DES

DES is a block cipher that operates on bits
encryption is performed on blocks of 64-bit data
the initial key passed is 64 bits
    but every 8th bit is ignored and permuted (yielding a 56-bit key)
    this 56-bit key is permuted again to yield 16 48-bit sub-keys that are used for encryption
DES results in a permutation among two possible arrangements of each of the 64-bit blocks
    and it performs 16 rounds of operations to encrypt a single block of data

padding
    when a data block is not 64 bits long, it is padded by appending some values at the end
    the most basic form of padding is to append 0's at the end

working mechanism
    assumption: 64-bit key and 64-bit data
    next, two major steps
        (1) creation of 16 sub-keys
        (2) encoding of each of the 64-bit data blocks

    creation of 16 sub-keys
        (1) from the 64-bit initial key, obtain a 56-bit key using the PC-1 table
            shifts the position of each bit, while ignoring every eighth bit
        (2) split the 56-bit key into two equal halves: C_i, D_i
        (3) use the Left-Shifts table to iterate 16 times while shifting the bits in both C_i and D_i
        (4) merge the shifted C_i and D_i get $C_i D_i$, where $i = 0 \ldots 16$
        (5) for each of the 16 merged keys, use the PC-2 table to get 16 sub-keys to use for encryption
            note that the PC-2 table only uses 48 bits (which means 8 bits are ignored)
        (6) at this point, we have 16 48-bit keys (each key is sequentially used for each round)

    encoding of each of the 64-bit data blocks
        (1) based on the IP table, rearrange the message
        (2) separate the permuted message into two equal blocks L_0 and R_0
            each of these blocks are 32-bits long
        (3) for each of the n=16 rounds, the left and right blocks are calculated as follows:
            (a) $L_n = R_{n-1}$
            (b) $R_n = L_{n-1} \oplus f(R_{n-1}, K_n)$

            the left block for each round is the right block calculated from the previous round
            the right block for each round is the XOR of the left block from the previous round
                with the result of a function that takes in the right block from the previous round
            what's this function?
                first, R_{n-1} is expanded by using an E-Bit selection table
                a 32-bit block is expanded to a 48-bit block (some bits are repeated in the table)
                e.g.:
                $R_0 = 1010\,1010\,0010\,1010\,1010\,1010\,0101\,0010$
                after expansion:
                $R_0 = 010101010100\,000101\,010101010101\,010100\,001010\,100101$

                after using the E-Bit selection table, we XOR the expanded right half
                    with the 48-bit sub-key of the current round

result: 8 blocks (B) of 6 bits; e.g.:

$R_0 = B_1 B_2 B_3 B_4 B_5 B_6 B_7 B_8$ (48 bits)

next, each of the 6-bit blocks are run through S-boxes to get 4-bit blocks
the first and last bits represent the row; the middle four bits represent the column
e.g.:

$B_0 = 010101$

first and last bits = 01 = 1 (so row 1)
middle four bits = 1010 = 10 (so column 10)
on the S1 (S-box 1) table (at row 1, column 10) = 12 (1100)
so 010101 yields 1100
each 6-bit block is run through a different S-box:

$R_0 = S_1(B_1) S_2(B_2) S_3(B_3) S_4(B_4) S_5(B_5) S_6(B_6) S_7(B_7) S_8(B_8)$ (32 bits)

once the S-box operations are finished, we get a 32-bit block from a 48-bit block

after the S-box operations, a permutation table P is used to get the final value of f

(4) the output of the function is XOR'ed with L_{n-1}
(5) everything before this point is repeated 16 times, sequentially
(6) after the final round, the left and right blocks are interchanged
i.e., $L_{16} R_{16}$ becomes $R_{16} L_{16}$
(7) finally, the final permutation is performed using the IP^{-1} table
(8) the output is the ciphertext for the initial 64-bit block data/plaintext

to decrypt the data, the same procedure is undertaken (in reverse order)

security of DES

mainly depends on confusion and diffusion (i.e., substitutions and shifting)
don't use DES since technology has been developed to break any DES within a reasonable time

3DES

just implements DES three times with a new key for each iteration of DES
this makes the cryptosystem slightly more secure
however, computing this takes a long time (so it is also not used)