Assignment No. 1.

1. Count digits in a number
Given an integer N, return the number of digit in N.

→ count = 0;
while (n > 0) {
     n = n / 10;
     count ++;
}
s.o.p(count);

e.g. N = 12345   (Dry run)

① count = 0
    while (12345 > 0) → n = 12345/10 = 1234
                  count = 1

② while (1234 > 0) → n = 1234/10 = 123
                  count = 2

③ while (123 > 0) → n = 123/10 = 12
                  count = 3

④ while (12 > 0) → n = 12/10 = 1
                  count = 4

⑤ while (1 > 0) → n = 0/10 = 0
                  count = 5

⑥ while (0 > 0) → false condition

Time complexity = $O(\log n)$ because, we are dividing n by 10.
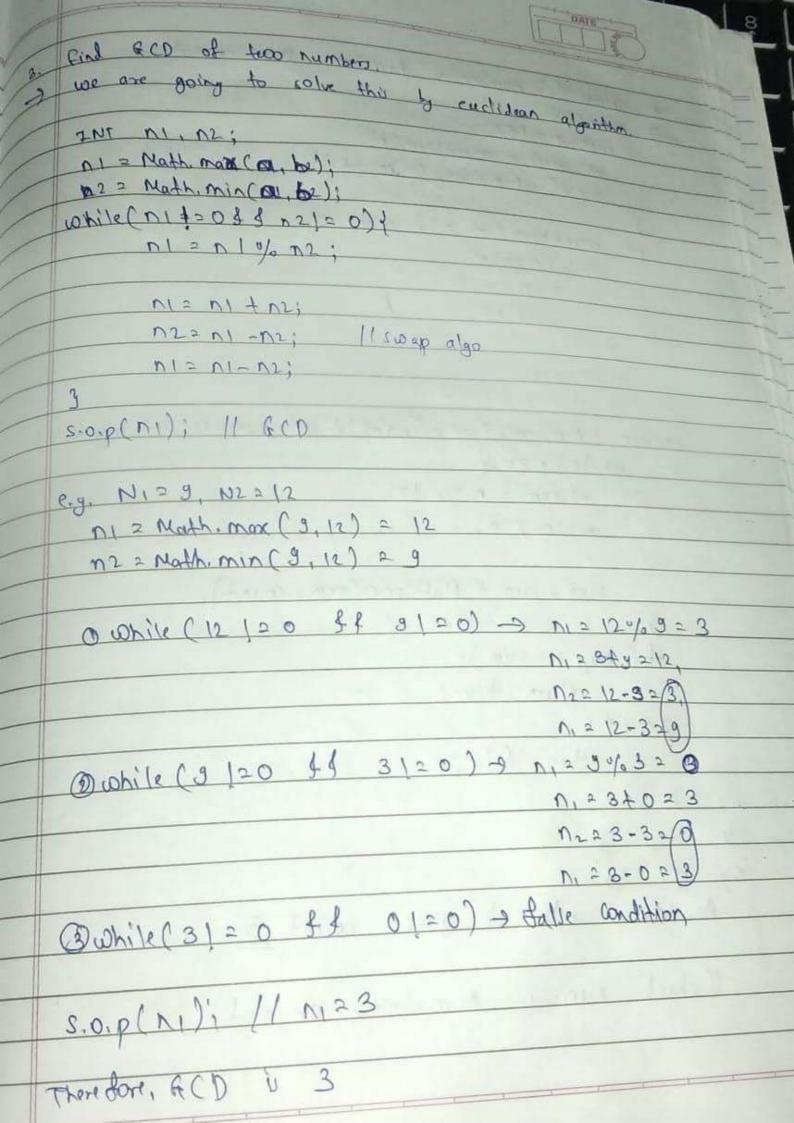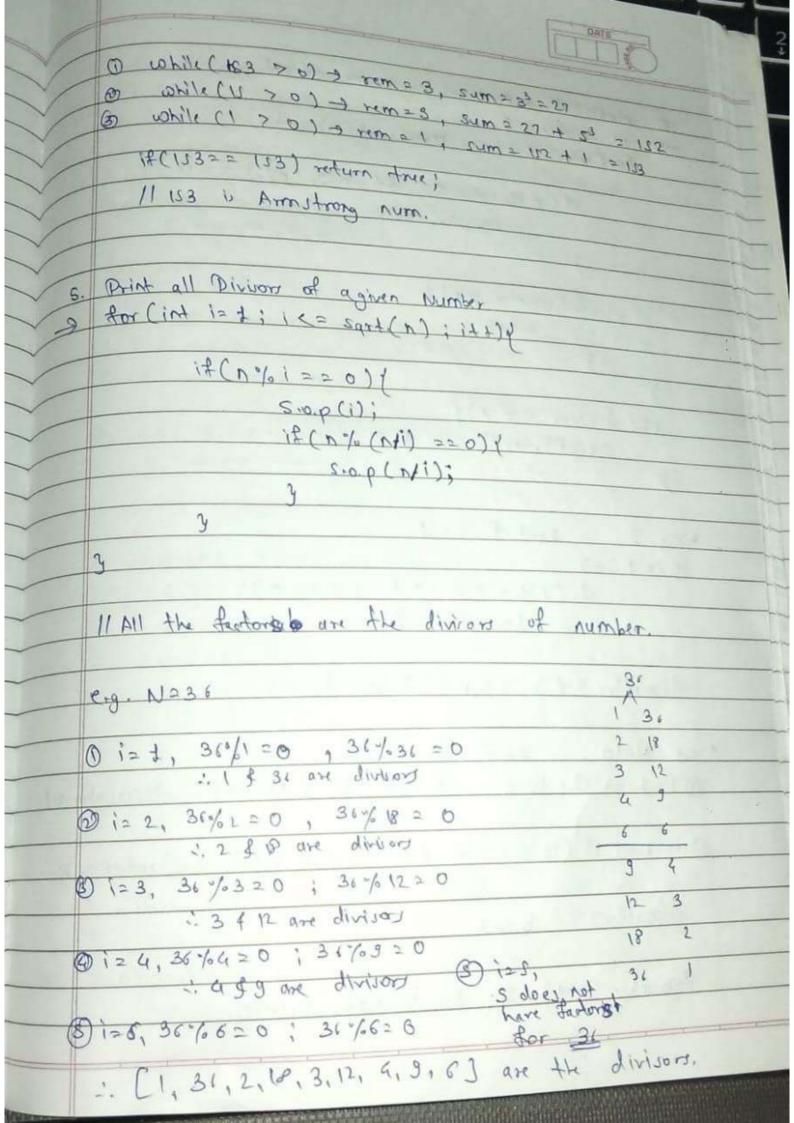
2. Check if a number is Palindrome or Not
→ int org = n; reverseNum = 0;
while (n > 0) {
    int rem = n % 10;
    n = n / 10;

    reverseNum = reverseNum * 10 + rem;
}
→ if (org == reverseNum) {
    S.o.p ("Palindrome");
}

e.g. N = 4554
① while (4554 > 0) → rem = 4, n = 455
                      reverseNum = 4
② while (455 > 0) → rem = 5, n = 45
                    reverseNum = 40 + 5 = 45
③ while (45 > 0) → rem = 5, n = 4
                   reverseNum = 450 + 5 = 455
④ while (4 > 0) → rem = 4, n = 0
                  reverseNum = 4550 + 4 = 4554
⑤ while (0 > 0) → false condition

if (4554 == 4554) true;
Therefore, 4554 is palindrome

Time complexity = $O(\log n)$

3. Find GCD of two numbers.

→ we are going to solve this by euclidean algorithm.

```
INT n1, n2;
n1 = Math.max(a, b);
n2 = Math.min(a, b);
while(n1 != 0 && n2 != 0){
    n1 = n1 % n2;

    n1 = n1 + n2;
    n2 = n1 - n2;      // swap algo
    n1 = n1 - n2;
}
s.o.p(n1);   // GCD
```

e.g. $N_1 = 9$, $N_2 = 12$

$n1 = Math.max(9, 12) = 12$

$n2 = Math.min(9, 12) = 9$

① while (12 != 0 && 9 != 0) → $n1 = 12 \% 9 = 3$

$n_1 = 3 + 9 = 12$,

$n_2 = 12 - 9 = \boxed{3}$,

$n_1 = 12 - 3 = \boxed{9}$

② while (9 != 0 && 3 != 0) → $n_1 = 9 \% 3 = \boxed{3}$

$n_1 = 3 + 0 = 3$

$n_2 = 3 - 3 = \boxed{0}$

$n_1 = 3 - 0 = \boxed{3}$

③ while(3 != 0 && 0 != 0) → false condition

s.o.p(n_1); // $n_1 = 3$

Therefore, GCD is 3

4. Check if a number is Armstrong Num or not.

→ main() {
    int n = 153;
    int count = countNum(n);
    S.o.p (is Armstrong (n, count));
}

int countNum (int n) { count = 0;
    while (n > 0) {
        n = n / 10;
        count ++;
    }
    return count;
}

boolean is Armstrong (int n, int count) { int sum = 0; int org = n;
    while (n > 0) {
        int rem = n % 10;
        n = n / 10;

        sum = sum + Math.pow (rem, count);
    }
    if (org == sum) {
        return true;
    } else {
        return false;
    }
}

e.g. N = 153
    As, previous code, it gives count 3.

    // check num is Armstrong or not

① while (153 > 0) → rem = 3, sum = 3³ = 27
② while (15 > 0) → rem = 5, sum = 27 + 5³ = 152
③ while (1 > 0) → rem = 1, sum = 152 + 1 = 153

if (153 == 153) return true;
// 153 is Armstrong num.

6. Print all Divisors of a given Number
→ for (int i = 1; i <= sqrt(n); i++) {

      if (n % i == 0) {
        S.o.p (i);
        if (n % (n/i) == 0) {
          S.o.p (n/i);
        }
      }

}

// All the factors are the divisors of number.

e.g. N = 36

36	
1	36
2	18
3	12
4	9
6	6
9	4
12	3
18	2
36	1

① i = 1, 36 % 1 = 0 , 36 % 36 = 0
   ∴ 1 & 36 are divisors

② i = 2, 36 % 2 = 0 , 36 % 18 = 0
   ∴ 2 & 18 are divisors

③ i = 3, 36 % 3 = 0 ; 36 % 12 = 0
   ∴ 3 & 12 are divisors

④ i = 4, 36 % 4 = 0 ; 36 % 9 = 0
   ∴ 4 & 9 are divisors

⑤ i = 5,
   5 does not
   have factors
   for 36

⑥ i = 6, 36 % 6 = 0 ; 36 % 6 = 0

∴ [1, 36, 2, 18, 3, 12, 4, 9, 6] are the divisors.

6. check if a number is prime or not

→ int counter = 0;
for(int i=1; i<= Math.sqrt(n); i++){

    if(n % i == 0 && n % (n/i) == 0){
        counter ++;
    }

    if (counter > 1){
        break;
    }
}
if(counter == 1){
    S.o.p("Prime");
}

* eg. 2   Sqrt of 2 = 1
① i=1; →
    if (2 % i = 0 && 2 % 2 = 0);
      ↳ conter = 1;

if(counter = 1) S.o.p ("Prime");

* eg. N=10,   sqrt of 10 = 3
① i=1 → if (10 % 1 = 0 && 10 % 10 = 0); → counter = 1;

② i=2 → if (10 % 2 = 0 && 10 % 5 = 0); → counter = 2;

if(counter = 2) break;

if(counter > 1) so, 10 is not prime.