Oppgave 2

Dokumentasjon A)

Fremgangs metoden min i Oppgave A var at jeg begynte med å skrive et helt vanlig script med å åpne filene jeg skulle bruke, og validere at de faktisk åpnet seg. deretter brukte jeg

```
fread(pInputfile, sizeof(char), 2, szPair);
fprintf(pOutputfile, "%c", hexToInt(szPair));
```

Dette var for å lese 2 og 2 bytes fra filen min, for å så skrive de direkte til ny fil og konvertere 2 chars (hex) til Int i samme sleng. Det jeg fant ut at etter noen timer med debugging, var at fread hentet ut '\n' også. Hele programmet mitt ble shiftet med 1 byte ned hver gang '\n' kom, det førte til mange rare tegn jeg kunne klart meg uten. Etter noen timer med arbeid kom jeg frem til en annen løsning der jeg burker fgetc og husker på prev og current char, for å så validere (at det ikke er '\n'), og så skrive til fil. Da funket det!

```
char chCurrent, chPrev;
char* szPair;
int count = 0;
// While File has chars.
while((chCurrent = fgetc(pInputFile)) != EOF ) {
    // I want to skip all if the char is \n, else it will bring problems.
    if ( chCurrent == '\n') {
        continue;
    }
    // adding Count to see when we have 2 chars to combine.
    count++;
    if ( count % 2 == 0 ) { // We GOT 2 chars.
        // Sending in both chars to combine function. and then write it to file.
        szPair = combineTwo(chPrev, chCurrent);
        fprintf(pOutputFile, "%c", hexToInt(szPair));
    } else {    // End if count % 2 == 0.
        chPrev = chCurrent;
    }   // End else count % 2 == 0
}   // End while getc(pInput);
```

Dokumentasjon B)

I oppgave b, lagde jeg meg 2 funksjoner, en for å validere tegnene til å være a-zA-z, også gjøre alle til uppercase. Deretter sendte jeg de til en annen funksjon som tok inn bokstaven og % 65, sånn at alle tallene ble mellom 0 - 25, og deretter incrementet jeg bare plassen i ett array.

```
void countOccorances(int *piSet, char chCharacter) {
    // checking if its actually characters we have, NO space, signs or numbers.
    if ( validateChar(chCharacter) == OK ) {
        chCharacter = toupper(chCharacter); // converting it to uppercase
        piSet[chCharacter % 65]++;  // Taking % 65 to make uppercase into 0 - 25.
    }
}

int validateChar(char chChar) {  // a-zA-Z0-9
    if ( chChar >= 65 && chChar <= 90 ) // Checking for uppercase letters.
        return OK;
    else if ( chChar >= 97 && chChar <= 122) // Checking for lowecase letters.
        return OK;

    return ERROR;
}
```