String interview programs

1. Ascending Order Of String

```
String s="TESTYANTRA";

for (int i = 0; i < s.length(); i++) {
        System.out.println(s.charAt(i));}}}
```

2. DesendingOrderOfString

```
String s="TESTYANTRA";
for (int i =s.length()-1; i>=0; i--) {
        System.out.println(s.charAt(i));
}
```

3. ReverseStringWithThirdVariable

```
String s="TESTYANTRA";
        String b="";
        for(int i=s.length()-1;i>=0;i--) {
                b=b+s.charAt(i);

                //System.out.print(s.charAt(i)+b);
        }
        //using second way
        System.out.println(b);
}
```

4. ReverseStringUsingThirdVariableWithOutLength

```
        String s="TESTYANTRA";
        String b="";
        char[] ch=s.toCharArray();
        int count=0;
        String rev="";
        for (char cha:ch) {
                count++;
        }
        for(int i=count-1;i>=0;i--) {
                rev=s.charAt(i)+b;
                System.out.print(rev);
        }
```

5. ReverseStringWithoutLengthMethod

```
String s="TESTYANTRA";
char[] ch=s.toCharArray();
int count=0;
for (char cha:ch) {
        count++;
}
for(int i=count-1;i>=0;i--) {
        System.out.print(s.charAt(i));
}
```

6. ReverseStringUsingPallendrome

```
public static void main(String[] args) {
        String s="1221";
        char[] ch=s.toCharArray();
        int count=0;
        String rev="";
        for (char cha:ch) {
                count++;
        }
        for(int i=count-1;i>=0;i--) {
                rev=s.charAt(i)+rev;

        }
        if (s.equals(rev)) {
                System.out.println("pallendrome number");
        }
        else {
                System.out.println("not pallendrome number");
        }
```

7. **CountStringCharacterRepeatation**

```
        String s="accurance";
        HashSet<Character>hs=new HashSet<Character>();
        for (int i = 0; i < s.length(); i++) {
        hs.add(s.charAt(i));        }
        for (Character ch : hs) {
                int count=0;
                for (int i = 0; i <s.length(); i++) {
                        if (ch.equals(s.charAt(i))) {
                                count++;
```

```
                    }}
        System.out.println(ch+" : is  accuring : "+count);    }
```

8. **PrintOnlyUniqueCharacter**

```
                String s="accurance";
            HashSet<Character>as=new HashSet<Character>();
            for (int i = 0; i < s.length(); i++)
            {
            as.add(s.charAt(i));
            }
            for (Character ch : as) {
                int count=0;
                for (int i = 0; i<s.length(); i++) {
                    if (ch==(s.charAt(i))) {
                        count++;      }}
if(count<2) { System.out.println(ch+" : is  accuring unique: "+count); }}
```

9. **OnlyDuplicateAccurance**

```
            String s="accurance";
            HashSet<Character>as=new HashSet<Character>();
            for (int i = 0; i < s.length(); i++) {
            as.add(s.charAt(i));
            }
            for (Character ch : as) {
                int count=0;
                for (int i = 0; i<s.length(); i++) {
                    if (ch==(s.charAt(i))) {
                        count++;}}
                if(count>1) {
System.out.println(ch+" : is  accuring duplicate : "+count);}}
```

11.segration of alphabets without array and collections

```
String s="abcdabcdabcdabcdabcd";
LinkedHashSet<Character>as=new LinkedHashSet<Character>();
for (int i = 0; i < s.length(); i++)
{
as.add(s.charAt(i));
}
for (Character ch : as) {
int count=0;
for (int i = 0; i<s.length(); i++) {
if (ch==(s.charAt(i))) {
System.out.print(ch);
}}
System.out.println();
}
```

10. **PrintPosition**

```
                String s="accurance";
LinkedHashSet<Character>as=new LinkedHashSet<Character>();
        for (int i = 0; i < s.length(); i++)
        {
        as.add(s.charAt(i));
        }
        for (Character ch : as) {
                int count=0;
                for (int i = 0; i<s.length(); i++) {
                        if (ch==(s.charAt(i))) {
                                count++;
                        }}
 If (count<2)
{ System.out.println(ch+":unique"+s.indexOf(ch)); }}
```

12.Tester o/p:: t4e5s3r6

```
        String s="Tester";
        s=s.toLowerCase();
LinkedHashSet<Character>as=new LinkedHashSet<Character>();
        for (int i = 0; i < s.length(); i++)
        {
        as.add(s.charAt(i));
        }
        for (Character ch : as) {
        for (int i=s.length()-1;i>=0;i--) {
        if (ch==s.charAt(i)) {
        System.out.print(ch+""+(i+1));
                        break;}
```

13.khatam tat a by by o/p::khatham::1 ta::2 bye::2

```
String s="khatham ta ta bye bye";
String[] str=s.split(" ");
LinkedHashSet<String>hs=new LinkedHashSet<String>();
for (int i = 0; i < str.length; i++)
{
hs.add(str[i]);
}
for (String ch : hs) {
int count=0;
for (int i = 0; i <str.length; i++) {
if (ch.equals(str[i])) {
count++;
}}
System.out.println(ch+" :: "+count);}
```

14.unique word from sentence o/p:: khatham

```
String s="khatham ta ta bye bye";
String[] str=s.split(" ");
LinkedHashSet<String>hs=new LinkedHashSet<String>();
for (int i = 0; i < str.length; i++)
{
hs.add(str[i]);
}
for (String ch : hs) {
int count=0;
for (int i = 0; i <str.length; i++) {
if (ch.equals(str[i])) {
count++;
}}
if (count==1)
{ System.out.println(ch); }}
```

15.0/p: mahtahk at at eyb eyb

```
String s="khatham ta ta bye bye";
String[] str=s.split(" ");
for (int i = 0; i < str.length; i++) {
String a=str[i];
for (int j =a.length()-1; j>=0; j--) {
System.out.print(a.charAt(j));
}
System.out.print(" ");}}}
```

16.o/p: a3b1c1a3

```
String s="aaabcaaa";
        for (int i = 0; i < s.length();i++ )
        {       int count=1;
                for (int j = i+1; j < s.length(); j++)
                {
                        if(s.charAt(i)==s.charAt(j))
                        {
                                count++;
                                i++;
                        }
                        else
                        {break;
                        }}
                System.out.print(s.charAt(i)+""+count);}}
```

17. o/p:: 5 1 2 3 4

```
int a[]= {1,2,3,4,5};
        int key=4;
        for (int j = 0; j < key; j++) {
                int temp=a[0];
                for (int i = 1; i < a.length; i++);
                        a[i-1]=a[i];
                }
                a[a.length-1]=temp;
        }
        for (int i = 0; i < a.length; i++) {
                System.out.print(a[i]+" ");
        }}
```

18.happy number o/p::1

```
int n=568;
        while(n>9)
        {
            int sum=0;
            while(n>0)
            {
                int rem=n%10;
                sum=sum+rem;
                n=n/10;
            }
            n=sum;              }
        System.out.println(n);
    }
```

19.print character from string in increasing order.

```
String s="india";
        for (int i = 0; i <s.length(); i++)
        {
        for (int j = 0; j <=i; j++) {
            System.out.print(s.charAt(j)+" ");
        }
        System.out.println();
        }}
```

20. CountingVovelsInString

```
        String s="india";  //3
        int count=0;
        for (int i = 0; i < s.length(); i++) {

    if(s.charAt(i)=='a'||s.charAt(i)=='e'||s.charAt(i)=='i'|
|s.charAt(i)=='o'||s.charAt(i)=='u')
    {
            count++;
    }    }
        System.out.println(s+"="+count);

    }}
```

21. Count Only unique Vovels o/p:2

```
          String s="india";
          int count=0;
LinkedHashSet<Character>hs=new LinkedHashSet<Character>();
          for (int i = 0; i < s.length(); i++) {
               hs.add(s.charAt(i));
          }
          for (Character character : hs) // i n d a
          {    if
(character=='a'||character=='e'||character=='i'||character==
'o'||character=='u')
               {
               count++;
               }
          }
          System.out.println(s+" "+count);
     }}
```

22. Count unique Vovels From String Array o/p::1 2 2

```
     String []str= {"hi","hello","india"};
          for (int i = 0; i < str.length; i++) {
                    String s=str[i];
                    int count=0;
LinkedHashSet<Character> hs=new LinkedHashSet<Character>();
     for (int j = 0; j < s.length(); j++) {
          hs.add(s.charAt(j));
                     }
     for (Character character : hs) // i n d a
     {
     if
(character=='a'||character=='e'||character=='i'||character==
'o'||character=='u')
{
          count++;
}}
System.out.println(s+" "+count);
}}
```