Amro Elbahrawy
4022176o
COMP 6481
Assignment 1

Part 1:

(1)

a) input: the input text (string) P, k, (char)
output: encrypted text (string)

```
for loop (i=0 to length of P){
    C = P[i] +(k+i)(mod 3)
    print C
}
```

b) input: C (string), k (char)
output: original text P

```
for loop (i=0 to length of C){
    P = C[i] - (k+i)(mod3)
    print P
}
```

c) Time complexity is O(n) for both algorithms.

d) Space complexity is also O(n) for both algorithms.

Q2)

Input: Array of integers A, int X
output: two ints, SumG, SumL

```
int SumG, SumL

for loop(i=0, i< A.length(), i++){
    if (A[i] > X){
      add A[i] to SumG }
    if (A[i] < X) {
      add A[i] to SumL }
}
  print SumG, SumL
```

a) time complexity is O(n)

b) space complexity is also O(n) because of array storage.

Q3) Input: Sorted array A, int X
output: SumG, SumL

```
current = middle index
while(A[current] = X){
  current = floor(current + (A.length/2))
}
if(A[current] > X){
    loop(j=0; A<current){
      add A[j] to SumL, continue if A[i] = X
    }
    loop (K=current; K< A.length){
      add A[K] to SumG
    }
}
```

Q3 cont.)

```
else if(A[current]< x){
    loop( i=0 ; i < current +1 ) {
      add  A[i]  to  sumL
    }
    loop ( j=(current +1); j< A.length ) {
      add A[j] to SumG, Continue if equal
    }
  print SumG, SumL;
```

a) it would be $\Theta(n\log(n))$ in the worst case, because we are searching and summing. The search can take $\log(n)$ in the worst case.

b) space complexity is $O(n)$

Q4) input : Array of ints A
output : A modified

```
if(A.length is odd)
   mid = floor(A.length/2)
else
    mid = (length/2)

for(int i=0; i< mid-1){
   temp= A[i];
   A[i]= A[i+1];
   A[i+1] = temp;
}
if(A.length is odd)
   mid++;
for(i = mid ; i<A.length-1) {
   A[i+1] = A[i] + A[i+1];
}
return A;
```

b) Time complexity is O(n) because both loops traverse only half of the array

c) Space complexity is O(n) since the array and 2 more constants are required to store data.