Optional Exercises: DES Scheme

These spreadsheet exercises are optional, but will give you a better understanding of how both DES and Excel work. Theoretical and practical!

Both sections here contain a lot of instructions. There are some new functions which are a little trickier to use, but most of the steps themselves should be quite simple if you've been keeping up with the spreadsheets so far. Some of the cell ranges are protected this week, to avoid errors creeping in, so you may end up with some popups if you select a cell which isn't editable. If you do want to start messing around with the locked cells then the password is '`feistel`'.

Spreadsheet Exercise: DES Keys

To begin with, download the Excel spreadsheet: `03-DES.xlsx`. We will use some functions from the last two weeks, with some new ones, to create a tool to look at parts of the DES encryption scheme, including key generation.

1. First make sure you are in the worksheet name '`Keys`'.
2. In the first two rows we will generate a random 56-bit hex key, for use in DES. In Cell B1, use the RANDBETWEEN function to generate a random integer between 0 and 127. E.g., '`=RANDBETWEEN(0,127)`'. Copy this across Row 1 to Column I. These will update each time you enter new data in the spreadsheet.
3. Use the DEC2HEX function in Cell B2 to convert the random number to a 2-digit hex pair. Make sure to specify the second argument as '2'.
4. Exercise: Why do we only want numbers from 0 to 127 and not 0 to 255?
5. Use the CONCAT to stick these pairs together in Cell B3.
6. In Cell B8 there is already a hex key. Either leave this as it is, or copy your random hex into this cell. If you copy it from the highlighted cell in B3 the function will copy across as well, which means it will change every time you make a change in the spreadsheet. You can right-click and choose 'Paste Values' from the Paste Options to just keep the text.
7. In Row 10, use the MID function to split the pairs of your chosen key back out. You will need to take 2 characters at a time, which is slightly different to how we have previously used this function.

8. In Row 11 use the HEX2BIN function to change each hex pair into a 7-bit binary number. You will need to specify the second argument as '7'.

9. In Row 12 we are going to start calculating the parity bits in order to extend our 56 bits to 64 bits. To do this we need to know whether the number of 1s in each 7-bit block is odd or even. One way of doing this is to use the SUBSTITUTE function. Enter '`=SUBSTITUTE(B11,0,"")`' in Cell B12 and copy it cross to I12. This should strip out the 0s and just leave the 1s.

10. Use the LEN function in Cell B13 to count the number of 1s in Cell B12. Copy this function across to I13.

11. This step introduces the IF function, which works similarly to how it does in programming. In Cell B14, we need to add a 1 to the end of the 7-bit block in B11 if the number of 1s is odd, or a 0 if the number of 1s is even. So we can enter '`=IF(ISEVEN(B13),CONCAT(B11,1),CONCAT(B11,0))`' in Cell B14 and copy this across to I14. Make sure that the 8-bit blocks have the correct parity bit on the end.

12. In B15, use the CONCAT function to stick together the 7-bit blocks and get the 56-bit key. In B16 do the same for the 8-bit blocks to get the 64-bit key.

13. The rest of that sheet will go through all of the steps to create the first round key from the key you have produced. You don't need to do anything with it, but may find it useful to look at the functions underlying it.

Spreadsheet Exercise: DES First Round

This section will use the same spreadsheet as above, but working in the '`Expansion and Permutation`' worksheet.

1. In Cell B2 there is an 8-character plaintext word '`decrypts`'. (Exercise: Why is it 8 characters?) This is split out using the MID function in Row 3, with the corresponding ANSI/ASCII codes displayed using the CODES function in Row 4. In Row 5 the DEC2BIN function has been used to generate 8-bit blocks to represent these codes and these are stuck together in Row 6 using CONCAT to give a 64-bit block for us to encrypt. (We're not going to do a full application of DES in Excel though as software implementations generally aren't the best way to do this.)

2. The first step in DES is to perform the Initial Permutation (IP). This permutation is displayed in Row 7 and tells us how to scramble our 64-bit plaintext block. In Row 8, use the MID function to split the 64-bit block in B6, but only copy this along to the 64th position, not the end of the spreadsheet as we have previously done. The 'Start Number' for the mid function is the value in Row 7. E.g., the first bit comes from the 58th position, the second bit comes from the 50th position, and so on. The 'Number of Characters' will just be '1' to return one bit at a time.

3. We need to break this scrambled 64-bit block into a Left block and a Right block. We can use the LEFT and RIGHT functions for this. In Cell B10, use the LEFT function to take the first 32 bits from Cell B9. Use the RIGHT function to do the same for the second half.

4. In Row 12 there is another fixed list of numbers, this time including some repeats. This tells us how to 'expand' and scramble the Right half that we have just found. Use the MID function again to do this in Row 13.

5. In Row 14 the first round key has already been generated and pre-filled. (To look into where this comes from you can look further down the `Keys` worksheet.)

6. In Row 15 we need to `XOR` the bits of the expanded Right half with the first round key. In Cell B15 use the BITXOR function to `XOR` the bits from Row 13 and Row 14. This will need copying along the row to the 48th position.

7. In Row 16, the concatenated bits should display, in Row 17 these should be split into 6-bit blocks, and in Rows 18-21 these are split out to give us the corresponding rows and columns for the S-Box in decimal. Optional: Take a look at the functions here and see if you can understand what they are doing.

    - The S-Box will turn these 6-bit blocks into 4-bit blocks, using the INDEX function. We will need to point it to a fixed array, a row number, and a column number. This is tricky since the S-Boxes are stored on another sheet: the formula is given below, but try doing this with the mouse and keyboard before just copying it.
    - In Cell B22, use the INDEX function to look up the row and column of the first S-Box stored on the '`S-Boxes`' sheet: the array range is '`'S-Boxes'!$B$2:$Q$5`'. You can select this by clicking into the sheet `S-Boxes`, selecting the range, and then pressing `F4` to add the dollar signs for absolute referencing. The row number is in Cell B20 and the column number is in Cell B21: these can just be clicked on back in the `Expansion and Permutation` sheet. The formula can be copied across, but the S-Box will need to change to a different array. (See below.)
    - The full command looks like this: `=INDEX('S-Boxes'!$B$2:$Q$5,B20,B21)`. If you click the row and column number, then these may have `'Expansion and Permutation'!` in front of the cell references. For the second S-box, it will look similar but be `=INDEX('S-Boxes'!$B$8:$Q$11,B20,B21)` in order to point at the right array. This will similarly need to change for each of the boxes.
    - Each 4-bit block will use a different S-box. E.g., the first block used S1, the second block uses S2, etc.

8. In Cell B24 you should find that the concatenated 4-bit blocks are shown as a single 32-bit block.

9. You should now have a spreadsheet that performs the first round of DES, including the Initial Permutation.