Theory of Programming Languages

CMPT 331

Programming In The Past - 100 points

Goals

- To enjoy a simple programming assignment done in a variety of early procedural programming languages.
- To reflect on this experience though a consulting log.
- To facilitate discussions about programming and languages (and Operational Semantics).

Instructions

Develop a set of functions that will allow you to **encrypt** a string using a Caesar cipher.
Develop a set of functions that will allow you to **decrypt** a string using a Caesar cipher.
Develop a set of functions that will help you to **solve** (break) a Caesar cipher.

Implement all of the above functions for all of the following languages:

Encrypt, *Decrypt*, *Solve* in Fortran	[10 points]
Encrypt, *Decrypt*, *Solve* in COBOL	[10 points]
Encrypt, *Decrypt*, *Solve* in BASIC [1]	[10 points]
Encrypt, *Decrypt*, *Solve* in Pascal	[10 points]
Encrypt, *Decrypt*, *Solve* in Scala (in a procedural manner)	[10 points]

[1] If you like, you may, with my approval, substitute another early, procedural programming language for BASIC. Just ask me first.

Log and Commentary [50 points]

Make a prediction about how long you think it will take you to program this assignment. Write it down. Then keep a log of your work, just like you would as a consultant. The format should be similar to the following:

```
Date    Hours Spent   E-mails to Prof   Tasks/Accomplishments/Issues/Thoughts
```

Be thorough and descriptive in your log. Sum the hours spent when you are finished. Note your original prediction on the log. Write a paragraph or two to explain the discrepancy. (It will likely be huge.)

Finally, and **most importantly**, keep a running commentary ("Dear Diary…") about your thoughts and experience with each language, including how each language is similar or dissimilar to the others. Tell me in **great detail** about your thoughts on each language regarding its *readability* and *writability*, and what you loved and hated about each. Include a list of your Google searches, as I find that fascinating. When you're done, rank the languages.

This is, **by far**, my favorite part of this assignment. I look forward to reading your thoughts, searches, and comments, so be thoughtful, thorough, and amusing; impress me.

Submitting

Make a PDF of your consulting log, commentary, source code, and output of (very) thorough test runs. Be sure that it's all **one** PDF document. Print it out and hand it in **on** or **before** the class in which it is due. Remember to include your name.

Examples

on the next page

Theory of Programming Languages

CMPT 331

Programming In The Past

Examples

The usage for **encrypt** and **decrypt** should be as follows:

```
encrypt(str, shiftAmount)
decrypt(str, shiftAmount)
```

Pascal code fragment:

```
var
   x: string;
   y: string;
x := encrypt("Operation Daybreak", 2);
writeln(x);
y := decrypt(x, 2);
writeln(y);
```

example output:

```
QRGTCVKQP FCADTGCM
OPERATION DAYBREAK
```

Things might be easier if you use only capital letters, so consider writing a "toUpper" function so you can deal with mixed-case input. It's okay if your output is all caps.

The usage for **solve** should be as follows:

```
solve(str, maxShiftValue);
```

In Pascal, again:

```
solve("HAL", 26);
```

Example output:

```
Caesar 26: HAL
Caesar 25: GZK
Caesar 24: FYJ
Caesar 23: EXI
Caesar 22: DWH
Caesar 21: CVG
Caesar 20: BUF
Caesar 19: ATE
Caesar 18: ZSD
Caesar 17: YRC
Caesar 16: XQB
Caesar 15: WPA
Caesar 14: VOZ
Caesar 13: UNY
Caesar 12: TMX
Caesar 11: SLW
Caesar 10: RKV
Caesar  9: QJU
Caesar  8: PIT
Caesar  7: OHS
Caesar  6: NGR
Caesar  5: MFQ
Caesar  4: LEP
Caesar  3: KDO
Caesar  2: JCN
Caesar  1: IBM
Caesar  0: HAL
```

Panagrams make great test cases. Here are a few from From Jonathan Hoefler:

- Mr. Jock, TV quiz Ph.D., bags few lynx.
- Jackdaws love my big sphinx of quartz.
- Mix Zapf with Veljović and get quirky Béziers.
- Wham! Volcano erupts fiery liquid death onto ex-jazzbo Kenny G.
- You go tell that vapid existentialist quack Freddy Nietzsche that he can just bite me, twice.

There are many more. Ask ChatGPT to make some for you to use as tests.

To "help" you, here is a solution in APL from Rosetta code:

```
     ∇CAESAR[⎕]∇
   ∇
[0]   A←K CAESAR V
[1]   A←'AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz'
[2]   ((,V∈A)/,V)←A[⎕IO+52|(2×K)+((A⍳,V)-⎕IO)~52]
[3]   A←V
   ∇
```