COMP3491 Codes and cryptography
2020-21 Summative assessment

Maximilien Gadouleau

1 Data compression (60 marks)

The main goal is to create an encoder/decoder for lossless compression of LaTeX files. Your program must be written in Python 3.7.4 and run on Windows 10. It must take an input .tex file, and operate as follows:

- Encoding: "`python encoder.py NameOfFile.tex`", creates a file called "`NameOfFile.lz`"
- Decoding: "`python decoder.py NameOfFile.lz`", creates a file called "`NameOfFile-decoded.tex`"

The input .tex file and the decoded .tex file must have the same content! To keep it simple, you should assume that the input .tex file is standalone: no input from other .tex files, no .bib file for the bibliography, no header kept in a separate .sty file.

You need to **submit a report and your code** that will be tested for performance.

Report (30 marks) The report will clearly list the main ideas used in your encoder/decoder. You can describe up to six ideas, with up to five marks for each idea and its discussion. The report must be no more than 3 pages long, and must be typed in LaTeX (a4paper, 11 pt, margin at least 2cm).

Test of performance: compression ratio (30 marks) I will test your program on a large tex file (circa two hundred pages), say `myfile.tex`. I will first verify that `myfile-decoded.tex` is identical to `myfile.tex`; if this is not the case, then you will get 0 marks for performance. Otherwise, the marking scheme for performance is competitive: the students will be ranked according to the size of their compressed file `myfile.lz` and the student's rank will determine their marks for performance.

I will automate this part of the marking, so please make sure your program follows the requirements above. I am giving you a simple program `testEncoderDecoder.py` that takes an input tex file (say `test.tex`), and encodes it, decodes it, checks the difference between input and decoded output files, and prints the sizes of the three files `test.tex`, `test.lz`, and `test-decoded.tex`. To run it, simply do

```
python testEncoderDecoder.py test
```

2 Cryptography (40 marks)

The website `what3words.com` offers a neat way of communicating a precise address anywhere in the world. Each 3m by 3m square of land and sea on Earth is encoded as three English words separated by dots. For instance, my office (E321) is encoded as "heavy.bravo.goals".

I encrypted a what3words address as follows: I used the `des` package for python; I used DES in ECB mode of operation; the key was 8 bytes long. The outcome was (in hexadecimal)

90 34 08 ec 4d 95 1a cf ae b4 7c a8 83 90 c4 75.

It's up to you to decrypt this ciphertext and find out where that place is.

To help you with it, I am giving you the des encoder that I've used (with the same key) as an executable file `encrypt.exe`. It works as such: it takes as input a hexadecimal string (length a multiple of 8) and returns a hexadecimal string. In cmd, just go the folder the .exe is in and run

```
encrypt.exe ‘‘0123456789abcdef’’
```

to encrypt the hexadecimal string `0123456789abcdef`. You should get `a80f2c74f235484e` as output.

You need to **submit another report** (no more than 2 pages) with your attack plan and your result.

Attack plan (30 marks) Your plan to crack the code (whether successful or not), with an analysis of the time/number of tests it should take. If you did carry out the attack, also record how much time it took and which sort of machine(s) you used.

Your result (10 marks) The encrypted what3words address, and where that place actually is.