COP 4521 Spring 2020
Homework - 4

Total Points: 100
Due: Friday 04/24/2020

1 Objective

The objective for this assignment is to make sure

- You are familiar with the principles of Computer Security and Symmetric Key Encryption.
- Given the description of the Security Algorithm, you can implement the Data Encryption Standard.
- You are comfortable with Python - syntax and coding conventions, to write Pythonic code.

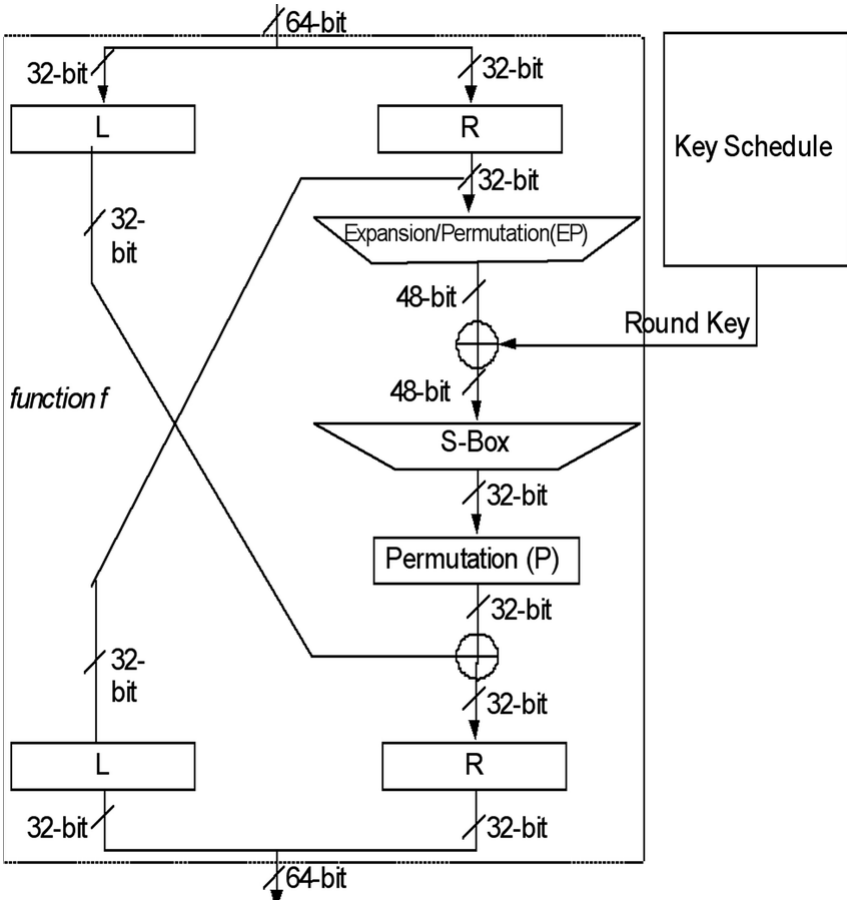
2 Specifications

Python has a `cryptography` and a `des` module, and several third party libraries that implement the DES. However, the objective of this assignment is to make sure you understand the DES algorithm by implementing it yourself. T that end, you are restricted to the standard Python libraries, and are not allowed any of the Python Cryptography libraries.

- Write a function called `encrypt`. This function will take 2 arguments, the plaintext and the key, and return the ciphertext (5 points).
- Write a function called `decrypt`. This function will take 2 arguments - the ciphertext and the key and return the plaintext (5 points).
- Write a function called `DES`. This function will take 2 arguments - a number and a key. It will them implement one iteration of DES, that includes the following:

  1. Key Scheduling using PC-2. PC-2 is a method of making a new 48 bit key from a 56 bit key using a predetermined sequence. For each round, do a cyclic left shift of the 56-bit key, and then apply PC-2. (10 points)
  2. Perform an initial permutation of the 64 bit text. (5 points)
  3. Apply 16 iterations of the round function. Each round consists of
     (a) Splitting the number into 32 bit halves. (5 points)
     (b) The left half for the next round is the right half. (5 points)
     (c) Then, apply an expansion permutation to the right half to expand it to 48 bits. (5 points)
     (d) XOR with the round key (5 points)
     (e) Use an S-box (given on slide 50 of the DEs slides) to shrink it back down to 32 bits. (5 points)
     (f) Use an intermediary permutation (5 points) item Finally, XOR with the left half to get the right half for the next round. (5 points)

  The block diagram below shows the sequence of operations in each round:



4. Perform a final permutation. (5 points)
5. The details of the inital, final, intermediary permutations and the PC-2 for key scheduling can be found here: `https://en.wikipedia.org/wiki/DES_supplementary_material`

- Both the `enrcypt` and `decrypt` functions should divide the text into 8 character (64 bit blocks) and then construct a 64 bit integer using the concatenated ASCII values of the substrings. Pad with 0's if the substring is not 8 characters long. (10 points) For example,
"Hello Wo" Should become 0100100001100101011011000110110001101111001000000101111101101111
- Apply the `DES` function to each block. For each value returned by the `DES` function, separate each 64 bit number into 8 characters and concatenate them into a string using the 8 bit numbers as ASCII values. Then, put the substring together to get the result string. This should be done for both encryption and decryption.
- Return the result string. (10 points)
- In the main function, seed the random number generator using the current system time. (5 points)
- Ask for strings (maximum length 100) from the user. For each string, generate a 56 bit key, call the functions and print the ciphertext, and the decrypted plaintext. Repeat until the User types "Exit". (5 points)
- The input will always be ASCII strings of maximum length 100. You do not have to check for that.
- Make sure you follow Python coding conventions and add comments to your code (5 points)

3 Sample Run

```
DES Implementation:
Enter text to encrypt ("Exit" to quit): This is a sample DES test
Encrypted text:  'z.J..#...M.x7.98f'
Decrypted text:  'This is a sample DES test'
Next text ("Exit" to quit): SmittyWerbenJeagerManJensen. He was number 1
Encrypted text: 'w....v"˜.? T...l.Fn e* r0'
Decrypted text: 'SmittyWerbenJeagerManJensen. He was number 1'
Next text ("Exit to quit"): Exit
```

4 General Instructions

- Add a docstring with your name, FSUID, the due date and the line "The program in this file is the individual work of <your name>".
- Name your program HW4.py
- If we have listed a specification and allocated point for it, you will lose points if that particular item is missing from your code, even if it is trivial.
- Your outputs will be different from mine due to the nature of the algorithm. As long as your program decrypts the ciphertext correctly, you are fine.
- Your program should load and run without issues. Every interpretation error will result in a loss of 5 points.
- You are restricted to standard Python (it should run on linprog).
- Testing your program thoroughly is a part of writing good code. We give you sample runs to make sure you match our output requirements and to get a general idea of how we would test your code.
- Only a file turned in through Canvas counts as a submission. A file on your computer, even if it has not been edited after the deadline, does not count.
- Please adhere to the Academic honor Policy.
- The student is responsible for making sure they have turned in the right file(s). We will not accept any excuses about inadvertently modifying or deleting files, or turning in the wrong files.
- **Program submissions** should be done through the Canvas class page, under the assignments tab (if it's not there yet I'll create it soon.) Do not send program submissions through e-mail e-mail attachments will not be accepted as valid submissions.
- The ONLY file you will submit via Canvas is HW4.py
- **General Advice** - always keep an untouched copy of your finished homework files in your email. These files will have a time-stamp which will show when they were last worked on and will serve as a backup in case you ever have legitimate problems with submitting files through blackboard. Do this for ALL programs.