**WEEK 1**

**1. Find whether given number is Even or Odd**

**Pseudocode –**

Enter a number

If number is divisible by 2, print even

Else print odd

**Algorithm –**

1. Start

2. Input a number ‘A’.

3. if A%2 == 0, print even.

4. else print odd.

5. end

**Flowchart –**

Input number ‘A’

A%2 == 0

**False**

**True**

Print ‘Even’

Print ‘Odd’

**2. Find sum of 3 numbers**

**Pseudocode –**

Enter 3 numbers a, b, c

Add three numbers together

Print sum

**3. Find which number from the given 2 is larger than the other**

**Pseudocode –**

Enter 2 numbers a, b

If a>b, print a is greatest

Else if a=b, print numbers are same

Else print b is greatest

**4. Swapping two numbers without an extra variable**

**Pseudocode –**

Enter 2 numbers a, b

Assign a + b to a

Assign a – b to b

Assign a – b to a

Print a and b which are now swapped

**5. Finding the sum of given *n* numbers (*n* is arbitrary, make sure your code works for all values of n)**

**Pseudocode –**

take input from user one by one inside a loop

subsequently add these inputs to a variable ‘sum’

after each input, ask whether to print sum or not

if yes print sum and exit loop and program, otherwise continue taking inputs

**6. Find whether a given number is prime or not**

**Pseudocode –**

Input number ‘A’

If A=1 assign dummy variable r=1 and for A=1 loop won’t be entered

If A=0 assign dummy variable r=2 and for A=1 loop won’t be entered

Using loop, check remainder of the number with 2,3,4…...(A-1).

If remainder comes zero assign dummy variable r=1 , otherwise assign r=0

Perform above step for all remainders and exit loop

If r=0 print ‘prime’

Else if r=1 print ‘composite’

Else if r=2 print ‘neither prime nor composite’

End

**Other ways –**

**FIND OUT**

**7. Finding all prime numbers from 1 to *n* (n∈ *N)***

**Pseudocode –**

Enter a natural number upto which primes are to be found, say ‘num’

Using a while loop (loop 1), start taking numbers (say l1) >= to 2 and <= to num

Using another while loop (loop 2) (inside the previous one) start taking numbers (say l2) < l1 and check l1%l2 == 0 for all possible values of l2 , if true l1 is not a prime , if false l1 is prime and print it

Repeat above step for all possible values of l1

**8. Finding the first *n* Fibonacci numbers (*n* ∈ *N*)**

**Pseudocode –**

Enter how many terms (num\_of\_terms) of Fibonacci are to be printed

Initialize ft=0, st=1, tt, term=1 (loop variable)

Print first term (ft) and second term (st)

Use while loop with condition term<=num\_of\_terms, inside loop -

tt = st + ft

assign value of st to ft

assign value of tt to st

add one to term

exit loop after execution of all possible values of term.