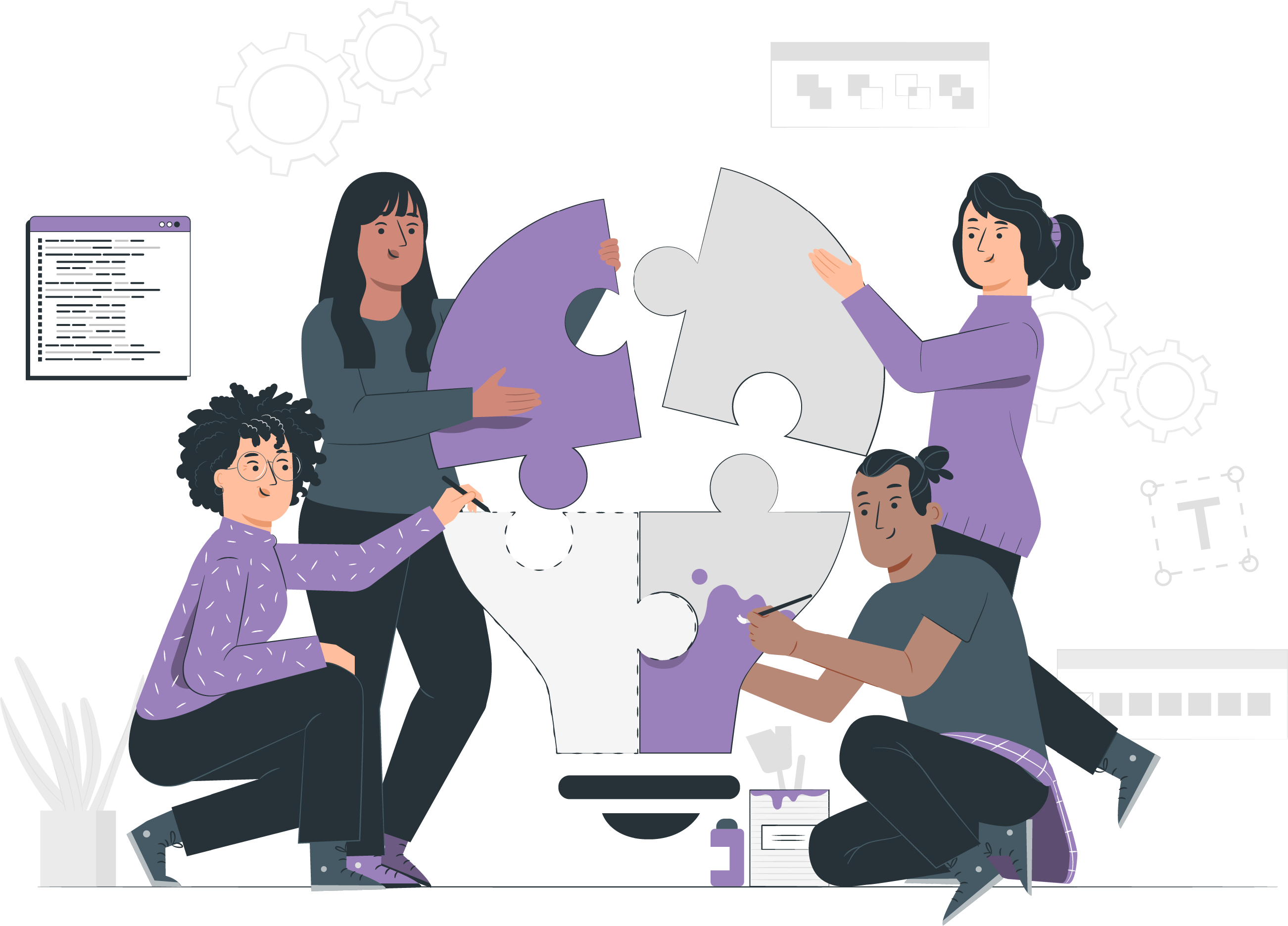
1.WAP(Write a Program) to remove Duplicates from a String.(Take any String example with duplicates character).



=> public class RemoveDuplicate {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 System.*out*.println("Enter String");  
 String s=sc.nextLine();  
 String t="";  
 int i,j;  
 for ( i = 0; i < s.length(); i++) {  
 for ( j = 0; j < i; j++) {  
 if (s.charAt(i)==s.charAt(j)){  
 break;  
 }  
 }if(i==j){  
 t=t+s.charAt(i);  
 }  
 }  
 System.*out*.println(t);  
 }  
}

2. WAP to print Duplicates characters from the String.

=> public class PrintDuplicateCharacter {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 System.*out*.print("Enter String : ");  
 String s=sc.nextLine();  
 s=s.replace(" ","");  
 char ch[]=s.toCharArray();  
 Arrays.*sort*(ch);  
 int count,i,j;  
 System.*out*.print("Duplicate value in this string is : ");  
 for ( i=0; i < ch.length; i++) {  
 count=1;  
 for (j = i+1; j <ch.length; j++) {  
 if(ch[i]==ch[j]){  
 ch[j]='0';  
 count++;  
 }  
 }if (count>1 && ch[i]!='0'){  
 System.*out*.print(ch[i]);  
 }  
 }  
 }  
}

3. WAP to check if “2552” is palindrome or not.

=> ***Method1-***

public class PalindromeProgram {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 String str1 =sc.nextLine();  
 String str2="";  
 for (int i = 0; i < str1.length(); i++) {  
 str2=str2+(str1.charAt(str1.length()-(i+1)));  
 }  
 System.*out*.println(str1.equalsIgnoreCase(str2));  
 }  
}

***Method2-***

public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 String str1 =sc.nextLine();  
 *//Method1* boolean check=true;  
 for (int i = 0; i < str1.length()/2; i++) {  
 if(str1.charAt(i)!=str1.charAt(str1.length()-(i+1))){  
 check=false;  
 }  
 }  
 System.*out*.println(check);  
 }  
}

4.WAP to count the number of WAP to count the number of consonants, vowels, special characters in a String.in a String.

=>public class Count\_Vonsonants\_Vowels\_specialCharacters {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 System.*out*.print("Enter String : ");  
 String str=sc.nextLine();  
 str=str.toLowerCase();  
 str=str.replace(" ","");  
 int vowels = 0, consonant = 0, specialChar = 0;  
 for (int i = 0; i < str.length(); i++) {  
  
 char ch = str.charAt(i);  
  
 if ( (ch >= 'a' && ch <= 'z') ) {  
  
 if (ch == 'a' || ch == 'e' || ch == 'i' ||  
 ch == 'o' || ch == 'u')  
 vowels++;  
 else  
 consonant++;  
 }  
 else  
 specialChar++;  
 }  
 System.*out*.println("Vowels: " + vowels);  
 System.*out*.println("Consonant: " + consonant);  
 System.*out*.println("Special Character: " + specialChar);  
 }  
}

5. WAP to implement Anagram Checking least inbuilt methods being used.

=> public class AnagramProgram {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 System.*out*.println("Enter 1st String");  
 String str1=sc.nextLine();  
 System.*out*.println("Enter 1st String");  
 String str2=sc.nextLine();  
  
 str1=str1.replace(" ","");  
 str2=str2.replace(" ","");  
  
 str1=str1.toLowerCase();  
 str2=str2.toLowerCase();  
  
 char ch1[]=str1.toCharArray();  
 char ch2[]=str2.toCharArray();  
  
 Arrays.*sort*(ch1);  
 Arrays.*sort*(ch2);  
  
 if(Arrays.*equals*(ch1,ch2)){  
 System.*out*.println("It's an Anagram");  
 }else System.*out*.println("Its not an Anagram");  
 }  
}

6. WAP to implement Pangram Checking with least inbuilt methods being used.

=> public class PangramProgram {  
 public static void main(String[] args) {  
   
 Scanner sc=new Scanner(System.*in*);  
 String str1=sc.nextLine();  
  
 str1=str1.replace(" ","");  
  
 str1=str1.toLowerCase();  
  
 char ch[]=str1.toCharArray();  
  
 int arr[]=new int[26];  
  
 for (int i = 0; i < ch.length; i++) {  
 arr[ch[i]-97]++;  
 }  
  
 boolean flag=true;  
 for (int i = 0; i < arr.length; i++) {  
 if(arr[i]!=0){  
 continue;  
 }else {  
 flag=false;  
 break;  
 }  
 }  
  
 if (flag==true){  
 System.*out*.println("Its a pangram");  
 }else System.*out*.println("Its not a pangram");  
 }  
}

7. WAP to find if String contains all unique characters.

=> ***Method1-***

public class CheckUniqueCharecter {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 System.*out*.print("Enter String : ");  
 String s=sc.nextLine();boolean check=true;  
 for (int i = 0; i < s.length(); i++) {  
 for (int j = i+1; j <s.length() ; j++) {  
 if (s.charAt(i)==s.charAt(j)){  
 check=false;  
 break;  
 }  
 }  
 }if (check==true){  
 System.*out*.println("String contain all unique charecters");  
 }else System.*out*.println("String has duplicate charecters")  
   
 }  
}

***Method2-***

public class CheckUniqueCharecter {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 System.*out*.print("Enter String : ");  
 String s=sc.nextLine();  
char ch[]=s.toCharArray();  
 Arrays.*sort*(ch);  
 boolean c=true;  
 for (int i = 0; i < ch.length-1 ; i++) {  
 if (ch[i]==ch[i+1]){  
 c=false;  
 break;  
 }  
 }  
 if (c==true){  
 System.*out*.println("String contain all unique charecters");  
 }else System.*out*.println("String has duplicate charecters");  
   
 }  
}

8. WAP to find the maximum occurring character in a String.

=> public class PrintDuplicateCharacter {  
 public static void main(String[] args) {  
 Scanner sc=new Scanner(System.*in*);  
 System.*out*.print("Enter String : ");  
 String s=sc.nextLine();  
 s=s.replace(" ","");  
 char ch[]=s.toCharArray();  
 Arrays.*sort*(ch);  
 int count,i,j;  
 System.*out*.print("Duplicate value in this string is : ");  
 for ( i=0; i < ch.length; i++) {  
 count=1;  
 for (j = i+1; j <ch.length; j++) {  
 if(ch[i]==ch[j]){  
 ch[j]='0';  
 count++;  
 }  
 }if (count>1 && ch[i]!='0'){  
 System.*out*.print(ch[i]);  
 }  
 }  
 }  
}