

Assignment No. 2

```
1.class array{
    public static void main(String[]args){
        int[]arr={1,2,3,4};
        printArray(arr,0);
    }
    static void printArray(int[] arr,int s){
        if(s>=arr.length){
            return;
        System.out.println(arr[s]);
        printArray(arr,s+1);
        System.out.println(arr[s]);
    }
}
```

4) print elements of an array (forward & backward)

```

main() {
    int arr = [1, 2, 3, 4];
    printArray(arr, 0);
}

void printArray(int arr[], int s) {
    if (s > arr.length - 1) {
        return;
    }
    print(arr[s]);
    printArray(arr, s + 1);
    print(arr[s]); // 1
}

void printArray(int arr[], int s) {
    if (s > arr.length - 1) {
        return;
    }
    print(arr[s]);
    printArray(arr, s + 1);
    print(arr[s]); // 2
}

void printArray(int arr[], int s) {
    if (s > arr.length - 1) {
        return;
    }
    print(arr[s]);
    printArray(arr, s + 1);
    print(arr[s]); // 3
}

```

```

Void printArray (int [] arr, int s) {
    if (s > arr.length - 1) {
        return;
    }
    print (arr [s]);
    printArray (arr, s + 1);
    print (arr [s]); // 4
}

Void printArray (int [] arr, int s) {
    if (s > arr.length - 1) {
        return;
    }
    print (arr [s]);
    printArray (arr, s + 1);
    print (arr [s]); // 4
}

print (arr [s]);
printArray (arr, s + 1);
print (arr [s]); // 5

```

OLP:- 1
2
3
4
5
3
2
1

```
2.class sum{  
    public static void main(String[]args){  
        int number=1234;  
        sumOfDigits(number,0);  
    }  
    static void sumOfDigits(int number,int sum){  
        if(number==0){  
            System.out.println(sum);  
            return; }  
        int remainder=number%10;  
        sum=sum+remainder;  
        number=number/10;  
        sumOfDigits(number,sum);  
    }  
}
```

Q.1. find the sum of digits

main() {

```
    int number = 123;  
    sum(number, 0);
```

```
void sum(int number, int sum) {  
    if (number == 0) {  
        print(sum);  
        return; } }
```

```
3 ← int remainder = number % 10;  
3 = 3 ← sum = sum + remainder;  
2 ← number = number / 10;  
sum(number, sum);  
    12     3
```

```
void sum(number, int sum) {  
    if (number == 0) {  
        print(sum);  
        return; } }
```

```
2 ← int remainder = number % 10;  
+2 = 5 ← sum = sum + remainder;  
1 ← number = number / 10;  
sum(number, sum);  
    5
```

```
void sum(int number, int sum) {  
    if (number == 0) {  
        print(sum);  
        return; } }
```

```
1 ← int remainder = number % 10;  
+1 = 6 ← sum = sum + remainder;  
0 ← number = number / 10;  
sum(number, sum);  
    0     6
```

```

void sum(int number, int sum){
    if (number == 0) {
        System.out.println(sum);
        return;
    }
    int remainder = number % 10;
    sum = sum + remainder;
    number = number / 10;
    sum(number, sum);
}

```

O/P:-

6

f(0, 6)
f(1, 5)
f(12, 3)
f(123, 0)

```

3.class product{
    public static void main(String[] args){
        int number=1234;
        productOfDigits(number,1);
    }

    static void productOfDigits(int number,int
product){
        if(number==0){
            System.out.println(product);
            return;
        }
        int remainder=number%10;
        product=product*remainder;
    }
}

```

```
number=number/10;  
productOfDigits(number,product);  
}  
}
```

find the product of digits in a number.

```
main() {  
    int number; - 123;  
    → product(number, 1); ?  
    void product(int number, int product) {  
        if (number == 0) ?  
            print(product);  
            return; ?  
        3 ← int remainder = number % 10;  
        1*3=3 ← product = product * remainder; ?  
        12 ← number = number / 10;  
        product(number, product);  
  
        void product(int number, int product);  
        if (number == 0) ?  
            print(product);  
            return; ?  
        2 ← int remainder = number % 10;  
        *2=6 ← product = product * remainder; ?  
        1 ← number = number / 10;  
        product(number, product);
```

```

Void product (int number, int product)?
    if (number == 0)?
        print (product);
        return;
    6 ← int remainder = number % 10^3
    6 ← Product = product * remainder;
    0 ← number = number / 10^3;
    Product (number, product);
    ↓
    void product (int number, int product)
        if (number == 0)?
            print (product);
            return;
        0 ← int remainder = number % 10^3;
        product = product * remainder;
        number = number / 10^3;
        product (number, product);
    O/P: -
    6

```

f(0, 6)
f(1, 6)
f(12, 3)
f(123, 1)

4. class coundigits{

```

public static void main(String[] args){
    int n=12345;
    int count=0;
    f(count,n);
}

static void f(int count,int n){
    if(n==0){
        System.out.println(count);
        Return;
    }
}

```

```

n=n/10;
count++;
f(count,n);}

}

```

D) count number of digits in a number.

```

main()
{
    int count = 0;
    int number = 123;
    countDigits(number, count);
}

void countDigits(int number, int count)
{
    if (number == 0)
        System.out.print(count);
    return;
}

n = n / 10; → 123 / 10 = 12.3 → 12
Count++; → 1
countDigits(number, count);

→ void countDigits(int number, int count)
{
    if (number == 0)
        print(count);
    return;
}

n = n / 10; → 12 / 10 = 1.2 → 1
Count++; → 2
countDigits(number, count);

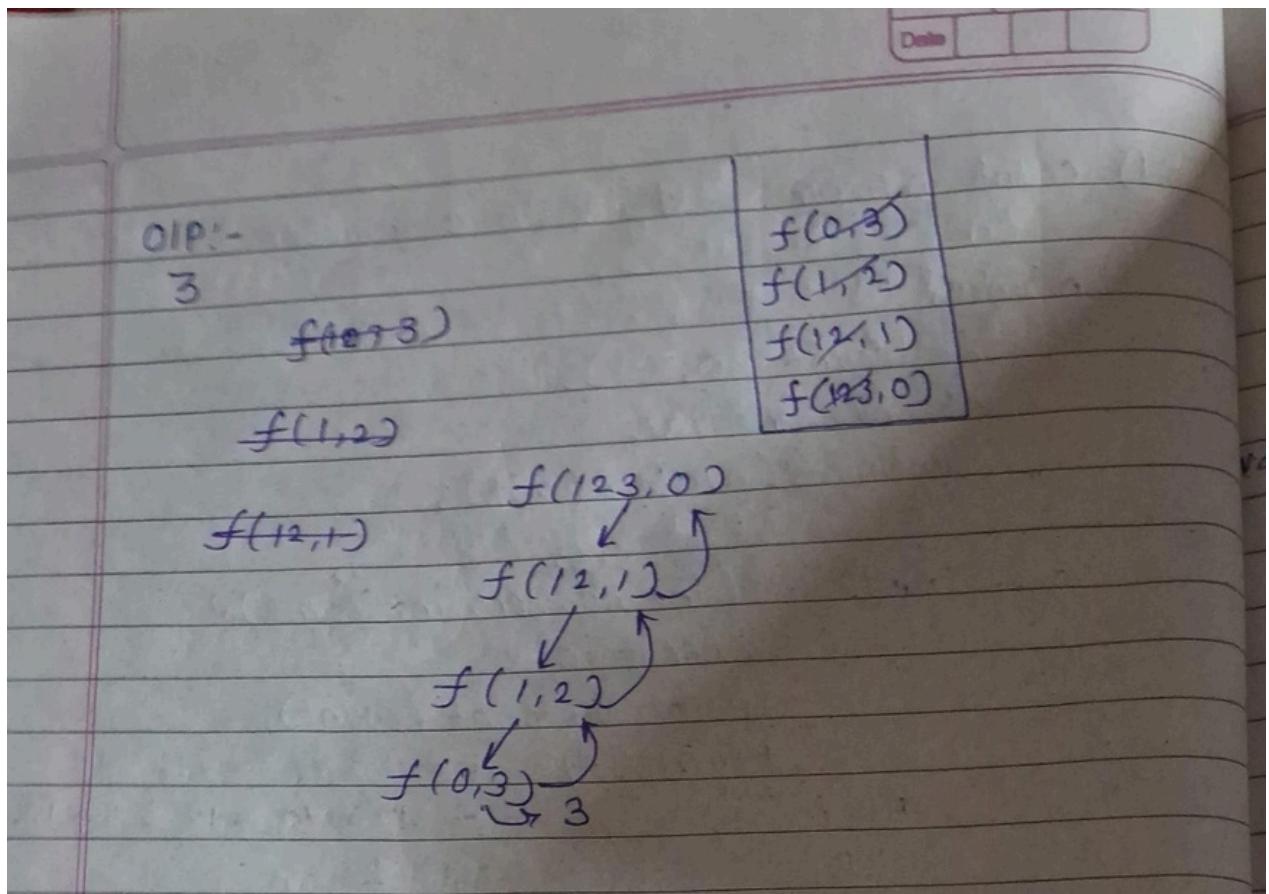
→ void countDigits(int number, int count)
{
    if (number == 0)
        print(count);
    return;
}

n = n / 10; → 1 / 10 → 0.1 → 0
Count++; → 3
countDigits(number, count); }

→ void countDigits(int number, int count)
{
    if (number == 0)
        print(count); → 3
    return;
}

n = n / 10;
Count++;
countDigits();

```



5. class array1{

```

public static void main(String[] args){
    int[] arr={1,5,3,4};
    int s=0;
    int max=arr[s];
    maxArray(arr,s,max);
}

static void maxArray(int[] arr,int s,int max){
    if(s>=arr.length){
        System.out.println(max);
        return;
    if(max <arr[s]){

```

```
    max=arr[s];}  
maxArray(arr,s+1,max);  
  
}  
}
```

5) Find maximum element of Array

~~main() {~~

~~int arr = {1, 2, 3, 4};~~

~~maxArray(arr, 0, max);~~

~~int max = arr[0];~~

~~}~~

~~main() {~~

~~int arr = {1, 2, 3, 4};~~

~~int max = arr[0];~~

~~maxArray(arr, max, 0);~~

void maxArray(int arr, int max, int s) {

if (s > arr.length - 1) {

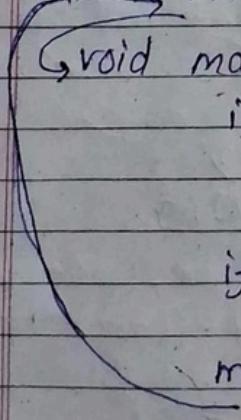
print(max);

return;

if (max < arr[s]) {

max = arr[s];

maxArray(arr, max, s + 1);



```

void maxArray (int [ ] arr, int max, int s) {
    if (s > arr.length - 1) {
        print (max);
        return;
    }
    if (max < arr[s]) {
        max = arr[s];
    }
    maxArray (arr, max, s + 1);
}

→ void maxArray (int [ ] arr, int max, int s) {
    if (s > arr.length - 1) {
        print (max);
        return;
    }
    if (max < arr[s]) {
        max = arr[s];
    }
    maxArray (arr, max, s + 1);
}

→ void maxArray (int [ ] arr, int max, int s) {
    if (s > arr.length - 1) {
        print (max);
        return;
    }
    if (max < arr[s]) {
        max = arr[s];
    }
    maxArray (arr, max, s + 1);
}

→ void maxArray (int [ ] arr, int max, int s) {
    if (s > arr.length - 1) {
        print (max);
        return;
    }
    if (max < arr[s]) {
        max = arr[s];
    }
    maxArray (arr, max, s + 1);
}

```

4 ←

O.P:-
4

6. class sortedArray{

```

public static void main(String [] args) {
    int [ ] arr = {1, 3, 3, 4};
    int s = 0;
    int max = arr[s];
    boolean check = sortedArray (arr, s, max);
}

```

```
if(check){
    System.out.println("Sorted");}
else{
    System.out.println(" not Sorted");}
}
static boolean sortedArray(int[] arr,int s,int max){
    if(s>=arr.length-1){
        return true;
    }
    if(max >=arr[s]){
        return false;}

    return sortedArray(arr,s+1,max=arr[s]);
}
}
```

Page No. _____
Date _____

6) check if array is sorted or not

```

main() {
    int arr = {1, 2, 3, 4}; // 1 2 3 4
    int max = arr[0]; // 0 1 2 3
    boolean check = isSorted(arr, 0, max);
    if (check) {
        print("sorted");
    } else {
        print("not sorted");
    }
}

boolean isSorted(int arr, int s, int max) {
    if (s > arr.length - 1) {
        return true;
    }
    if (max >= arr[s]) {
        return false;
    }
    return isSorted(arr, s + 1, max = arr[s]);
}

boolean isSorted(int arr, int s, int max) {
    if (s > arr.length - 1) {
        return true;
    }
    if (max >= arr[s]) {
        return false;
    }
    return isSorted(arr, s + 1, max = arr[s]);
}

boolean isSorted(int arr, int s, int max) {
    if (s > arr.length - 1) {
        return true;
    }
    if (max >= arr[s]) {
        return false;
    }
    return isSorted(arr, s + 1, max = arr[s]);
}

```

7.class prime{

```
public static void main(String[] args){
```

```
    int number=1;
```

```
    if(isPrime(number,2)){
```

```
        System.out.println("Number is prime");
```

```
    }

else{
    System.out.println("Number is not
prime");}
}

static Boolean isPrime(int number,int i){

if(number<=2){

    return number==2;}

if(number%i==0){

    return false;}

if(number>Math.sqrt(number)){

    return true;}

return isPrime(number,i++);}

}
```

5) check if a Number is prime or not.

```
→ main() {  
    int Number = 25;  
    boolean check = isPrime(2, Number);  
    if (isPrime check) {  
        print("Number is prime");  
    } else {  
        print("Number not prime");  
    }  
}
```

```
boolean isPrime(int i, int Number) {  
    if (Number <= 2) {  
        return number == 2; }  
    if (Number % i == 0) {  
        return false; }  
    if (Number > math.sqrt(Number)) {  
        return true; }  
    isPrime(i + 1, Number);  
}
```

```
boolean isPrime(int i, int Number) {  
    if (Number <= 2) {  
        return number == 2; }  
    if (Number % i == 0) {  
        return false; }  
    if (Number > math.sqrt(Number)) {  
        return true; }  
    isPrime(i + 1, Number);  
}
```

```
boolean isPrime(int i, int Number) {  
    if (Number <= 2) {  
        return number == 2; }  
    if (Number % i == 0) {  
        return false; }  
    if (Number > math.sqrt(Number)) {  
        return true; }  
}
```

Page No. _____
Date _____

```

    isprime(i+1, number); }

    boolean isprime(int i, int number) {
        if (Number <= 2) {
            return Number == 2; }
        if (Number % i != 0) {
            return false; }
        if (Number > math.sqrt(Number)) {
            return true; }
        isprime(i+1, Number); }
    
```

O/P:-
Number Not palindrome

```

isprime(2, 25)
    |
    +--> isprime(3, 25)
        |
        +--> isprime(4, 25)
            |
            +--> isprime(5, 25)
    
```

isprime(5, 25)
isprime(4, 25)
isprime(3, 25)
isprime(2, 25)

8. class FirstOccurrence{

```
public static void main(String[] args){
```

```
    int[] arr = {1, 2, 2, 4};
```

```
    int number = 2;
```

```
    FirstOccurrence(arr, 0, number);
```

```
}
```

```
static void FirstOccurrence(int[] arr, int s, int
number){
```

```
if(s>=arr.length-1){
    System.out.println("Element not
found");
    return;
}

if(number==arr[s]){
    System.out.println(s);
    return ;}
FirstOccurence(arr,s+1,number);
}

}
```

7) find first occurrence of element in array

```

main() {
    int arr = {1, 2, 3, 4, 2};
    int check = 2;
    firstOccurrence(arr, 0, check);
}

void firstOccurrence(int arr[], int s, int check) {
    if (s > arr.length - 1) {
        System.out.println("element not found");
        return;
    }
    if (arr[s] == check) {
        System.out.print(s);
        return;
    }
    firstOccurrence(arr, s + 1, check);
}

void firstOccurrence(int arr[], int s, int check) {
    if (s > arr.length - 1) {
        System.out.println("element not found");
        return;
    }
    if (arr[s] == check) {
        System.out.print(s);
        return;
    }
    firstOccurrence(arr, s + 1, check);
}

```

O/P:-

9. class LastOccurrence{

```

public static void main(String[] args) {
    int[] arr = {1, 5, 8, 4};
    int number = 2;
    int index = 0;
    FirstOccurrence(arr, 0, number, index);
}

```

```

static void FirstOccurrence(int[] arr,int s,int
number,int index){

    if(s>arr.length-1){
        if(index==0 && number==arr[0]){
            System.out.println(index);
            return;
        } if(index==0&&number!=arr[0]){
            System.out.println("not found");
        }else{
            System.out.println(index);
            return;
        } }
        if(number==arr[s]){
            index=s;
            ;
        FirstOccurrence(arr,s+1,number,index);
    }
}

```

8) find last occurrence of element in array

```

main() {
    int arr = [1,2,2];
    int check = 2;
    int index = 0;
    firstLastOccurrence(firstLastOccurrence(arr, check, index, 0));
}

void lastOccurrence(int arr, int check, int index, int s) {
}
}

```

```

Page No. _____
Date. _____
if (s > arr.length - 1) {
    if (index == -1 || check == arr[0]) {
        if (index == -1) {
            print("Not found");
        } else {
            print(index);
        }
        return;
    }
}

if (check == arr[s]) {
    index = s;
    lastOccurrence(arr, index, check, s + 1);
}

void lastOccurrence(int arr[], int check, int index, int s) {
    if (s > arr.length - 1) {
        if (index == -1 || check == arr[0]) {
            if (index == -1) {
                print("Not found");
            } else {
                print(index);
            }
            return;
        }
    }

    if (check == arr[s]) {
        index = s;
        lastOccurrence(arr, check, index, s + 1);
    }
}

void lastOccurrence (int arr[], int check, int index, int s) {
    if (s > arr.length - 1) {
        if (index == -1 || check == arr[0]) {
            if (index == -1) {
                print("Not found");
            } else {
                print(index);
            }
            return;
        }
    }

    if (check == arr[s]) {
        index = s;
        lastOccurrence (arr, check, index, s + 1);
    }
}

void lastOccurrence (int arr[], int check) {
    if (s > arr.length - 1) {
        if (index == -1 || check == arr[0]) {
            if (index == -1) {
                print("Not found");
            } else {
                print(index);
            }
            return;
        }
    }

    if (check == arr[s]) {
        index = s;
        lastOccurrence (arr, check, index, s + 1);
    }
}

```

The handwritten code is a C program for a binary search. It includes a main function with a check for index 0, a recursive search function, and a helper function for character comparison.

```
if(index==0) if(index == 0) { if(check == arr[0]) { print("NOT found"); return; } else { print(index); return; } }

if(check == arr[0]) {
    index = s;
    lastOccurrence(arr, check, index, s);
}

int lastOccurrence(char arr[], char check, int index, int s) {
    if(index == 0) {
        if(arr[0] == check) {
            s = 0;
            return 0;
        }
        return -1;
    }
    if(arr[index] == check) {
        s = index;
        return lastOccurrence(arr, check, index-1, s);
    }
    return lastOccurrence(arr, check, index-1, s);
}
```

10. class reverse{

```
public static void main(String[] args){
```

```
    int number=12345;
```

```
    int reversedNumber=0;
```

```
reverseTheNumber(number,reversedNumber);
```

```
}
```

```
public static void reverseTheNumber(int  
number,int reversedNumber){
```

```
    if(number==0){
```

```
        System.out.println(reversedNumber);
```

```
        return;}
```

```
    int remainder=number%10;
```

```
reversedNumber=reversedNumber*10+remainde
r;

number=number/10;

reverseTheNumber(number,reversedNumber);

}

}
```

```
11. class OccurrenceinNumber{

public static void main(String[]args){

int number=14444;

int check=2;

int count=0;

OccurrenceinNumber(number,check,count);

}

static void OccurrenceinNumber(int
number,int check,int count){

if(number==0){

if(count==0){

System.out.println(count);

return;}

else{

System.out.println(count);
```

```
    return;  
}  
}  
  
int remainder=number%10;  
if(check==remainder){  
    count++;  
}  
number=number/10;
```

OccurrenceinNumber(number,check,count);

```
}  
}
```

9) find how many times digit occurs in number

main () {

 int number = ~~12345~~; 118
 int check = 1;
 int count = 0;

 occurrence(number, check, count); } }

void occurrence (int number, int check, int count) {

 if (number == 0) {

 if (count == 0) {

 simply the print count else {

 without

 if - else

 print (count); } return;

 print (count); } }

 113

 int remainder = number % 10; → 3

 if (check == remainder) {

 number = number / 10;

 occurrence (number, check, count); } }

10)

→ void occurrence (int number, int check, int count) {

 if (number == 0) {

 print (count); } }

 return; }

 int remainder = number % 10; → 1

 if (check == remainder) {

 count++; } → 1

Page No. _____
Date. _____

```

number = number / 10; → 1
occurrence (number, check, count);

→ void occurrence (int number, int check, count++) {
    if (number == 0) {
        print (count); → 1
        return; → 2
    }
    int remainder = number % 10; → 1
    if (check == remainder) {
        count++; → 2
    }
    number = number / 10; → 0
    occurrence (number, check, count); → 3
}

→ void occurrence (int number, int check, int count) {
    if (number == 0) {
        print (count); → 2
        return; → 3
    }
    int remainder = number % 10;
    if (check == remainder) {
        count++; → 3
    }
    number = number / 10;
    occurrence (number, check, count); → 4
}

```

12. class palindrome{

```
public static void main(String[] args){
```

```
    int n=11;
```

```
    int reversedNumber=0;
```

```
    int original=n;
```

```
    isPalindrome(reversedNumber,n,original);
```

```
}
```

```
static void isPalindrome(int reversedNumber,int
n,int original){  

    if(n==0){  

        if(original== reversedNumber){  

            System.out.println("Number is  

Palindrome");}  

        else{  

            System.out.println("Number is not  

Palindrome");}  

        return;  

    }  

    int remainder=n%10;  

    n=n/10;  

    reversedNumber=reversedNumber*10+remainde  

r;  

    isPalindrome(reversedNumber,n,original);}  

}
```

2) check if a number is palindrome or not

main() {

int n = 121;

int reversedNumber = 0;

int original = n;

void ispalindrome(int n, int reversedNumber, int original) {

if (n == 0) {

if (original == reversedNumber) {

printf("Number is palindrome"); }

else {

printf("Number is Not palindrome");

return; }

int remainder = n % 10;

reversedNumber = reversedNumber + 10 + remainder;

n = n / 10;

ispalindrome(n, reversedNumber, original); }

void ispalindrome(int n, int reversedNumber, int original) {

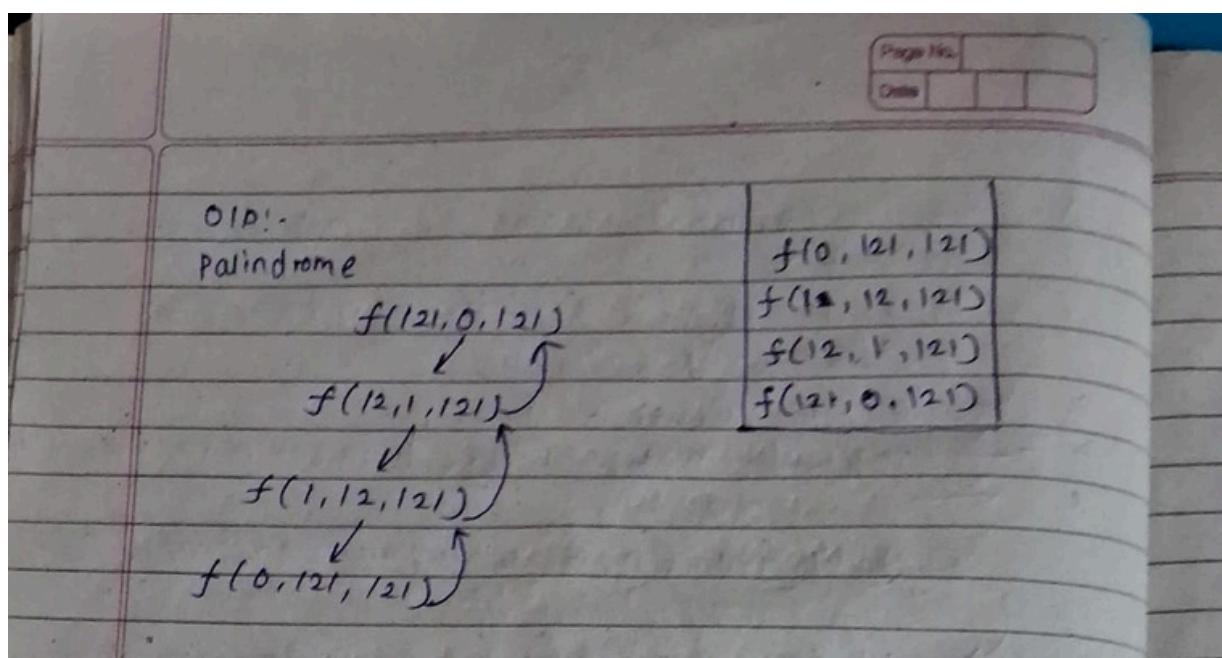
if (original == 0) {

if (original == n) {

```

    print("Number is palindrome"); }
else {
    print("Not palindrome");
    return;
}
2 ← int remainder = n%10;
121 ← reversedNumber = reversedNumber * 10 + remainder;
1 ← n = n/10;
isPalindrome(n, reversedNumber, original);
    1   12   121
→ void isPalindrome(int n, int reversedNumber,
    int original) {
if(n==0) {
    if(reversedNumber==original) {
        print("palindrome");
    } else {
        print("Not palindrome");
    }
    return;
}
1 ← int remainder = n%10;
121 ← reversedNumber = reversedNumber * 10 + remainder;
0 ← n = n/10;
isPalindrome(n, reversedNumber, original);
    0   121   121
→ void isPalindrome(int n, int reversedNumber,
    int original) {
if(n==0) {
    if(reversedNumber==original) {
        print("palindrome");
    } else {
        print("Not palindrome");
    }
    return;
}
int remainder = n%10;
reversedNumber = reversedNumber * 10 + remainder;
n = n/10;
isPalindrome(n, reversedNumber, original);
    121   121

```



```

13. class gcd{
    public static void main(String[]args){
        int n1=12; int n2=36;
        int gcd=1;
        gcdTwo(n1,n1,1,gcd);

    }

    public static void gcdTwo(int n1,int n2,int i,int gcd){
        if(i>n1||i>n2){
            System.out.println(gcd);
            return;}
        if(n1%i==0 && n2%i==0){
            gcd=i;}
        gcdTwo(n1,n2,i+1,gcd);

    }

```

⑦ Find GCD of two Numbers

main() {

 int $n_1 = 15$
 int $n_2 = 15$; // 15
 int gcd = 1;
 gcd($n_1, n_2, gcd, 1$);

 void gcd(int n_1, n_2, gcd, i) {
 if ($i > n_1 \text{ || } i > n_2$) {
 print(gcd);
 return; // Y
 }
 }
}

if ($n_1.y.i == 0$ && $n_2.y.i == 0$)
 gcd = i; y
 gcd($n_1, n_2, gcd, i+1$); y

2 void gcd(int n_1 , int n_2 , int gcd, int i);
 if ($i > n_1$ || $i > n_2$)
 print(gcd);
 return; y
 if ($n_1.y.i == 0$ && $n_2.y.i == 0$)
 gcd = i; y
 gcd($n_1, n_2, gcd, i+1$); y

3 void gcd(int n_1 , int n_2 , int gcd, int i);
 if ($i > n_1$ || $i > n_2$)
 print(gcd);
 return; y
 if ($n_1.y.i == 0$ && $n_2.y.i == 0$)
 gcd = i; y
 gcd($n_1, n_2, gcd, i+1$); y

4 void gcd(int n_1 , int n_2 , int gcd, int i);
 if ($i > n_1$ || $i > n_2$)
 print(gcd);
 return; y
 if ($n_1.y.i == 0$ && $n_2.y.i == 0$)
 gcd = i; y
 gcd($n_1, n_2, gcd, i+1$); y

5 void gcd(int n_1 , int n_2 , int gcd, int i);
 if ($i > n_1$ || $i > n_2$)
 print(gcd);
 return; y
 if ($n_1.y.i == 0$ && $n_2.y.i == 0$)
 gcd = i; y

```

14. class printDivisible{
    public static void main(String[]args){
        int number=12;
        printAllDivisible(number,1);
    }
    static void printAllDivisible(int number,int i){
        if(i>number){
            return;
        if(i%3==0){
            System.out.println(i);
        }
        printAllDivisible(number,i+1);
    }
}

```

3) print numbers divisible by 3 from 1 to N.

→ main() {

 int number = ~~12~~⁶;

 divisible(number, 1);

 y ↗

 void divisible(int number, int i) {

 if (i > number) {

 return; y

 if (i % 3 == 0) {

 print(i);

 y ↗

 divisible(number, i + 1);

 }

Page No. _____
Date _____

```

void divisible(int number, int i) {
    if (i > number) {
        return;
    }
    if (i % 3 == 0) {
        cout << i;
        divisible(number, i + 1);
    }
}

void divisible(int number, int i) {
    if (i > number) {
        return;
    }
    if (i % 3 == 0) {
        cout << i;
        divisible(number, i + 1);
    }
}

void divisible(int number, int i) {
    if (i > number) {
        return;
    }
    if (i % 3 == 0) {
        cout << i;
        divisible(number, i + 1);
    }
}

void divisible(int number, int i) {
    if (i > number) {
        return;
    }
    if (i % 3 == 0) {
        cout << i;
        divisible(number, i + 1);
    }
}

void divisible(int number, int i) {
    if (i > number) {
        return;
    }
    if (i % 3 == 0) {
        cout << i;
        divisible(number, i + 1);
    }
}

```

```
void divisible(int number, int i) {
    if (i > number) {
        return;
    }
    if (i * 3 == 0) {
        print(i);
        divisible(number, i + 1);
    }
}

O/P:-
3
6
```

15. public class PowerCalculator {

```
public static int power(int base, int exponent) {
    if (exponent == 0) {
        return 1;
    } else {
        return base * power(base, exponent - 1);
    }
}
```

```
public static void main(String[] args) {
    int base = 2;
    int exponent = 3;
```

```

int result = power(base, exponent);
System.out.println(base + "^" + exponent +
"= " + result);
}
}

```

10) power of number

class powercalculator

main() {

 int base = 2;

 int exponent = 3;

 int result = power(base, exponent);

 print(result); } //

int power(int base, int exponent) {

 if(exponent == 0) {

 return 1; }

 else {

 return base * power(base, exponent - 1); }

 // 2 * 4

```
int power(int base, int exponent) {
    if (exponent == 0)
        return 1;
    else
        return base * power(base, exponent - 1);
}

int power(int base, int exponent) {
    if (exponent == 0)
        return 1;
    else
        return base * power(base, exponent - 1);
}

int power(int base, int exponent) {
    if (exponent == 0)
        return 1;
    else
        return base * power(base, exponent - 1);
```