CISC 121 Readiness: Self-Assessment Quiz

The following is a self-assessment quiz that will help you determine if you should start in CISC 101 or CISC 121. It is a copy of the first assignment given in CISC 121 in Winter 2022. If you are ready for CISC 121, you will look at this assignment and be able to figure out how you would approach the problem and code it in whatever language that you currently know. If you can do this and are comfortable with a task like this, then you are ready for CISC 121. If not, you should consider starting in CISC 101. Please note that it isn't important that you come up with an exact, correct answer (and we won't be providing one so please don't ask) – it is enough that you are able to use programming concepts and craft a solution.

Starting in CISC 101 does not put you behind in your program – you can take 101 in the fall, 121 in the winter, and 124 either in the summer online (6 week course) or in the fall of next year. You will still be able to finish your degree in 4 years.

We cannot make the 101/121 decision for you – it totally depends on your background and your abilities. We can only advise. Looking at the assignment that follows will give you a good way to evaluate where you are right now which is most important.

You can switch out of 121 and into 101 on SOLUS. If after seeing the assignment you have decided that you should move from CISC 101 to CISC 121, you may find the class full but keep watching as seats will probably open up. You will have until the end of the Add/Drop period (about the 3rd week of September) to make adjustments to your class schedule.

If you have any questions, please feel free to contact us.

The Undergraduate Admin Team
School of Computing
Queen's University

The Vigenère Cypher

Robin Dawes

January 14, 2022

In this assignment you are tasked with implementing a version of data encryption which was believed to be unbreakable for many years.

.

Background

The Vigenère Cypher was first described by Bellaso in 1553. Centuries later it was incorrectly attributed to Vigenère (a contemporary of Bellaso) and for some reason that is the name that has stuck.

Most modern authors (and spellcheck programs) prefer *cipher* over *cypher*. I prefer *cypher* - it used to be the standard spelling, and it seems more exotic now.

The Vigenère Cypher is an improvement on the much older Caesar Cypher, in which each letter of a message (or "plaintext") is replaced by the letter that is offset from the original letter by a fixed amount.

Example: In a Caesar Cypher with offset 3, each "A" in the plaintext is replaced with "D", each "B" with "E", and so on. The alphabet is considered to wrap-around, so each "X" in the plaintext is replaced with "A", etc.
So the plaintext "MYDOGHASFLEAS" would be encrypted as "PBGRJKDVIOHDV".

We will only work with plaintext that is all capital letters, and contains no spaces or punctuation.

We call the offset number the key of the cypher. To use the cypher, the sender and recipient both need to know the key.

The problem with the Caesar Cypher is that it is trivially easy to break - every instance of a common letter in English text, such as "E" and "T", will be replaced by the same letter. So finding the most common letters in the encrypted text gives strong indications of the letters of the plaintext.

The Vigenère Cypher addresses this weakness by applying multiple Caesar Cyphers to the plaintext. It is based on 26 Caesar Cyphers, represented by the rows of this table.

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

Instead of the key being a single number, the key is a word. The keyword can be of any length and can contain repeated letters. To perform the encryption, we place repeated copies of the keyword under the plaintext. Each letter of the plaintext is encrypted using the row of the table corresponding to the letter of the keyword that is under it.

I certainly don't want you to type this table into your program by hand. Think about how to create the first row using a loop, and then how to create all the other rows by applying slicing operations to the first row.

Example: Suppose the plaintext is originally "Blue blue windows behind the stars." and the given key is "moose". First we convert everything to upper case and eliminate spaces and punctuation, giving "BLUEBLUEWINDOWSBEHINDTHESTARS" and "MOOSE". We line the plaintext and keyword copies up like this:

```
BLUEBLUEWINDOWSBEHINDTHESTARS
MOOSEMOOSEMOOSEMOOSEMOOSEMOOS
```

To encrypt the first "B", we go to row "M" of the table and find the letter in the column for "B" ... which is "N".

How do we go to row "M" when all indices in Python are integers? We use the built-in **ord**() function!

For the "L" and "U" we use row "O", getting "Z" and "I".

For the "E" we use row "S", getting "W"

So the first "BLUE" is encrypted as "NZIW"

You can see that the second "BLUE" will have a different encryption because it lines up with the keyword letters "EMOO" whereas the first "BLUE" lines up with "MOOS".

The Scenario

You have been hired by a start-up called **Renaissance Technologies**. The company's mission is to re-introduce 16*th* century technology to the modern world. Many of your colleagues are working on re-inventing the spinning wheel and the astrolabe. You have been assigned a solo project: implementing state-of-the-art 16*th* century data security.

Your Assignment

You are required to write a Python 3 program that will prompt the user for some text and a keyword, and then display the Vigenère encrypted form of the text on the screen.

Where to Start

There is a link to a Python outline of a solution on the same page as this assignment file.

You are not required to follow this outline!

Acknowledging Sources

In the process of completing assignments in this course it is natural to access online resources to learn more about the problem to be solved and the techniques that might be used to solve it. I encourage you to learn from all available sources. However, when it comes to submitting your assignment you need to maintain academic integrity.

If your solution is based on ideas or code that you found on a website such as stackoverflow, you must

- State the source in a comment in your code
- Avoid copy/pasting ... learn the material/ideas, but write your own code

Programming Style

There are extensive guidelines for writing Python programs, one of the most popular of which goes by the name PEP-8. Professor Richard Linley has modified and simplified these guidelines for CISC-121. You will find these modified guidelines on the same page as this assignment file.

How You Will Be Graded

The assignment will be marked out of 100. 70% of the grade will be for correctness and 30% of the grade will be for programming style.

The grader will read your code and will run your program to test its correctness.

What to Submit

For this assignment, you are required to upload a single file called Assignment_1.py.

Your program (and all of your assignment solutions) must begin with the following docstring (with your information filled in!)

```
'''
   CISC-121 2022W

   Name:   <Your name here>
   Student Number: <Your student number here>
   Email:  <Your email here>

   I confirm that this assignment solution is my own work and conforms to
   Queen’s standards of Academic Integrity
'''
```

Things You Can Assume

You can assume:

- The plaintext will only contain letters, numbers, spaces and punctuation symbols. It will not contain letters with accents or characters that cannot be printed.
- The keyword will contain only letters.
- the plaintext and keyword may contain a mix of upper case and lower case letters.