Computational Problem Solving I
CPET-121
Coding Challenge 2B : Cipher Code Decryption

Problem Overview:

Your goal for this Coding Challenge is to decrypt (decode) a message that was encoded using a variation of the Baconian Cipher. The Baconian Cipher hides a phrase within message based on the capitalization of the letters message. The algorithm for decoding the phrase hidden in a message is as follows:

1. Read the entire encoded message into a string.
2. Parse the string to remove all non-alpha characters.
3. Starting from the left end of the string, strip off the first five characters.
4. Using the uppercase / lowercase pattern (see below) determine the first letter in the phrase.
5. Repeat steps 3 & 4 until all the groups of five-characters have been decoded. Any remaining characters at the end of the encoded message are discarded.

Letter	Cipher Code	Letter	Cipher Code	Letter	Cipher Code	Letter	Cipher Code	
A	UUUUU	I	ULUUU	Q	LUUUU	Y	LLUUU	
B	UUUUL	J	ULUUL	R	LUUUL	Z	LLUUL	
C	UUULU	K	ULULU	S	LUULU	.	LLULU	period
D	UUULL	L	ULULL	T	LUULL	;	LLULL	semi-colon
E	UULUU	M	ULLUU	U	LULUU	!	LLLUU	
F	UULUL	N	ULLUL	V	LULUL	?	LLLUL	
G	UULLU	O	ULLLU	W	LULLU	0	LLLLU	
H	UULLL	P	ULLLL	X	LULLL		LLLLL	space

For example, if the encoded message is: **HUmPTY DumPTY, saT oN ThE wall**

The hidden phrase is: **EGGS** (get it, Humpty Dumpty is an egg ☺)

Walking through the algorithm:

1. Read the entire encoded message into a string.
   - **HUmPTY DumPTY, saT oN ThE wall**
2. Parse the string to remove all non-alpha characters.
   - **HUmPTYDumPTYsaToNThEwall**
3. Starting from the left end of the string, strip off the first five characters.
   - First five characters: **HUmPT**
   - Remaining encoded message: **YDumPTYsaToNThEwall**
4. Using the uppercase / lowercase pattern of the five characters, determine the first letter in the decode message.
   - **HUmPT** (UULUU) maps to letter **E**

5. Repeat steps 3 & 4 until all the groups of five-characters have been decoded. Any remaining characters at the end of the encoded message are discarded.
    - Next five characters: **YDumP**
    - Remaining encoded message: **TYsaToNThEwall**
    - **YDumP** (UULLU) maps to letter **G**

    - Next five characters: **TYsaT**
    - Remaining encoded message: **oNThEwall**
    - **TYsaT** (UULLU) maps to letter **G**

    - Next five characters: **oNThE**
    - Remaining encoded message: **wall**
    - **oNThE** (LUULU) maps to letter **S**

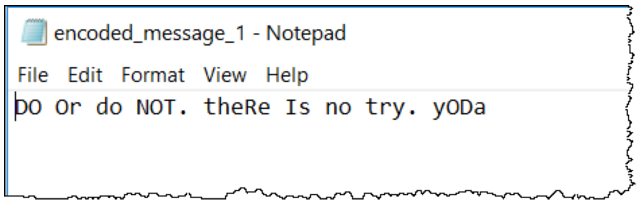
    - Remaining encoded message: **wall** is disregarded because there are less than (5) characters.

Code Design Specifications:

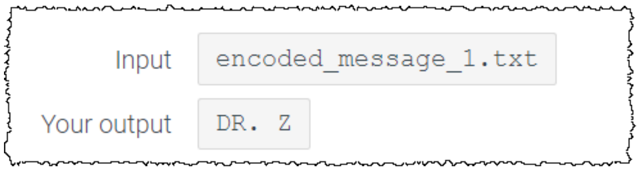
Design, code and test a procedural C++ program that decrypts a phrase hidden in a message stored in a data file.

- The program has one string input, the name of the input file that contains the encrypted message.
- The program has one output, the decoded phrase.
- Below are the three input data files provided for this coding exercise and the phrase hidden in each (i.e. the correct output).

  – File : "*encoded_message_1.txt*"

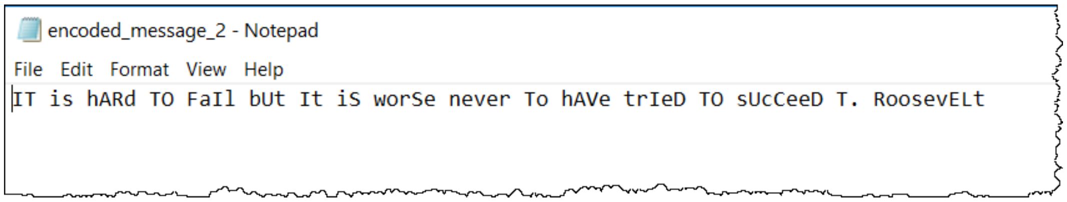


```
DO Or do NOT. theRe Is no try. yODa
```

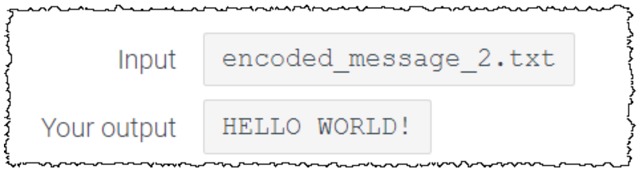


```
encoded_message_1.txt
DR. Z
```

- File : "*encoded_message_2.txt*"

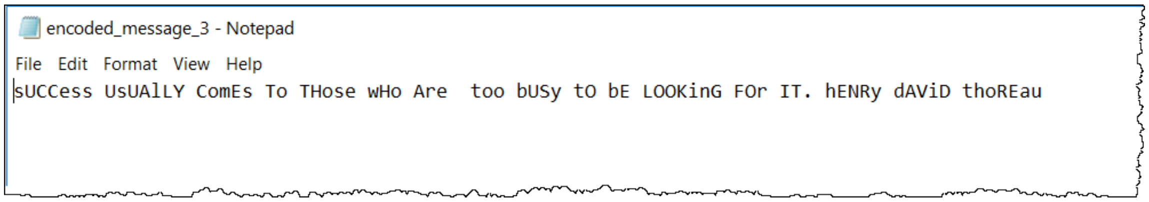


```
IT is hARd TO FaIl bUt It iS worSe never To hAVe trIeD TO sUcCeeD T. RoosevELt
```

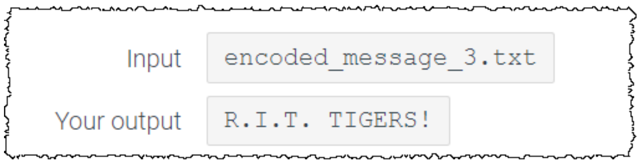


```
encoded_message_2.txt
HELLO WORLD!
```

- File : "*encoded_message_3.txt*"

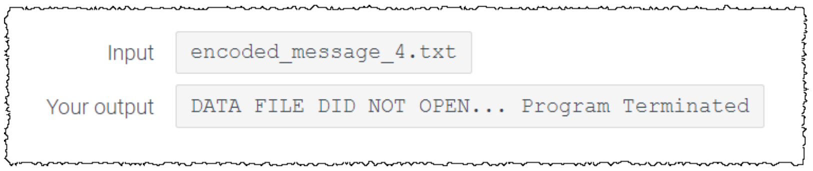


```
sUCCess UsUAlLY ComEs To THose wHo Are  too bUSy tO bE LOOKinG FOr IT. hENRy dAViD thoREau
```



```
encoded_message_3.txt
R.I.T. TIGERS!
```

- If a file name is entered for a non-existent file, the program should print an error message and terminate the program.



```
encoded_message_4.txt
DATA FILE DID NOT OPEN... Program Terminated
```

- Note, the messages in each file are of varying length. Your program must work with any message size, not just the examples shown.
- **Your program must use at least one user defined function.**
- If you prefer to do your code development outside the zyBook environment, the three data files are available for download.

Grading:

- Your grade for this Coding Challenge will be based on the complete and accurate implementation of the design specifications (80%) and adherence to proper coding style and commenting guidelines (20%).
- Any code that is found to be a fraudulent representation of your work, will receive a grade of zero.
- Any code that attempts to simply "match" the zyBook test-benches, will receive a grade of zero.
- Late assignments will be penalized 10% per day they are late. NO assignments will be accepted after Friday August 6, 2021 @ 11:59 pm.