ is: “tell them to go to ambrose chapel.”

With key = 13

+ ciphertext = 'wziv kyv jyfk nyve kyv tpdsrcj tirjy.' keyword = ‘cymbal'



So there is no encyption text for the keyword ‘cymbal’.

+ ciphertext = 'baeeq klwosjl osk s esf ozg cfwo lgg emuz.' no keyword

The output will be:

```
wlan-napt-002:Lab1 thoatran$ ./Prob1b_skeleton.o
Input the ciphertext: baeeq klwosjl osk s esf ozg cfwo lgg emuz.
Input the keyword:
Key: 0
The decryption message: baeeq klwosjl osk s esf ozg cfwo lgg emuz.
Key: 1
The decryption message: azddp jkvnrik nrj r dre nyf bevn kff dlty.
Key: 2
The decryption message: yxbbn hitlpgi lph p bpc lwd zctl idd bjrw.
Key: 3
The decryption message: vuyyk efqimdf ime m ymz ita wzqi faa ygot.
Key: 4
The decryption message: rquug abmeizb eia i uiv epw svme bww uckp.
Key: 5
The decryption message: mlppb vwhzduw zdv d pdq zkr nqhz wrr pxfk.
Key: 6
The decryption message: gfjjv pqbtxoq txp x jxk tel hkbt qll jrze.
Key: 7
The decryption message: zycco ijumqhj mqi q cqd mxe adum jee cksx.
Key: 8
The decryption message: rquug abmeizb eia i uiv epw svme bww uckp.
Key: 9
The decryption message: ihllx rsdvzqs vzr z lzm vgn jmdv snn ltbg.
Key: 10
The decryption message: yxbbn hitlpgi lph p bpc lwd zctl idd bjrw.
Key: 11
The decryption message: nmqqc wxiaevx aew e qer als oria xss qygl.
Key: 12
The decryption message: baeeq klwosjl osk s esf ozg cfwo lgg emuz.
Key: 13
The decryption message: onrrd xyjbfwy bfx f rfs bmt psjb ytt rzhm.
Key: 14
The decryption message: azddp jkvnrik nrj r dre nyf bevn kff dlty.
Key: 15
The decryption message: lkooa uvgyctv ycu c ocp yjq mpgy vqq owej.
Key: 16
The decryption message: vuyyk efqimdf ime m ymz ita wzqi faa ygot.
Key: 17
The decryption message: edhht nozrvmo rvn v hvi rcj fizr ojj hpxc.
Key: 18
The decryption message: mlppb vwhzduw zdv d pdq zkr nqhz wrr pxfk.
Key: 19
The decryption message: tswwi cdogkbd gkc k wkx gry uxog dyy wemr.
Key: 20
The decryption message: zycco ijumqhj mqi q cqd mxe adum jee cksx.
Key: 21
The decryption message: edhht nozrvmo rvn v hvi rcj fizr ojj hpxc.
Key: 22
The decryption message: ihllx rsdvzqs vzr z lzm vgn jmdv snn ltbg.
Key: 23
The decryption message: lkooa uvgyctv ycu c ocp yjq mpgy vqq owej.
Key: 24
The decryption message: nmqqc wxiaevx aew e qer als oria xss qygl.
Key: 25
The decryption message: onrrd xyjbfwy bfx f rfs bmt psjb ytt rzhm.
wlan-napt-002:Lab1 thoatran$
```

Problem 2:

The key of the cipher text is: (with the upper case character is the letter in cipher text and the lower case character in the below line is the corresponding letter in the plaintext)

A	B	C	D	E	F	G	H	I	J	K	L	M
b	g	e	p	s	o	n	m		l	k		i

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
h	y		a	t		u	w	f	r	d	v	c

Problem 3:
(a) Implement a C function that performs frequency attacks on a mono-alphabetic substitution ciphers. This function should take a cipher text string compute a histogram of the incidence each letter and return a list of Paris sorted by incidence percentage.
The implementation is in the Prob3_skeleton.c
(function struct incidence_pair getIncidence(char *ciphertext))

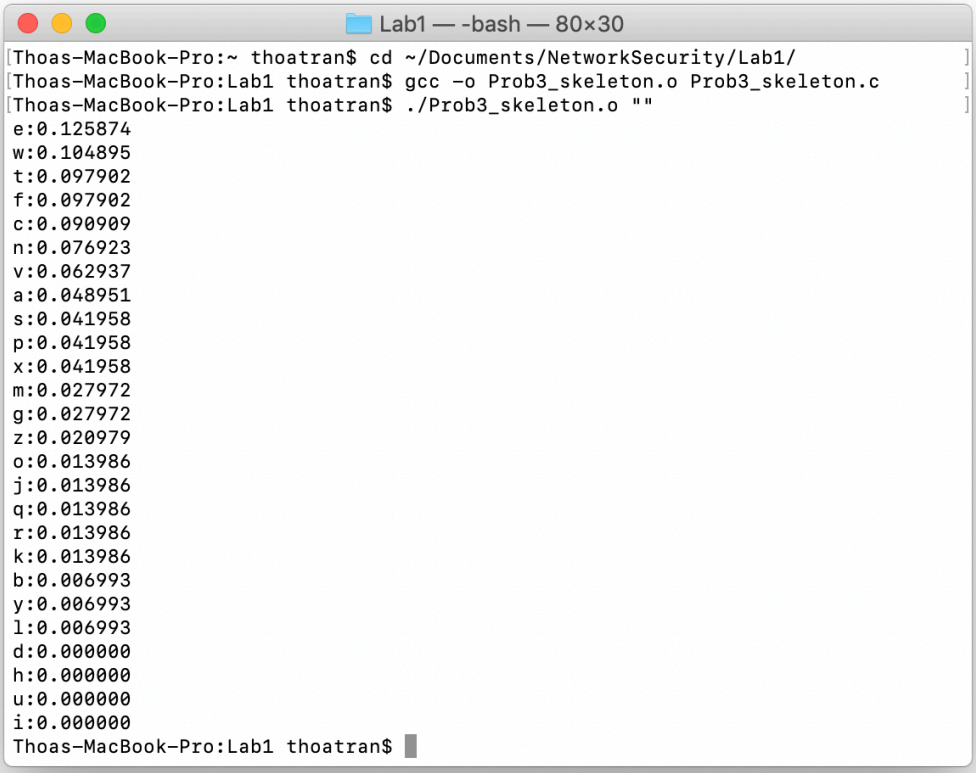
(b) Implement a C function that takes a partial mono-alphabetic substitution (i.e., subs in Problem 2) and a ciphertext and returns a potential plaintext.
The implementation is in the Prob3_skleton.c
(function char *monoalphebetic_substitution(char *ciphertext, char *subs))

(c) Use functions from (a) and (b) to decrypt the following cipher text:
"ztmn pxtne cfa peqef kecnp cjt tmn zcwsenp ontmjsw ztnws tf wsvp xtfwvfefw, c feb fcwvtf, xtfxevqea vf gvoenwk, cfa aeavxcwea wt wse rntrtpvwvtf wscw cgg lef cne xnecwea eymcg."

Run the file Prob3_skeleton.c by the following command lines:

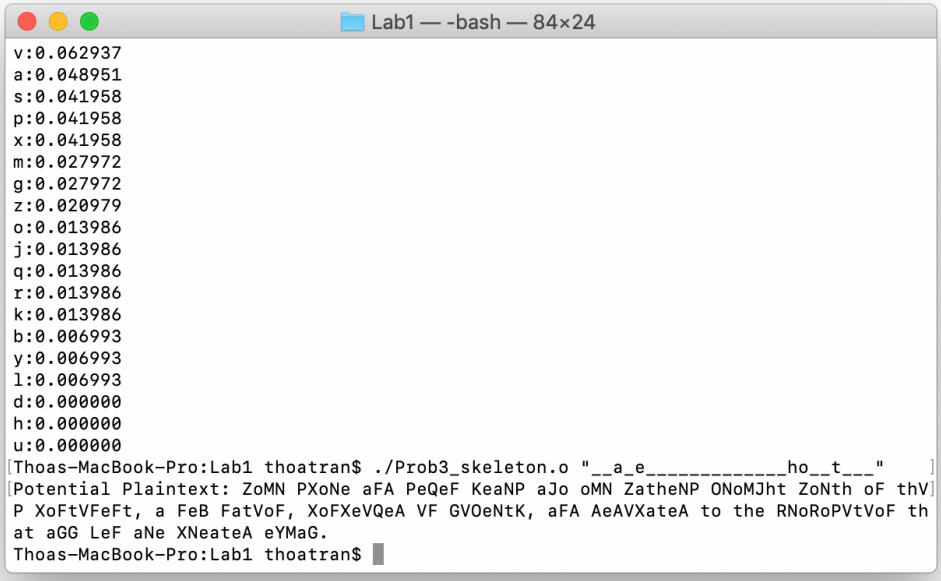
```
gcc –o Prob3_skeleton.o Prob3_skeleton.c
./Prob3_skeleton.o <the substitution string>
```

Firstly, run the program with the input of the subs (string) = “” to see the frequency of each character in the ciphertext:



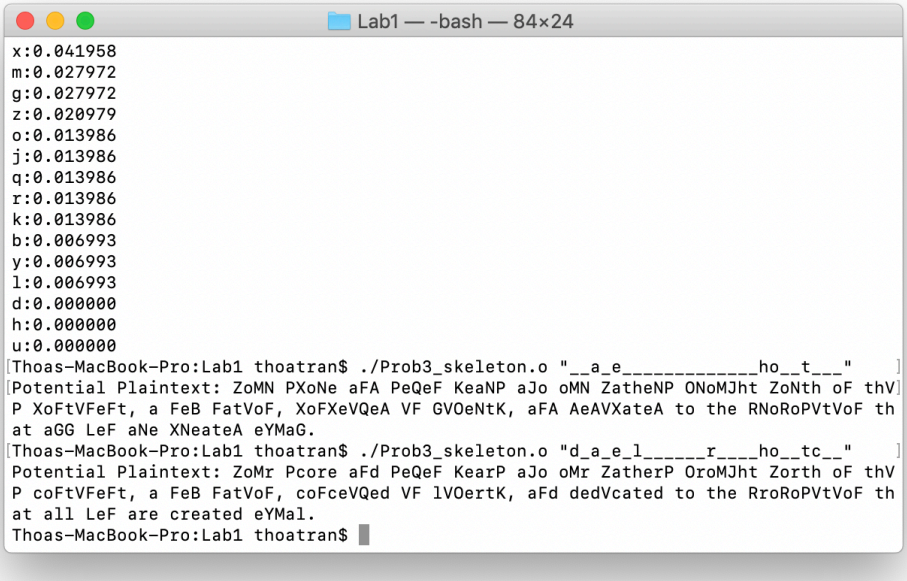
Then, input the value of the subs as a 26-character string where the character at position i is the substitution of ith character of the alphabet, OR an underscore '_' if the corresponding substitution is unknown.
We can see that the character has the highest value of appearance frequency is 'e' and 'e' is also the character appearing the most in English language, so we can guess that 'e' is cipher text is also the 'e' in plaintext. Moreover, notice that the letter 'c' appear alone in the cipher text, so we can guess that it is equal to the letter 'a' in plaintext. From the words 'wse' and 'wt', we also can guess that they are qual to 'the' and 'to' in plaintext respectively, or 't' -> 'w', 'o' -> 't', 'h' -> 's'
Therefore, we have the next command line

```
./Prob3_skeleton.o "__a_e_____________ho__t___"
```



Look at the plaintext we have now, we can see the three-letter work 'aGG' and three letter 'aNe', so 'n' in cipher text is most probably 'r' in plaintext and 'g' is cipher text is most probably 'l' in plaintext. The, look at the work 'XNeateA', we also can guess that it is equal to the word 'created' or in plaintext 'c' -> 'x', 'd' -> 'a'. We have the next command line:

```
./Prob3_skeleton.o "d_a_e_l_____r____ho__tc__"
```

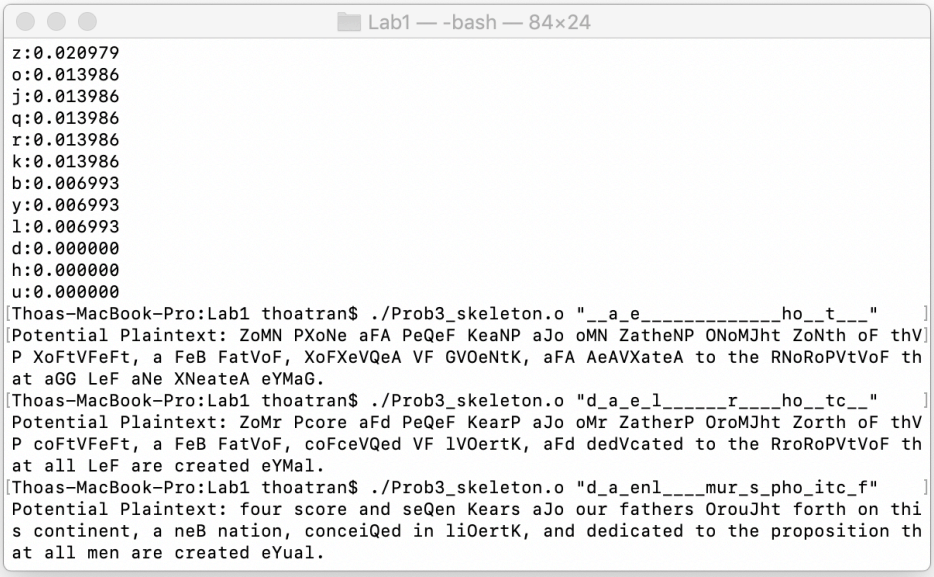


After running the command line, we can see that:

+ From the words ‘aFd’ and ‘LeF’, we can guess that in plaintext, ’n’ -> ‘f’, ‘m’ -> ‘l’.
+ From the word ‘dedVcated’, in the plaintext, ‘i’ -> ‘v’
+ Then, from the words ‘RroRoPVtVoF’ and ‘thVP’, we can guess it is equal to the words ‘proposition’ and ‘this’ respectively, or in the plaintext, ‘p’ -> ‘r’, s’ ->’p’
+ From the word ‘oMr’, in the plaintext ‘u’ -> ‘m’
+ From the word ‘Zorth’ and ‘ZatherP’, ‘z’ is most probably ’f’ in plaintext

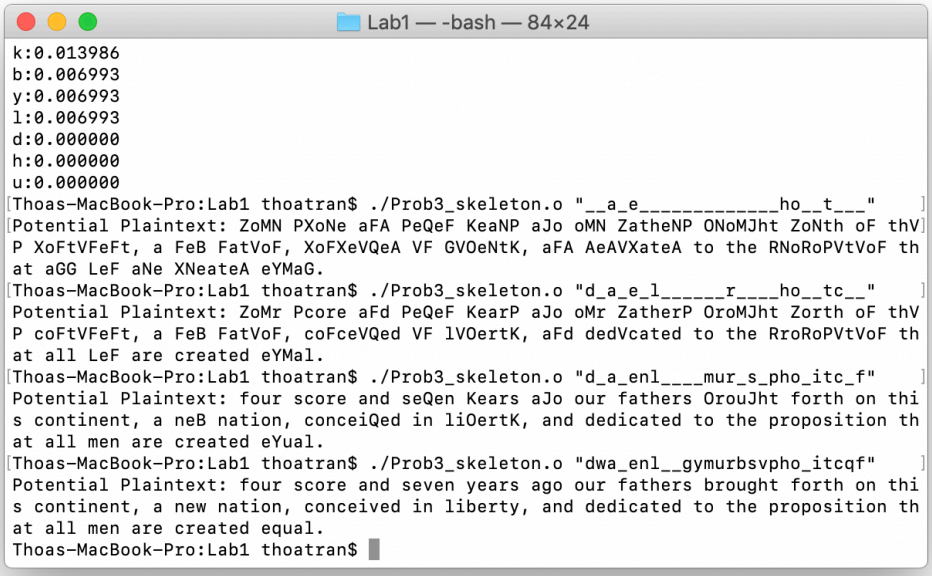
Therefore, run the below command line:

```
./Prob3_skeleton.o "d_a_enl____mur_s_pho_itc_f"
```



Look at the new text we have after running the command line now, we can guess that in the plaintext, ‘w’ -> ‘b’, ‘v’->’q’, ‘q’->’y’, ‘y’->’k’, ‘g’->’j’,’b’->’o’, so the next command line will be:

```
./Prob3_skeleton.o “dwa_enl__gymurbsvpho_itcqf"
```



After that, the plaintext we get is:

```
four score and seven years ago our fathers brought forth on this
continent, a new nation, conceived in liberty, and dedicated to the
proposition that all men are created equal.
```