CryptoScript

A python-based Programming Language

St. Edward's University
Dr. Bilal Shebaro

COSC 3336
PROJECT

Paloma Samaniego
Alex Banachh

Table of Contents

1. Getting Started

1.1 About the Project

CryptoScript is a new programming language created by students Paloma Samaniego and Alex Banach, as a project for their Programming Languages course at St. Edward's University.
Developed on November 28th, 2016.

1.2 Introducing CryptoScript

Ciphers and cryptography have been used in many of the earliest civilizations in human history. Secrecy will never cease to be an important aspect of our lives and CryptoScript is another innovation in the world of crypto-coding. Our programming language, CryptoScript, takes the values from a comma separated values (.csv) table, and executes the code in python.

1.3 Purpose of the Language

> The purpose of our language is to ensure that collaboration of a python script will remain secret if the code is compromised. Each person contributing to the project will have a look-up-table of the typical Python commands, integers, and alphabet used. If the look up table becomes compromised (ie. The enemy acquires our code) the table can be put in any order desired, so long as it matches on each end. Security is the main purpose of our language.

2 All About Programming

2.1 Setting it Up

Programming in CryptoScript requires a Python interpreter (earlier version preferred) and our three files:

1. CryptoLang.py
2. lookuptable.csv
3. myInputFile.txt

2.2 How to Program

Following are the steps to program in CryptoScript:

1. *Open "CryptoLang.py" as your source code.*
2. *Write down (in standard Python language) the code to be decrypted.*
3. *Revise "lookuptable.csv" and write down the reference numbers of each code, starting with row number then column number (refer to section 3), in order to encrypt your code from step #2.*
4. *In "myInputFile.txt", write down the numbers per line of code with a space in between each pair.*
5. *Execute the code.*

> **Notes:** It is recommended to write down the code to be decrypted BEFORE attempting to encrypt it. Although our program handles any syntax errors, it is more effective to have a visual representation of an executable code. Please refer to section 2.3 and 3 of the manual to better understand how to read the dictionary and write down the reference numbers. Moreover, the code MUST be line by line.

2.3 CryptoScript Dictionary

Our dictionary compromises a set of 90 Python keys:

- 10 Built-in functions
- 4 Arithmetic Operators
- 7 Comparison Operators (all)
- Assignment operator

	0	1	2	3	4	5	6	7	8	9
0	null	if	else	print	input	a	b	c	d	+
1	e	f	g	h	i	j	k	l	m	-
2	n	o	p	q	r	s	t	u	v	/
3	w	x	y	z	0	1	2	3	4	*
4	5	6	7	8	9	for	while	with	get	True
5	def	:	>	<	=	"	‘	(	)	False
6	\n	\t	A	B	C	D	E	F	G	or
7	!	H	I	J	K	L	M	N	O	and
8	P	Q	R	S	T	U	V	W	X	return
9	Y	Z	.	,	[	]	{	}	'	

Each key is represented by intersecting numbers of the row and column. For instance, the “print” function has a dictionary key of 30—this number is created by looking at the intersection of row #0 with column #3.

> **Notes:** the combination of the given characters and operators can create all logical, bitwise, membership, identity, and assignment operators. For our purposes, we did not create a table with all the dictionary keys next to the Python keys, due that our list would come up to 90 values. The key in cell 99 refers to a space. For an idea, this would be a more detailed dictionary table:

null	00
if	01
else	02

…

	99

2.4 Handling Syntax Errors

CryptoScript can handle any type of errors that the actual Python interpreter does, such as misspelled built-in functions or the missing of characters. In essence, the language follows any rules that Python does, since this language translates Python’s library keys into its own dictionary keys.

For an added layer of error checking, **the language provides specific error-handling messages for the user**. For instance, when the parser looks through “myInputFile.txt” it checks to see if all values are Integers. If any other type of value is found, the an error message stating “All values must be integers” will be thrown. In addition to this, it checks that the values of each line are a string, if they are not the error message “Line must be of Type String”.

2.5 Understanding the Output

After running CryptoScript, a new python file is automatically outputted, which is instantly executed through the interpreter. Following is a step-by-step process to better understand the internal workings of the language.

After the user writes down the numbers in "myInputFile.txt" and executes the file:

1. *The program iterates through each pair of numbers and looks up the reference key from the "lookuptable.csv"*
2. *As it runs per each pair of numbers and per line of code (assuming it all validates), it will create a new python file named "newCryptoScript.py" that will reside in the program's folder.*
3. *Such newly created file will have the code decrypted in standard Python language.*
4. *The program will automatically execute the newly created file, and it will output it in the interpreter window.*

> **Notes:** Although this process may sound complicated, it is highly suggested to try out the example listed below. Moreover, have all files (source code, look up table, and your input file) open for easier access and understanding.

3. Examples

3.1 Hello World

Source code

(See Section 4.1)

lookuptable.csv

(See Section 4.2)

myInputFile.txt

11 21 21 56 54 56 55 71 10 17 17 21 56 87 21 24 17 08 55
03 56 57 11 21 21 58

OUTPUT: newCryptoScript.py

```
def main():
 foo = "Hello World"
 print (foo)
main()
```

OUTPUT

Hello World

4. Materials

4.1 Source Code (“cryptoLang.py”)

See Attachment

4.2 Text file needed (“lookuptable.csv”)

See Attachment

4.3 Text file needed (“myInputFile.txt”)

See Attachment

4.4 References

Tutorialspoint.com. "Python Basic Operators." *Www.tutorialspoint.com*. N.p., n.d. Web. 29 Nov. 2016.

Dawson, Mike. *Python Programming for the Absolute Beginner*. Boston, MA: Course Technology Cengage Learning, 2010. Print.

"Page." *UsingAssertionsEffectively - Python Wiki*. N.p., n.d. Web. 29 Nov. 2016.

"History of Cryptography." *Wikipedia*. Wikimedia Foundation, n.d. Web. 29 Nov. 2016.