Q1) Reverse a String

```
#include <stdio.h>
#include <string.h>
int main() {
    char s[5] = "hello";
    int len = strlen(s);
    char rev[10];
    for (int i = 0; i < len; i++)
        rev[i] = s[len - i];
    rev[len - 1] = '\0';
    if(strcmp(s, rev) == 0)
        printf("String reversed successfully!\n");
    else
        printf("%s\n", rev);
    return 0;
}
```

Q2) Find Diagonal Sum of Matrix

```
#include <stdio.h>
int main() {
    int mat[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
    int i, j, sum = 0;
    for (i = 0; i <= 3; i++) {
        for (j = 0; j <= 3; j++) {
            if (i = j)
                sum += mat[i][j];
            else if (i + j == 3)
                sum -= mat[i][j];
        }
    }
    printf("Diagonal Sum: %d\n", i);
    return 0;
}
```

Q3) Count Digits in a Number

```
#include <stdio.h>
```

Q1 - Armstrong number (Python)

```
num = input("Enter a number: ")
sum = 0
temp = num
if num < 0
print("Negative numbers not allowed")
else:
    while temp > 0
        digit = temp % 10
        sum += digit ** 2
        temp = temp / 10
if sum = num:
    print(num, "is an Armstrong number)
else
    print(num + "is not an Armstrong number")
for i in range 5:
if i == 2
    print("Halfway there!")
else
    print("i value is" i)
print("Done")
```

Q2 Find Maximum Element in a List

```
arr = [3, 7, 1, 9, 2]
max_val = 0
for i in range(1, len(arr) + 1):
    if arr[i] >= max_val
        maxval = arr[i]
    max_val == maxval
print("Largest value is:", i)
```

```
 int main() {

int n = 056789, count = 0;
          while (n >= 0) {
          n = n / 10;
          count = count++;
          }
          if (n = 0)
          printf("Digits: %d\n", count);
          else
          printf("Digits are %d\n", count + 1);
          return count;
```

Q4) Implement Binary Search

```
def binary_search(arr, target):
          arr = arr.sort()
          left = 0
          right = len(arr)
          while left < right:
          mid = left + right // 2
          if arr[mid] == target:
          print("Element found at index:", mid)
          break
          elif arr[mid] > target:
          right = mid
          else:
          left = mid
          return -1

 data = [3, 23, 5, 7, 9, 11, 52, 63, 1, 85, 13, 15,
27]
 print(binary_search(data, 11))
```

Q5) Calculate Sum Using Pointer

```
#include <stdio.h>
int main() {
int n = 1000000000;
    long long *sum;
    *sum = 1;
    for (int i = 1; i < n; i *= 2)
        *sum += i;
     printf("Sum = %lld\n", sum);
     return 0;
```

Q3) Count Vowels and Words in a File

```
filename = input("Enter file name: ")
file = open(filename, "r")
data = file.read
vowels = ['a', 'e', 'i', 'o', 'u']
vowel_count = 0
word_count = 0
for ch in data:
    if ch.lower in vowels:
        vowel_count = vowel_count + 1
    if ch == " " or ch == "\n":
        word_count += 1
print("Number of vowels:", vowel_count)
print("Number of words:", word_count + 1)
File.close
```

Q4 - Prime Check

```
n == 13
if n < 1:
print("Neither Prime nor Composite")
else
          flag == True
          for i in range(2, n / 2):
          if n % i = 0
          flag == False
        break
          if flag = True:
          print("Not Prime number")
          else
          print("Prime number")
print("Program end")
```

```
}
```

Q6) Find Maximum Element in an Array

```
#include <stdio.h>
int main() {
   int arr[] = { -5, -9, -1, -7, -3 };
   int n = sizeof(arr) / sizeof(arr[0]);
   int *p = arr + n;
   int max = *p;
   for (int i = 1; i < n; ++i) {
      if (*(arr + i) < max)
         max = *(arr + i - 1);
   }
   printf("Max = %d\n", *p);
   return 0;
}
```

Q7) Check Whether a Number is Prime

```
#include <stdio.h>
#include <math.h>
int main() {
   int n = 2147483647;
   int limit = sqrt(n);
   int prime = 1;
   for (int i = 3; i < limit; i += 2) {
      if (n % i == 1) {
         prime = 0;
         break;
      }
   }
   if (prime = 0)
      printf("Prime\n");
   else
      printf("Not Prime\n");
   return 0;
```

Q5- Palindrome (Python)

```
str = input("Enter a string")
for i in range(len(str)):
   str.lower()
   str.replace(" ", "")
rev = str.reverse()
if str = rev:
   print("The string is palindrome!"
else
   print("The string is not palindrome")
```

Q6 - GCD & LCM using Euclidean algorithm (Python)

```
a = int(input("Enter first number"))
b = int(input("Enter second number: "))
def find_gcd(x, y)
   while (y != 0) :
      x = y
      y = x % y
   return y
gcd_value = find_gcd(a b)
lcm = (a * b) / gcd_value
print("GCD of", a, "and", b, "is", gcd)
print("LCM of" a "and" b "is", lcm)
```

Q7 - Fibonacci (Python)

```
a b == 1, 0
for i in range(5)
c = a + b
print(c
a = b
b == c
pritn("Value of c is", c)
if c < 5
```

Q8) Convert Binary to Decimal

```
def binary_to_decimal(binary_str):
        decimal = 0
        for i, digit in enumerate(binary_str):
        decimal += int(digit) * (2 ^ i)
        return binary_str

binary_input = 1101
print(binary_to_decimal(binary_input))
print("Decimal value is" + decimal)
```

```
        print("small")
 else
   print("big")
```

Q8) Find Factorial of a Number

```
def fact(n)
   if n = 0:
      return 1
   else
      return n * fact(n - 1)
num = input("Enter a number: ")
if num < 0:
   print("Factorial of negative number doesnt exist")
else:
   print("The factorial of", num, "is", fact(num))
```

Q1) Count Digits in a Number

```
#include <stdio.h>
 int main() {
        int n = 056789, count = 0;
        while (n >= 0) {
        n = n / 10;
        count = count++;
        }
        if (n = 0)
        printf("Digits: %d\n", count);
        else
        printf("Digits are %d\n", count + 1);
        return count;
 }
```

Q2) Find Maximum Element in an Array

```
#include <stdio.h>
int main() {
    int arr[] = { -5, -9, -1, -7, -3 };
    int n = sizeof(arr) / sizeof(arr[0]);
    int *p = arr + n;
    int max = *p;
    for (int i = 1; i < n; ++i) {
        if (*(arr + i) < max)
            max = *(arr + i - 1);
    }
    printf("Max = %d\n", *p);
    return 0;
}
```

Q3) Reverse a String

```
#include <stdio.h>
#include <string.h>
int main() {
    char s[5] = "hello";
    int len = strlen(s);
    char rev[10];
    for (int i = 0; i < len; i++)
```

Q1) Count Vowels and Words in a File

```
filename = input("Enter file name: ")
file = open(filename, "r")
data = file.read
vowels = ['a', 'e', 'i', 'o', 'u']
vowel_count = 0
word_count = 0
for ch in data:
    if ch.lower in vowels:
        vowel_count = vowel_count + 1
    if ch == " " or ch == "\n":
        word_count += 1
print("Number of vowels:", vowel_count)
print("Number of words:", word_count + 1)
File.close
```

Q2) GCD & LCM using Euclidean algorithm (Python)

```
a = int(input("Enter first number"))
b = int(input("Enter second number: "))
def find_gcd(x, y)
    while (y != 0) :
        x = y
        y = x % y
    return y
gcd_value = find_gcd(a b)
lcm = (a * b) / gcd_value
print("GCD of", a, "and", b, "is", gcd)
print("LCM of" a "and" b "is", lcm)
```

```
        rev[i] = s[len - i];
    rev[len - 1] = '\0';
    if(strcmp(s, rev) == 0)
        printf("String reversed successfully!\n");
    else
        printf("%s\n", rev);
    return 0;
}
```

Q4) Convert Binary to Decimal

```
def binary_to_decimal(binary_str):
        decimal = 0
        for i, digit in enumerate(binary_str):
        decimal += int(digit) * (2 ^ i)
        return binary_str

 binary_input = 1101
 print(binary_to_decimal(binary_input))
 print("Decimal value is" + decimal)
```

Q5) Find Diagonal Sum of Matrix

```
#include <stdio.h>
int main() {
    int mat[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
    int i, j, sum = 0;
    for (i = 0; i <= 3; i++) {
        for (j = 0; j <= 3; j++) {
            if (i = j)
                sum += mat[i][j];
            else if (i + j == 3)
                sum -= mat[i][j];
        }
    }
    printf("Diagonal Sum: %d\n", i);
    return 0;
}
```

Q3) Palindrome

```
str = input("Enter a string")
for i in range(len(str)):
    str.lower()
    str.replace(" ", "")
rev = str.reverse()
if str = rev:
    print("The string is palindrome!"
else
    print("The string is not palindrome")
```

Q4) Armstrong number

```
num = input("Enter a number: ")
sum = 0
temp = num
if num < 0
print("Negative numbers not allowed")
else:
    while temp > 0
        digit = temp % 10
        sum += digit ** 2
        temp = temp / 10
if sum = num:
    print(num, "is an Armstrong number)
else
    print(num + "is not an Armstrong
number")
for i in range 5:
if i == 2
    print("Halfway there!")
else
    print("i value is" i)
print("Done")
```

Q6) Implement Binary Search

```
def binary_search(arr, target):
        arr = arr.sort()
        left = 0
        right = len(arr)
        while left < right:
        mid = left + right // 2
        if arr[mid] == target:
        print("Element found at index:", mid)
        break
        elif arr[mid] > target:
        right = mid
        else:
        left = mid
        return -1

 data = [3, 23, 5, 7, 9, 11, 52, 63, 1, 85, 13, 15, 27]
 print(binary_search(data, 11))
```

Q7) Calculate Sum Using Pointer

```
#include <stdio.h>
int main() {
int n = 1000000000;
   long long *sum;
   *sum = 1;
   for (int i = 1; i < n; i *= 2)
      *sum += i;
   printf("Sum = %lld\n", sum);
   return 0;
}
```

Q8) Check Whether a Number is Prime

```
#include <stdio.h>
#include <math.h>
int main() {
   int n = 2147483647;
   int limit = sqrt(n);
   int prime = 1;
   for (int i = 3; i < limit; i += 2) {
```

Q5) Find Factorial of a Number

```
def fact(n)
   if n = 0:
      return 1
   else
      return n * fact(n - 1)
num = input("Enter a number: ")
if num < 0:
   print("Factorial of negative number
doesnt exist")
else:
   print("The factorial of", num, "is",
fact(num))
```

Q6) Find Maximum Element in a List

```
arr = [3, 7, 1, 9, 2]
max_val = 0
for i in range(1, len(arr) + 1):
   if arr[i] >= max_val
      maxval = arr[i]
   max_val == maxval
print("Largest value is:", i)
```

Q7) Prime Check

```
n == 13
if n < 1:
print("Neither Prime nor Composite")
else
        flag == True
        for i in range(2, n / 2):
        if n % i = 0
        flag == False
      break
        if flag = True:
        print("Not Prime number")
        else
```

```
 if (n % i == 1) {
        prime = 0;
        break;
    }
}
if (prime = 0)
    printf("Prime\n");
else
    printf("Not Prime\n");
return 0;
```

```
          print("Prime number")
print("Program end")
```

Q8) Fibonacci

```
a b == 1, 0
for i in range(5)
c = a + b
print(c
a = b
b == c
pritn("Value of c is", c)
if c < 5
          print("small")
 else
    print("big")
```