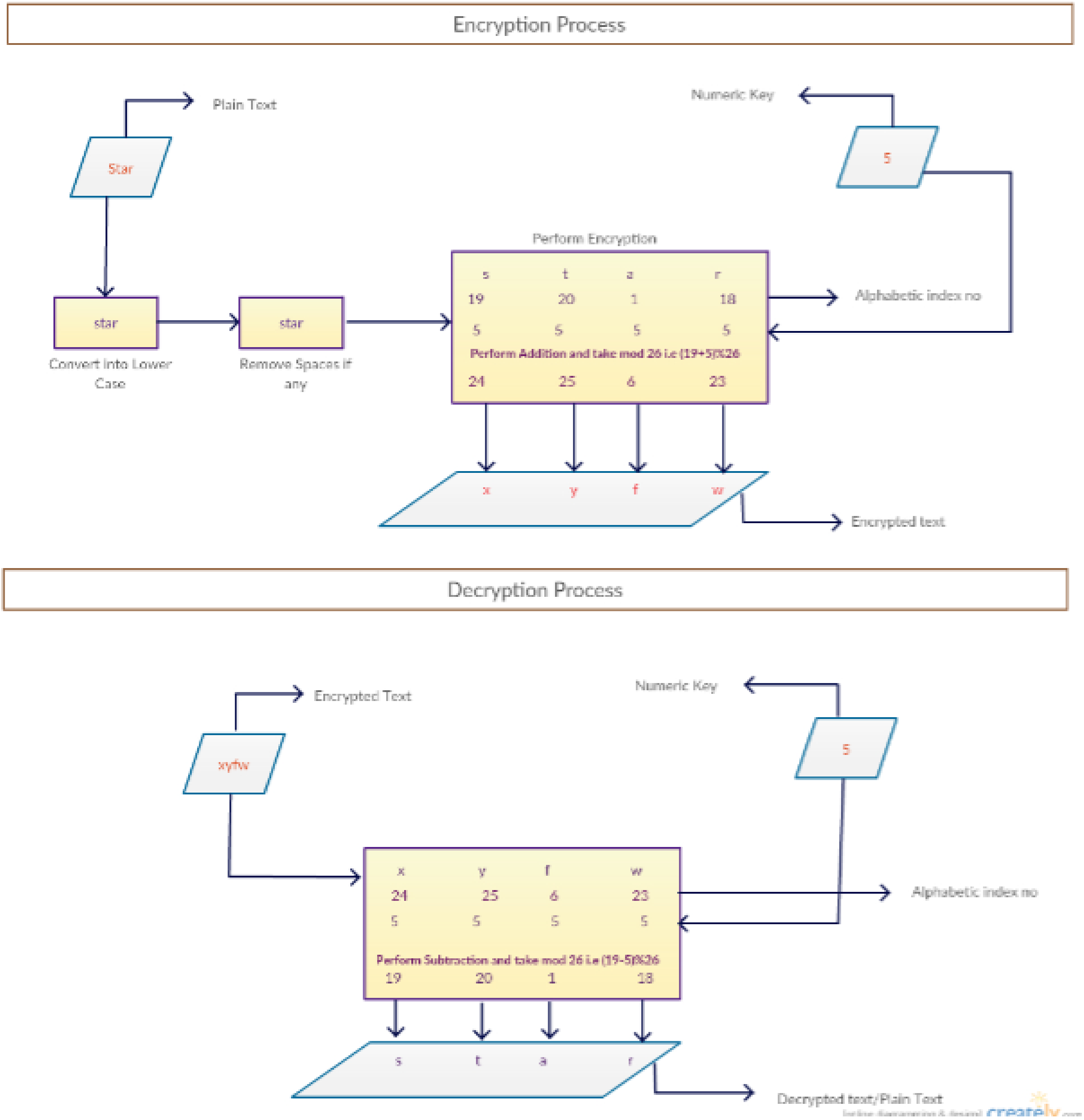
Vier De-Cipher Plus

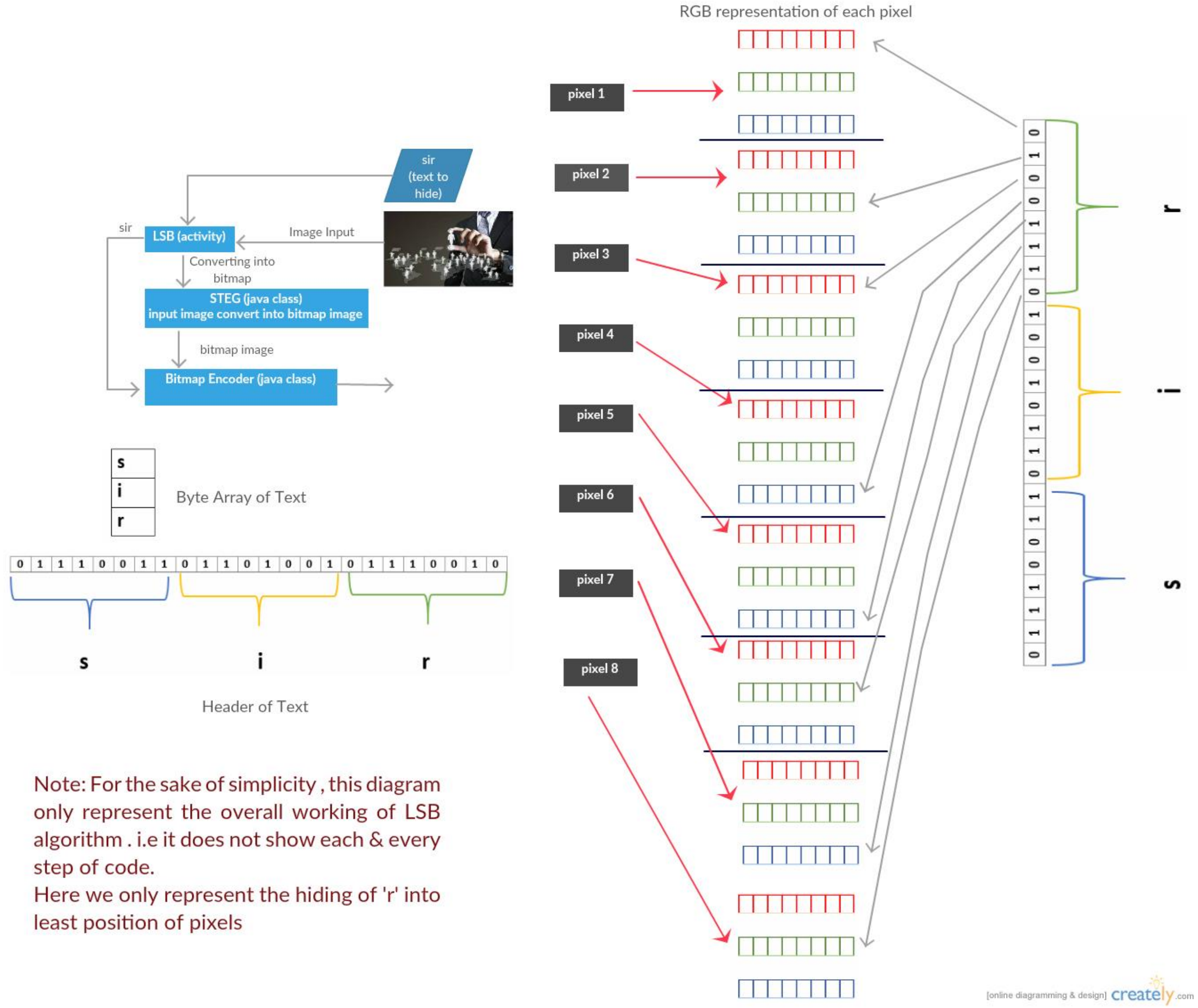
Working Diagrams of Algorithms

Prepared By: Fatima Iqbal & Insha Siddiqui

Ceaser Cipher Algorithm



Least Significant Bit Algorithm



Note: For the sake of simplicity , this diagram only represent the overall working of LSB algorithm . i.e it does not show each & every step of code.
Here we only represent the hiding of 'r' into least position of pixels

Figure 1 Text Hide

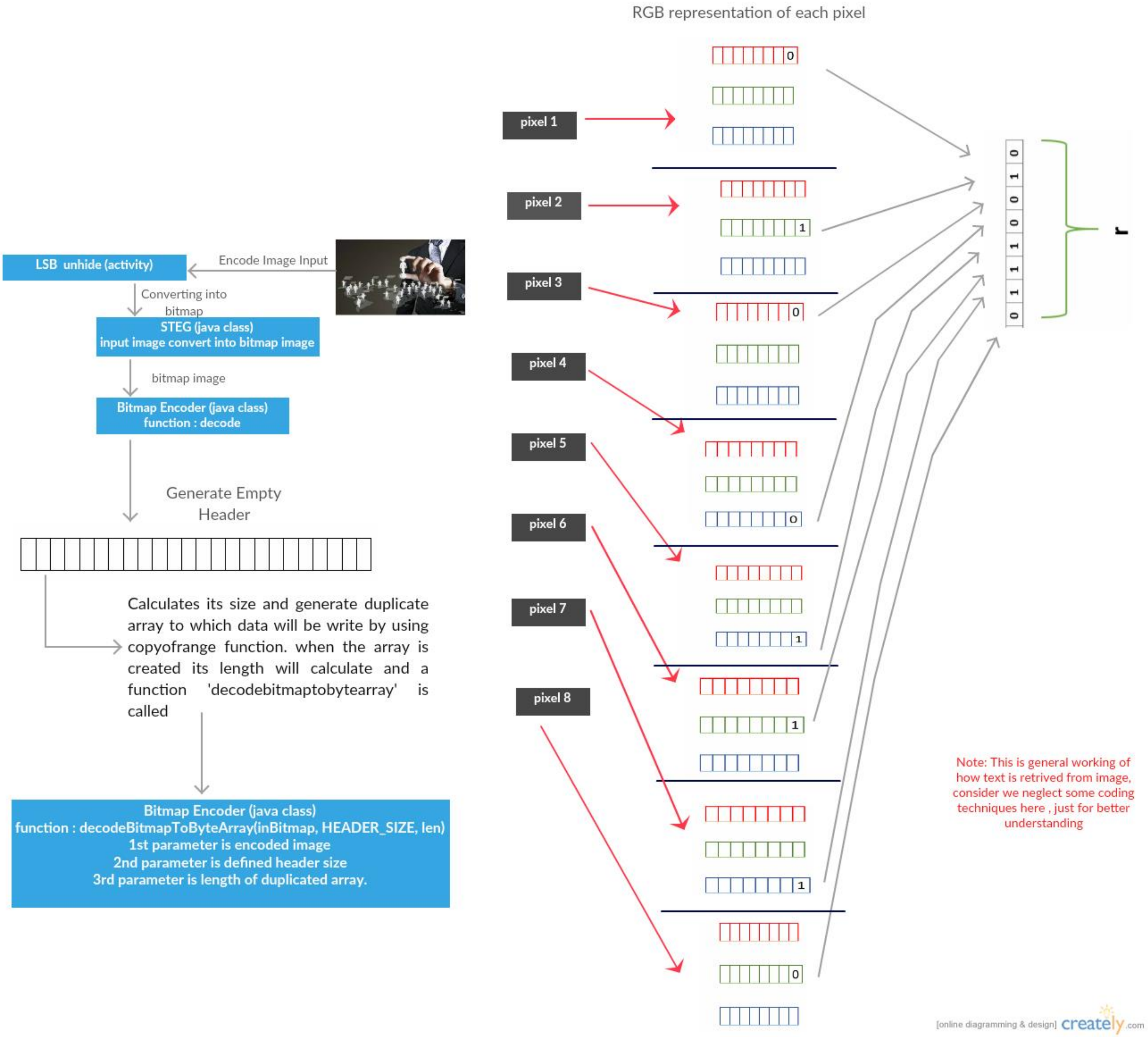
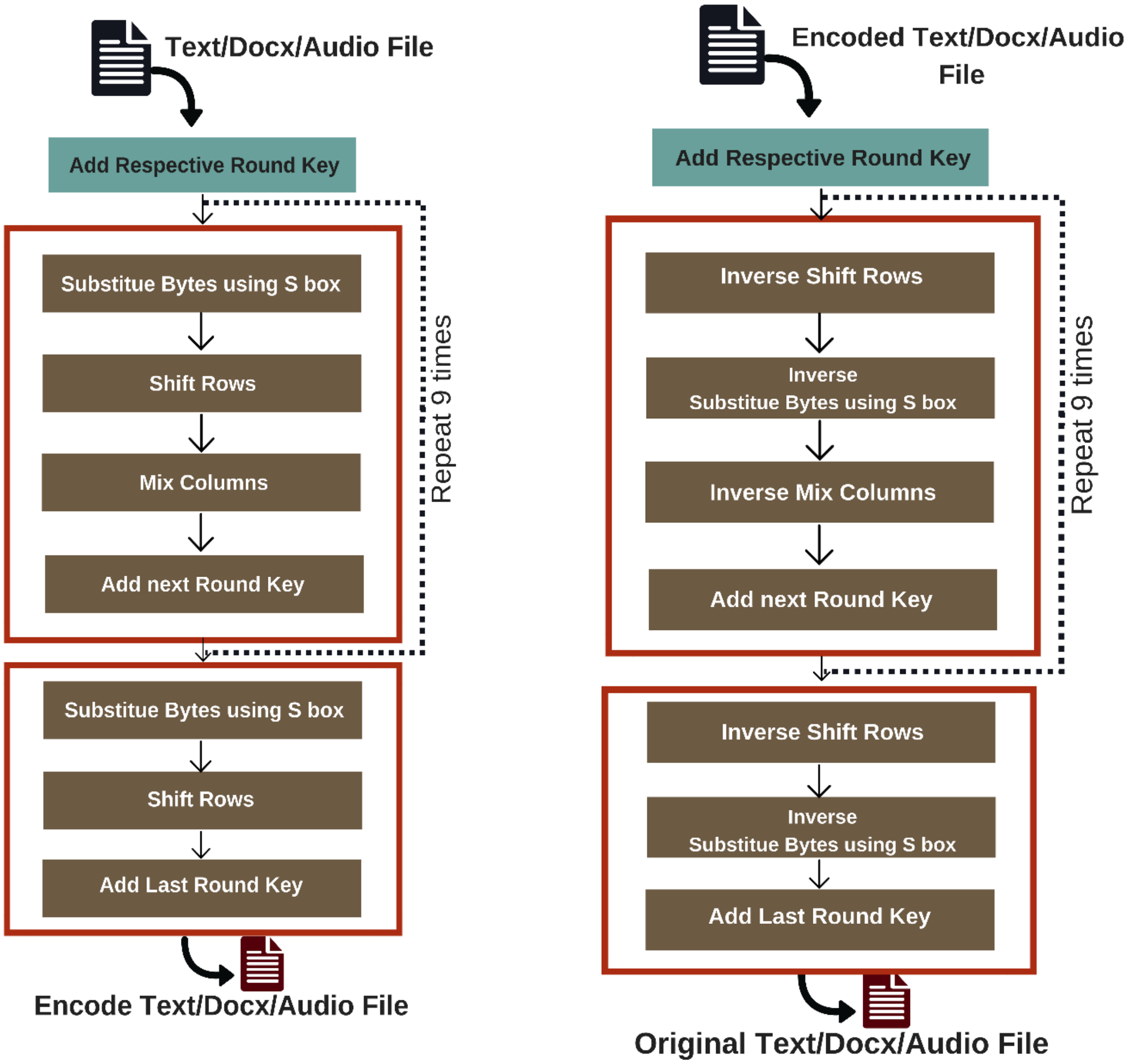

RGB representation of each pixel
LSB unhide (activity)
Encode Image Input
Converting into bitmap
STEG (java class)
input image convert into bitmap image
bitmap image
Bitmap Encoder (java class)
function : decode
Generate Empty Header
Calculates its size and generate duplicate array to which data will be write by using copyofrange function. when the array is created its length will calculate and a function 'decodebitmaptobytearray' is called
Bitmap Encoder (java class)
function : decodeBitmapToByteArray(inBitmap, HEADER_SIZE, len)
1st parameter is encoded image
2nd parameter is defined header size
3rd parameter is length of duplicated array.
pixel 1
pixel 2
pixel 3
pixel 4
pixel 5
pixel 6
pixel 7
pixel 8
0
1
0
0
1
1
1
0
0
1
0
0
1
1
1
0
r
Note: This is general working of how text is retrived from image, consider we neglect some coding techniques here , just for better understanding
[online diagramming & design] creately.com

Figure 2 Text Unhide

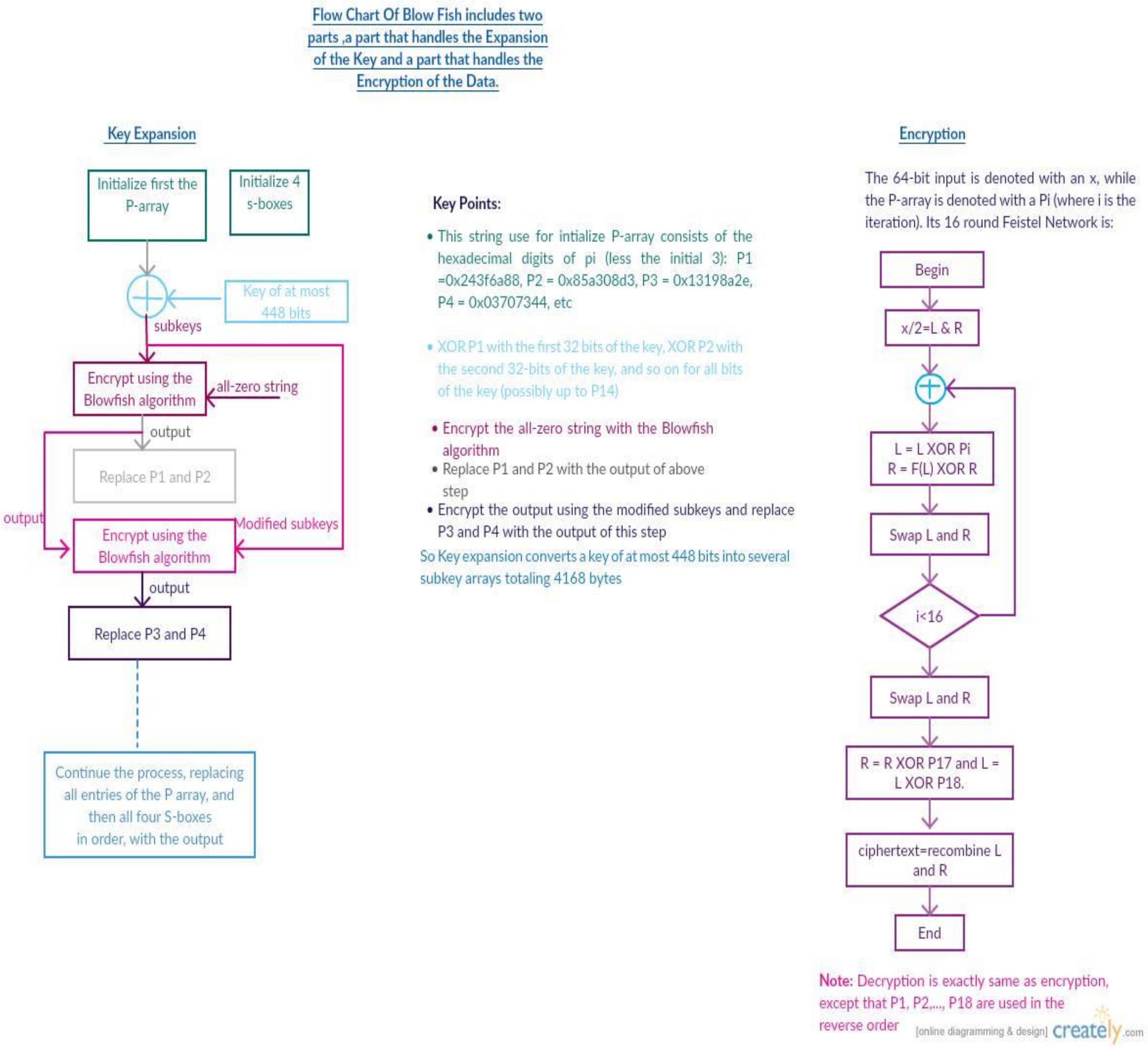
AES Encryption Decryption



For better understanding of each and every step, kindly do visit this link.

https://kavaliro.com/wp-content/uploads/2014/03/AES.pdf

Blow Fish Algorithms



Flow Chart Of Blow Fish includes two parts ,a part that handles the Expansion of the Key and a part that handles the Encryption of the Data.

Key Expansion

Initialize first the P-array
Initialize 4 s-boxes
Key of at most 448 bits
subkeys
Encrypt using the Blowfish algorithm
all-zero string
output
Replace P1 and P2
output
Modified subkeys
Encrypt using the Blowfish algorithm
output
Replace P3 and P4
Continue the process, replacing all entries of the P array, and then all four S-boxes in order, with the output

Key Points:

- This string use for intialize P-array consists of the hexadecimal digits of pi (less the initial 3): P1 =0x243f6a88, P2 = 0x85a308d3, P3 = 0x13198a2e, P4 = 0x03707344, etc
- XOR P1 with the first 32 bits of the key, XOR P2 with the second 32-bits of the key, and so on for all bits of the key (possibly up to P14)
- Encrypt the all-zero string with the Blowfish algorithm
- Replace P1 and P2 with the output of above step
- Encrypt the output using the modified subkeys and replace P3 and P4 with the output of this step

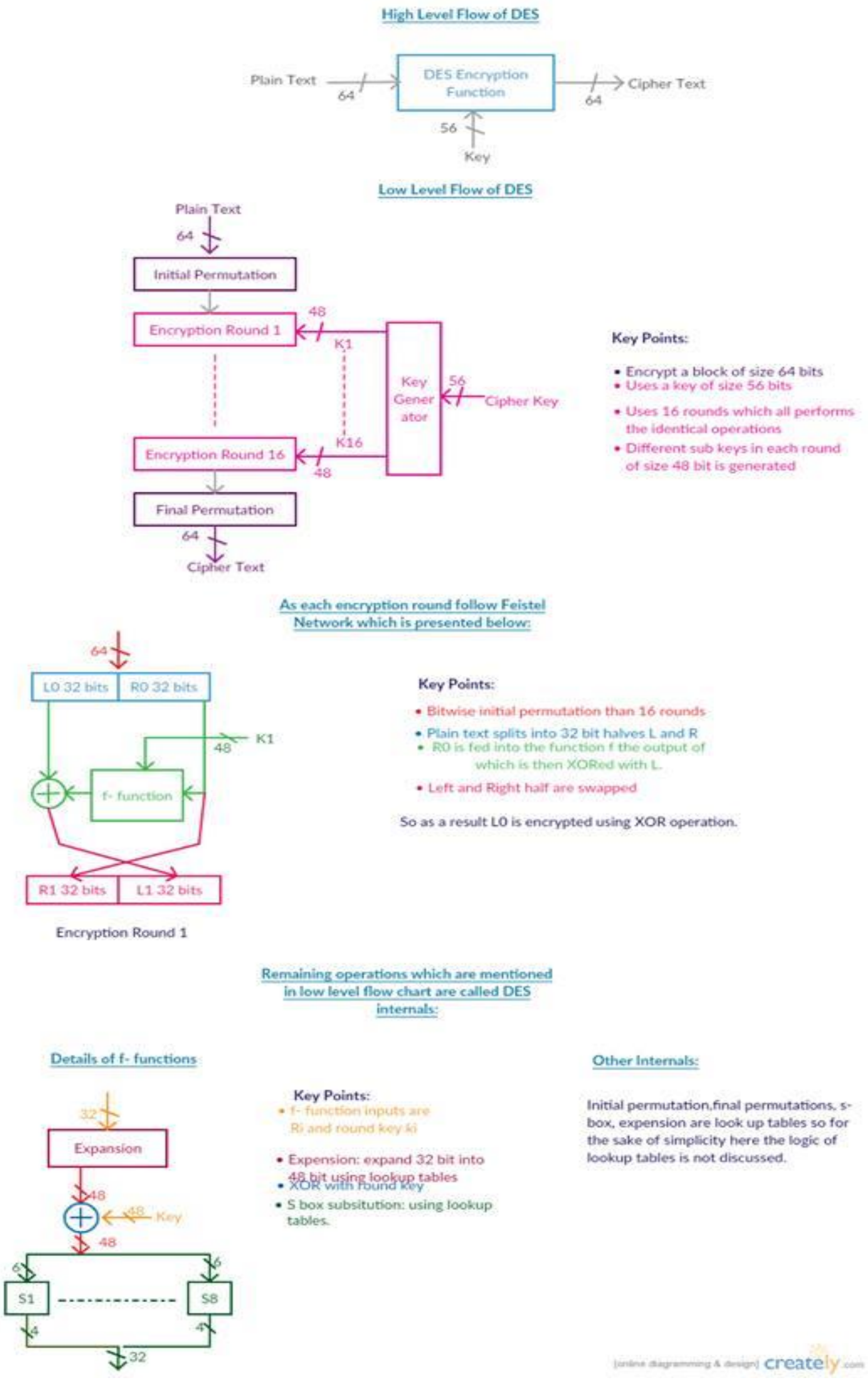
So Key expansion converts a key of at most 448 bits into several subkey arrays totaling 4168 bytes

Encryption

The 64-bit input is denoted with an x, while the P-array is denoted with a Pi (where i is the iteration). Its 16 round Feistel Network is:

Note: Decryption is exactly same as encryption, except that P1, P2,..., P18 are used in the reverse order

Data Encryption Standard [DES] Algorithm



High Level Flow of DES

Low Level Flow of DES

Key Points:

- Encrypt a block of size 64 bits
- Uses a key of size 56 bits
- Uses 16 rounds which all performs the identical operations
- Different sub keys in each round of size 48 bit is generated

As each encryption round follow Feistel Network which is presented below:

Encryption Round 1

Key Points:

- Bitwise initial permutation than 16 rounds
- Plain text splits into 32 bit halves L and R
- R0 is fed into the function f the output of which is then XORed with L.
- Left and Right half are swapped

So as a result L0 is encrypted using XOR operation.

Remaining operations which are mentioned in low level flow chart are called DES internals:

Details of f- functions

Key Points:

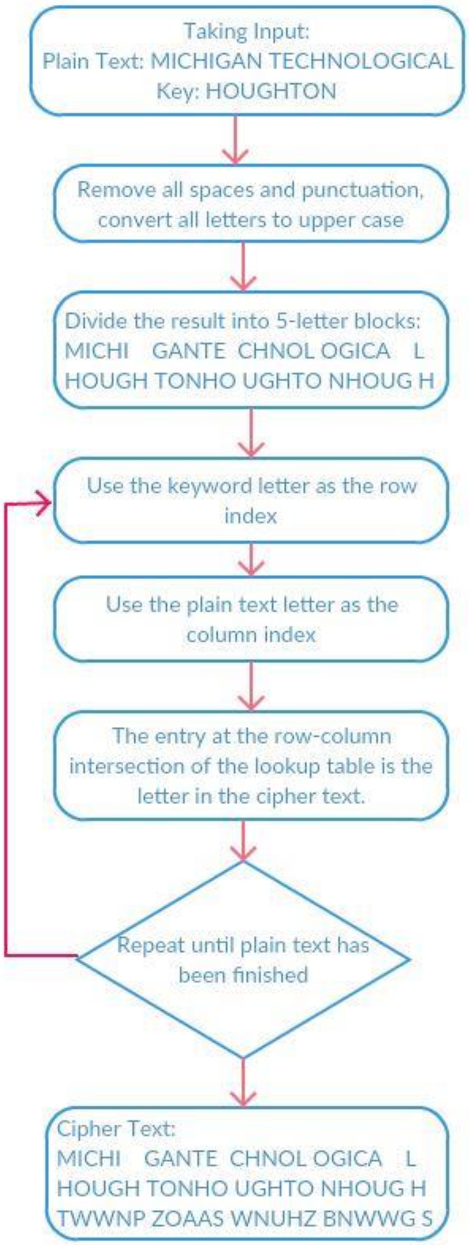
- f- function inputs are Ri and round key ki
- Expension: expand 32 bit into 48 bit using lookup tables
- XOR with round key
- S box subsitution: using lookup tables.

Other Internals:

Initial permutation,final permutations, s-box, expension are look up tables so for the sake of simplicity here the logic of lookup tables is not discussed.

Vigenere Cipher Algorithm

Vigenère Cipher Encryption
Flow Chart



Taking Input:
Plain Text: MICHIGAN TECHNOLOGICAL
Key: HOUGHTON

Remove all spaces and punctuation, convert all letters to upper case

Divide the result into 5-letter blocks:
MICHI GANTE CHNOL OGICA L
HOUGH TONHO UGHTO NHOUG H

Use the keyword letter as the row index

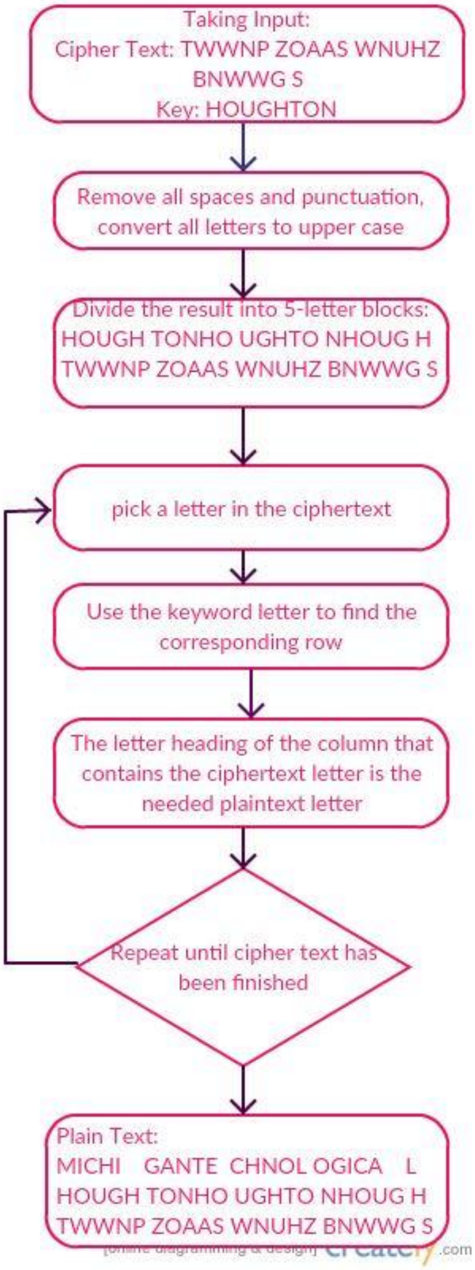
Use the plain text letter as the column index

The entry at the row-column intersection of the lookup table is the letter in the cipher text.

Repeat until plain text has been finished

Cipher Text:
MICHI GANTE CHNOL OGICA L
HOUGH TONHO UGHTO NHOUG H
TWWNP ZOAAS WNUHZ BNWWG S

Vigenère Cipher Encryption
Flow Chart



Taking Input:
Cipher Text: TWWNP ZOAAS WNUHZ BNWWG S
Key: HOUGHTON

Remove all spaces and punctuation, convert all letters to upper case

Divide the result into 5-letter blocks:
HOUGH TONHO UGHTO NHOUG H
TWWNP ZOAAS WNUHZ BNWWG S

pick a letter in the ciphertext

Use the keyword letter to find the corresponding row

The letter heading of the column that contains the ciphertext letter is the needed plaintext letter

Repeat until cipher text has been finished

Plain Text:
MICHI GANTE CHNOL OGICA L
HOUGH TONHO UGHTO NHOUG H
TWWNP ZOAAS WNUHZ BNWWG S