output : 1 2 3 4 5

1. print elements of an array (forward & backwar

Given statement :-

Given an array print all elements using recursion in both forward and reverse ord

example 1 :

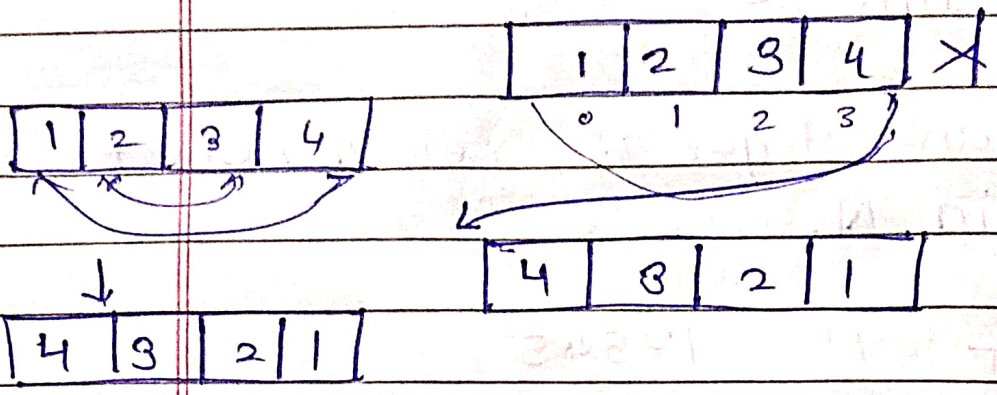
Input : arr = [1, 2, 3, 4]

output (forward) : 1 2 3 4

output (Backward) : 4 3 2 1

Date : / /
Page No.:

for output (Backword) :

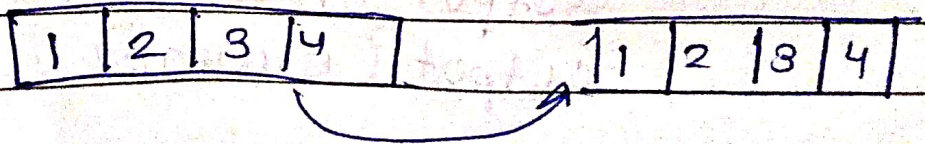



```
Int[] arr = {1,2,3,4,};
int[] newArr = new in[4];

for(int i=0 ; i<4 ; i++){
    newArr[i] = arr[arr.length-1-i]
}
```

for output (Forward) :

```
Int[] arr = {1,2,3,4,};
int [] new arr = newin [4]

for [int i=0 ; i<4 ; ]{
    new Arr[i] = arr [arr.length
```

```
Int start = 0
int end = str.length()-1
while (start<end){
    If(str.char(start) != str.charAt(end)
```



Date : / /
Page No.:

2] find sum of digit of number :

PS → Given an integer N, find sum of its digits using recursion.

Ex.1. Input N = 1234

output : 10

→

```
main () {
Int sum = F(n);
  print (sum);
}
while (n>0);
  a = n % 10
  print a;
    count ++ ;
}
  sop(()); print (sum);
}
```

$\frac{1234}{10} = 123$

$\frac{123}{10} = 12$

$\frac{12}{10} = 1$

$\frac{1}{10} = 0.$

① + ② + ③ + ④.

3] Find product of digit of number.

PS →

Given an integer N, find product of its digit using recursion..

Date: / /
Page No.:

Ex 3. Input N = 291
output = 6

→

~~int factor = f(n);~~
public class Digit product {
int N = 291;
If n < 10 {
return n;
}
return (n%10) * product of
digit (n/10);
}
S.o.p.ln ("product of digits :" + result);
}
}

= 291 !=
= 29 3
= 2 2
= 1 × 3 × 2
= 6

output

Q 4. count number of digits in number

PS: Given integer N count how many digit it contain using recursion.

Ex.1. N = 98765 output 5.

→

```
main() {
    f();
}
count = 0;
void f() {
    if (count == 5) {
        return;
    }
    printf(count);
    count++;
    f();
}
```

0 =

```
main() {
    f();
}
count = 0;
void f() {
    if (count == 5) {
        return;
    }
    printf(count);
    count++;
    f();
}
```

1 =

Page No.:

```
main ()
{
    f();
}

count = 0;
void f();
if (count == 5)
    return;

printf (count + 1);
count ++;
f();
}
```

```
main ()
{
    f();
}

count = 0;
void f();
if (count == 5)
    return;

printf (count);
count ++;
f();
}
```

```
main ()
{
    f();
}
```

Date : / /
Page No.:

count = 0;
void f();
if (count == 5){
Return;
}
print (count);
count ++;
f();
}

output : 5

Q. 5. find maximum element in array.

Ans : Given an array of integers, find maximum element using recursion.

Ex. 1. INPUT arr = [2, 5, 9, 1, 6]
output : 9

2, 5, 9, 1, 6

Int[] arr (~~1, 2, 3, 4, 5~~);
int[] newArr = new in[5];

for (int i=0 ; i ~~<10~~; i++){
~~Ans~~ int max in Ret = FindMax (arr, index+1);
Return Math.max (arr[index], max in Rst);

Page No.:

cheak. 2, 5, 9, 1, 6.

1) cheak. 2
2) cheak. 5
3) cheak 9
4) cheak 1
5) cheak 6.

then in all of this, find maximum NO. 9.

Q.6. Cheak if an array is sorted (stickly increasing)

PS:

Given an array, cheak whether it is sorted in stickly increasing order using recursion.

ex-1. Input arr = [1, 2, 3, 4]
output = true

→

int[]arr (1, 2, 3, 4);
int[]
for (int=0; i<4 ; i++);
count ++;
print ();

step 1

0 =

if. (index >=arr.length-1){
return true;
}
else; return false;

Date : / /
Page No.:

int [] arr (1,2,3,4);
for (int = 2; i < 4 ; i++);
count ++;

2 ≡ 8

print ();

5

step 3 → int [] arr (1,2,3,4);
for (int = 3 ; i < 4 ; i++);
count ++;
print ();

3 5

step 4 int [] arr (1,2,3,4);
for (int = 4 ; i < 4 ; i++)
count ++;
print ();

4 5

output = 1,2,3,4 = true

2] Input : arr = [1,2,2,3]
output : false

→

int [] arr (1,2,2,4)
for (int = 1 ; i < 4 ; i++)
count++;
if (index) = arr.length - 1) {
return true;
else; return false;

Date : / /
Page No.:

Q.7. Cheak if No. is prime

PS :- Given an integer N, check whether it is prime or not A prime No. is number that is only divisible by 1 and itself & total No. of divisors is 2.

Ex.1 Input N = 2
Output : true

→

```
public static boolean isPrime(int n){
    int count = 0;
    for (int i = 1; i < Math.sqrt(2); i++){
        if (n % i == 0){
            count ++;
        if (n / i != i) {
            count ++;
        }
    }
}
if count == 2;
    return true;
}
    return false;
}
```

2 have two divisor 1 and 2 itself.
∴ it is prime No.

Ex. 7. N = 10 output : false

→

```
public static boolean isPrime(int n){
    int count = 0;

    for(int i=1; i<Math.sqrt(n);i++){
        count++;
        if(n%i==1){
            count++;
        }
    }
}
if(count==2){
    return true;
}
return false;
}
```

but it has 4 divisors 1, 2, 5 and 10.
so it is not prime no.
It print false.

Q. 8. find first index of element in array.

PS: Given array and key find first occurence (index) of key using recursion.

ex 1. Input arr = [4, 2, 7, 7, 9], key = 7
output : 2

→

int [] arr = {4, 2, 7, 7, 9}
int [] newArr = new int[5];

for (int i = 0; i < 5; i++) {
newArr[i] = arr[arr.length - 1 - i]
}

for (
~~index~~
if (arr[index] == key) {
return index;
}
return findFirstIndex(arr, index + 1, key);
}

4	2	7	7	9
0	1	2	3	4

7 have 4 index.

recursive of array →

4	2	7	7	9

↓

9	7	7	2	4
0	1	2	3	4

Date: / /
Page No.:

on 2 index. 7. is Number

so,

output = 7

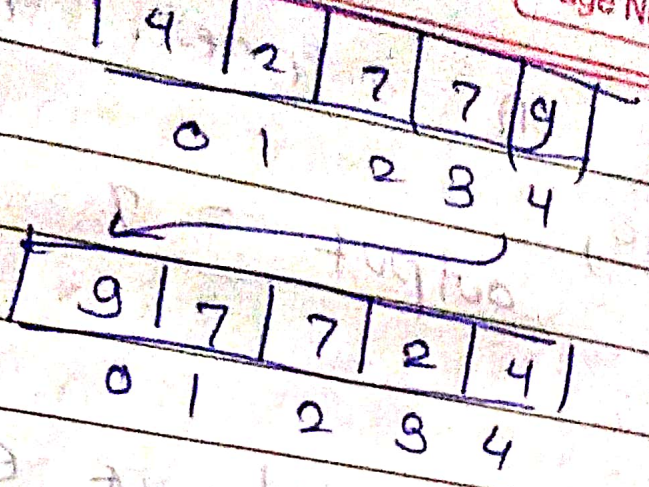
Q. 9. Find last index of element in array

PS: Given array and key, find last occrane (index) of key using recursion.

Ex 1. Input arr = {4, 2, 7, 7, 9}, key = 7
output = 3.

→

```
Int [] arr = {4,2,7,7,9}
Int [] newArr = new int[9];
for (int i = 0; i < 9; i++){
    new arr[i] = arr[arr.
        length - 1 - i]
}
if (arr[index] == key){
    return index;
}
return findFirst index (arr,
    index - 1, key);
}
```



In last occurance (index) of ~~the index~~. 7 is at the index. 3

Q.10. Reverse number using recursion

PS : Given integer N, reverse its digits using recursion.

Ex.2. Input N = 1234.
Output : 4321

→
```
main () {
    int drs = f(4);
}
int f(int n) {
    if (n==0 || n==1) {
        return n;
    }
    start
    int last = f(n-1);
    int second last = f(n-2);
    return last + second last
                  start
```

Date : / /
Page No.:

```
Int F(int n) {
    if (n == 0) || n == 1)
        return n;
    }
    start
    int start = F(n-1)
    int second start = F(n-2);
    Return start + secnd start;
```

```
Int F(int n) {
    if (n == 0) || n == 1)
        return n;
    }
    int start = F(n-1)
    Int second start = F(n-2);
    Retun start + second start;
```

```
Int f (int n) {
    if (n == 0 || n == 1)
        return n;
    }
    int start = f(n-1)
    Int second start + secnd start;
```

3
~~f(0)~~
f(1)
f(2)
f(3)
f(4)

= 4, 3, 2, 1

Q.11. count how many time, digit appears in number.

PS: Given an integer N and digit D cou. how many time D appears in N using recurssion.

Ex 1. N = 717237, D=7
output = 3

→
```
int f (int n) {
    if (n == 20) {
        Return 0;
    }
    int count = (N%10 == D);
    return count + countdigit occurac
                (N/10, D);
    S.O.Pln();
}
```

Q.12. cheak if number is palindrome

Ex 1. N = 121
output : true.

→
```
int strt = 0
Int end = str.length()-1;
```

Date: / /
Page No.:

while (start < end) {
if (str.charAt(start)
!= str.charAt(end)) {
Return false;
}
start ++;

= 1 2 1

= reverse = 121.

so it is a palindrome.

Q.13. Find GCD of two numbers using recursion.

PS: Given two integers A and B find their greatest common divisor using recursion.

ex 1: Input A = 24, B = 36

→ public static int gcd (int 24, int 36);

int min = Math.min (24, 36)

int gcd = 1;

```
for (int i = min; i >= 1; i--)
    if (a % i == 0 && b % i
        == 0)
        return i;
}
```

divisor of 24 = 1, 2, 4, 6, 8, 12,
divisor of 36 = 1, 2, 4, 6, 9, 12,

Highest common divisor = 12.

output = 12