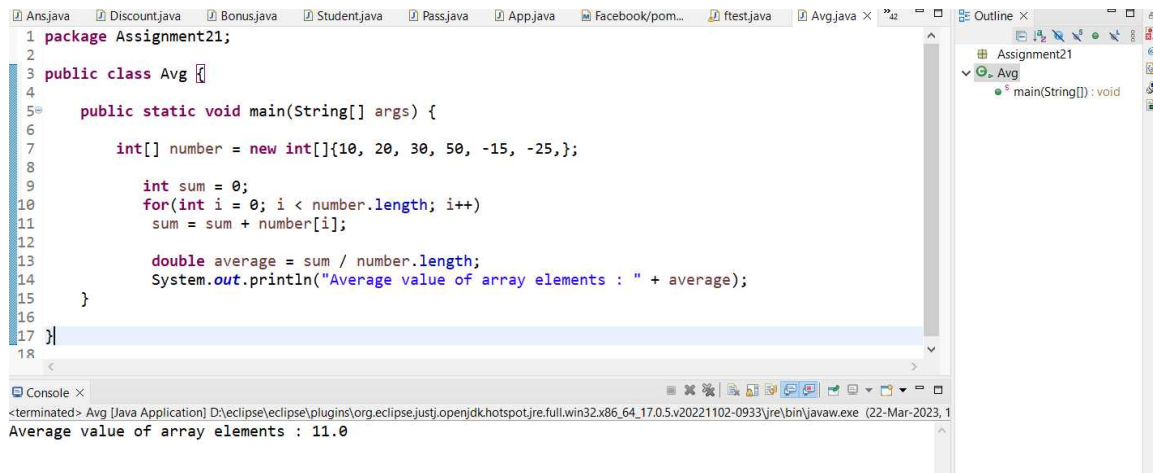


## Assignment-21

### Arrays, Strings, Abstract Classes & File Handling

1. Write a Java program to calculate the average value of array elements.

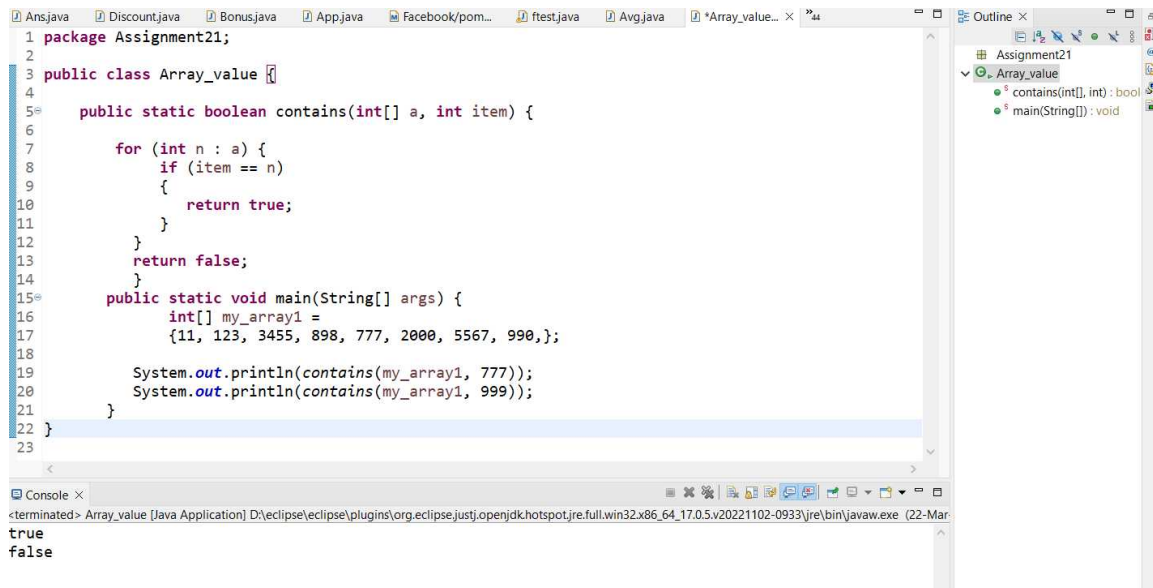


The screenshot shows the Eclipse IDE with a Java project named 'Assignment21'. The main editor displays the following code:

```
1 package Assignment21;
2
3 public class Avg {
4
5     public static void main(String[] args) {
6
7         int[] number = new int[]{10, 20, 30, 50, -15, -25,};
8
9         int sum = 0;
10        for(int i = 0; i < number.length; i++)
11            sum = sum + number[i];
12
13        double average = sum / number.length;
14        System.out.println("Average value of array elements : " + average);
15    }
16 }
17
18
```

The console output at the bottom shows: `Average value of array elements : 11.0`. The Outline view on the right shows the project structure with 'Assignment21' and 'Avg' class.

2. Write a Java program to test if an array contains a specific value.



The screenshot shows the Eclipse IDE with a Java project named 'Assignment21'. The main editor displays the following code:

```
1 package Assignment21;
2
3 public class Array_value {
4
5     public static boolean contains(int[] a, int item) {
6
7         for (int n : a) {
8             if (item == n)
9                 return true;
10        }
11        return false;
12    }
13
14    public static void main(String[] args) {
15        int[] my_array1 =
16            {11, 123, 3455, 898, 777, 2000, 5567, 990,};
17
18        System.out.println(contains(my_array1, 777));
19        System.out.println(contains(my_array1, 999));
20    }
21 }
22
23
```

The console output at the bottom shows: `true` and `false`. The Outline view on the right shows the project structure with 'Assignment21' and 'Array\_value' class.

3. Write a Java program to remove a specific element from an array.

The screenshot shows an IDE with a Java file named `Remove.java`. The code defines a class `Remove` with a `main` method. It initializes an array `my_array` with values `{5, 10, 15, 20, 25, 50, 100}`. It then prints the original array. Next, it sets `removeIndex` to 0 and enters a loop from `i = removeIndex` to `my_array.length - 1`. Inside the loop, it shifts each element one position to the left (`my_array[i] = my_array[i + 1]`). Finally, it prints the array after removing the first element. The console output shows the original array and the array after removal, where the first element (5) has been removed.

```
1 package Assignment21;
2 import java.util.*;
3
4 public class Remove {
5
6     public static void main(String[] args) {
7         int[] my_array = {5, 10, 15, 20, 25, 50, 100};
8         System.out.println("Original Array : "+Arrays.toString(my_array));
9
10        int removeIndex = 0;
11
12        for(int i = removeIndex; i < my_array.length -1; i++)
13        {
14            my_array[i] = my_array[i + 1];
15        }
16        System.out.println("After removing the first element: "+Arrays.toString(my_array));
17    }
18 }
```

Console Output:

```
<terminated> Remove [Java Application] D:\eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-20)
Original Array : [5, 10, 15, 20, 25, 50, 100]
After removing the first element: [10, 15, 20, 25, 50, 100, 100]
```

4. Write a Java program to copy an array by iterating the array.

The screenshot shows an IDE with a Java file named `Iterating.java`. The code defines a class `Iterating` with a `main` method. It initializes an array `a` with values `{5, 10, 15, 20, 25}` and creates a new array `b` of the same length. It then iterates through array `a` and copies each element into array `b`. Finally, it prints both arrays. The console output shows the elements of array `a` and array `b`, which are identical.

```
1 package Assignment21;
2
3 public class Iterating {
4
5     public static void main(String[] args) {
6         int a[] = {5, 10, 15, 20, 25};
7         int b[] = new int[a.length];
8
9         b = a;
10
11        System.out.println("Elements of array a[ ]");
12        for (int i = 0; i < a.length; i++)
13            System.out.print(a[i] + " ");
14
15        System.out.println("\nElements of array b[ ]");
16        for (int i = 0; i < b.length; i++)
17            System.out.print(b[i] + " ");
18    }
19 }
20 }
```

Console Output:

```
<terminated> Iterating [Java Application] D:\eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-20)
Elements of array a[ ]
5 10 15 20 25
Elements of array b[ ]
5 10 15 20 25
```

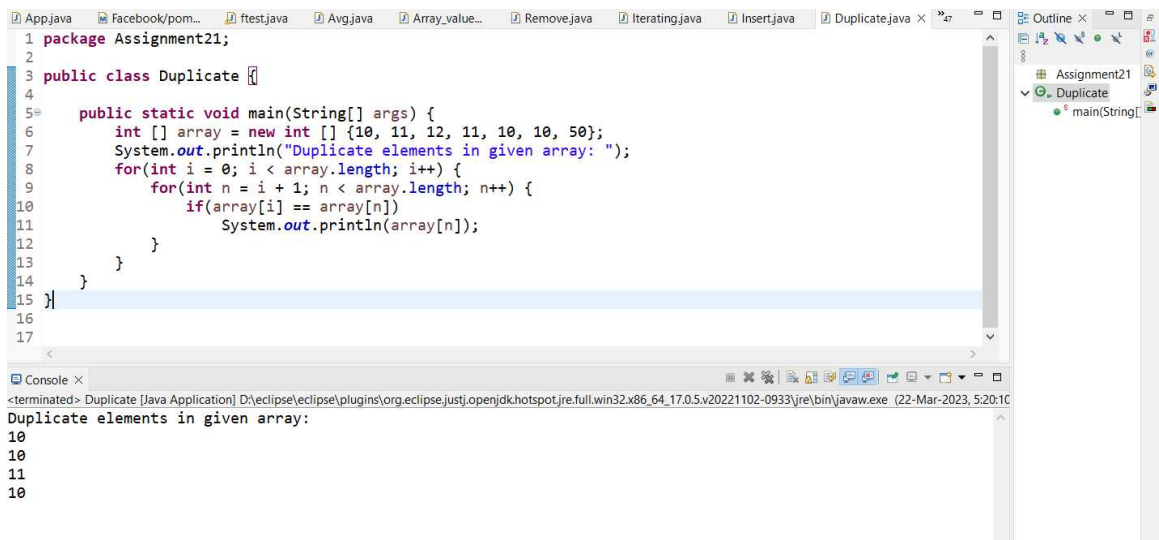
5. Write a Java program to insert an element (specific position) into an array.



```
1 package Assignment21;
2 import java.util.*;
3
4 public class Insert {
5
6     public static void main(String[] args) {
7         int[] my_array = {1, 2, 3, 4, 6, 7};
8         int Index_position = 4;
9         int newValue = 5;
10        System.out.println("Original Array : "+Arrays.toString(my_array));
11        for(int i=my_array.length-1; i > Index_position; i--){
12            my_array[i] = my_array[i-1];
13        }
14        my_array[Index_position] = newValue;
15        System.out.println("New Array: "+Arrays.toString(my_array));
16    }
17 }
```

<terminated> Insert [Java Application] D:\eclipse\workspace\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 5:14:07 pm)  
Original Array : [1, 2, 3, 4, 6, 7]  
New Array: [1, 2, 3, 4, 5, 6]

6. Write a Java program to find the duplicate values of an array of integer values.



```
1 package Assignment21;
2
3 public class Duplicate {
4
5     public static void main(String[] args) {
6         int [] array = new int [] {10, 11, 12, 11, 10, 10, 50};
7         System.out.println("Duplicate elements in given array: ");
8         for(int i = 0; i < array.length; i++) {
9             for(int n = i + 1; n < array.length; n++) {
10                if(array[i] == array[n])
11                    System.out.println(array[n]);
12            }
13        }
14    }
15 }
16
17 }
```

<terminated> Duplicate [Java Application] D:\eclipse\workspace\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 5:20:10 pm)  
Duplicate elements in given array:  
10  
11  
10

7. Write a Java program to find the common elements between two arrays of integers.

The screenshot shows an IDE with a Java file named 'common.java'. The code defines a package 'Assignment21' and a class 'common'. Inside the 'main' method, two integer arrays are declared: 'array1' with values {7, 8, 9, 33, 55} and 'array2' with values {10, 11, 33, 9, 55, 100}. A nested loop iterates through both arrays. For each element in 'array1', it checks if it exists in 'array2'. If found, it prints the element. The console output shows the values 9, 33, and 55.

```
1 package Assignment21;
2
3 public class common {
4
5     public static void main(String[] args) {
6         int[] array1 = {7, 8, 9, 33, 55};
7         int[] array2 = {10, 11, 33, 9, 55, 100};
8
9         for(int i = 0; i < array1.length; i++) {
10             for(int j = 0; j < array2.length; j++) {
11                 if(array1[i] == array2[j]) {
12                     System.out.println(array1[i]);
13                 }
14             }
15         }
16     }
17 }
```

Console output:  
<terminated> common [Java Application] D:\eclipse\workspace\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 5:24:57)  
9  
33  
55

8. Write a Java program to remove duplicate elements from an array.

The screenshot shows an IDE with a Java file named 'removeD.java'. The code defines a package 'Assignment21' and a class 'removeD'. It includes a method 'removeDuplicateElements' that takes an array and its length as input. It uses a temporary array to store unique elements. The 'main' method initializes an array {1, 2, 2, 3, 3, 4, 4, 4, 4}, calls the 'removeDuplicateElements' method, and prints the resulting array. The console output shows the array [1, 2, 3, 4].

```
1 package Assignment21;
2
3 public class removeD {
4     public static int removeDuplicateElements(int arr[], int n){
5         if (n==0 || n==1){
6             return n;
7         }
8         int[] temp = new int[n];
9         int j = 0;
10        for (int i=0; i<n-1; i++){
11            if (arr[i] != arr[i+1]){
12                temp[j++] = arr[i];
13            }
14        }
15        temp[j++] = arr[n-1];
16        for (int i=0; i<j; i++){
17            arr[i] = temp[i];
18        }
19        return j;
20    }
21
22    public static void main (String[] args) {
23        int arr[] = {1, 2, 2, 3, 3, 4, 4, 4, 4};
24        int length = arr.length;
25        length = removeDuplicateElements(arr, length);
26        for (int i=0; i<length; i++)
27            System.out.print(arr[i]+" ");
28    }
29 }
30 }
```

Console output:  
<terminated> removeD [Java Application] D:\eclipse\workspace\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 5:29:39)  
1 2 3 4

9. Write a Java program to find the second largest element in an array.

```

1 package Assignment21;
2
3 public class SecondLargest {
4
5     public static int getSecondLargest(int[] a, int total){
6         int temp;
7         for (int i = 0; i < total; i++)
8             {
9                 for (int j = i + 1; j < total; j++)
10                    {
11                        if (a[i] > a[j])
12                        {
13                            temp = a[i];
14                            a[i] = a[j];
15                            a[j] = temp;
16                        }
17                    }
18            }
19            return a[total-2];
20        }
21
22     public static void main(String args[]){
23         int a[]={11, 22, 33, 99, 9};
24         System.out.println("Second Largest: "+getSecondLargest(a,4));
25     }
26 }

```

Console x

<terminated> SecondLargest [Java Application] D:\eclipse\workspace\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 5:42:11 pm) [pid: 10512]

Second Largest: 33

10. Write a Java program to find smallest and second smallest elements of a given array.

```

1 package Assignment21;
2
3 public class smallest {
4
5     public static void main(String[] args) {
6         int arr[] = {-1, 1, 0, 1, 2};
7
8         int first_element, second_element, arr_size = arr.length;
9         if (arr_size < 2)
10            {
11                System.out.println("Array size less than two.");
12                return;
13            }
14         first_element = second_element = Integer.MAX_VALUE;
15         for (int i = 0; i < arr_size; i++)
16            {
17                if (arr[i] < first_element)
18                {
19                    second_element = first_element;
20                    first_element = arr[i];
21                }
22                else if (arr[i] < second_element && arr[i] != first_element)
23                    second_element = arr[i];
24            }
25         if (second_element == Integer.MAX_VALUE)
26             System.out.println("No second smallest element.");
27         else
28             System.out.println("The smallest element is " + first_element + " and second Smallest element is " + second_element);
29     }
30 }

```

Console x

<terminated> smallest [Java Application] D:\eclipse\workspace\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86\_64\_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 5:42:11 pm - 5:42:12 pm) [pid: 10512]

The smallest element is -1 and second Smallest element is 0.

11. Write a Java program to get the character (Unicode code point) at the given index within the String.

The screenshot shows the Eclipse IDE with a Java file named `Char.java`. The code defines a package `Assignment21` and a class `Char` with a `main` method. The `main` method takes a string `"Rana Akshat"` and prints its original form, then the Unicode point of the first character (97) and the second character (110). The console output shows these results.

```
1 package Assignment21;
2
3 public class Char {
4
5     public static void main(String[] args) {
6         String str = "Rana Akshat";
7         System.out.println("Original String : " + str);
8         int val1 = str.codePointAt(1);
9         int val2 = str.codePointAt(2);
10
11         System.out.println("Character(unicode point) = " + val1);
12
13         System.out.println("Character(unicode point) = " + val2);
14     }
15 }
16
```

Console Output:

```
<terminated> Char [Java Application] D:\eclipse\workspace\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 5:47:26 pm - 5:47:27 pm) [pid: 15336]
Original String : Rana Akshat
Character(unicode point) = 97
Character(unicode point) = 110
```

12. Write a program to remove first occurrence of a character from string.

The screenshot shows the Eclipse IDE with a Java file named `occurrence.java`. The code defines a package `Assignment21` and a class `occurrence` with a `main` method. The `main` method takes a string `"Height of a person!"`, prints it, and then replaces the first occurrence of the character 'H' with 'L'. The console output shows the original string and the modified string.

```
1 package Assignment21;
2
3 public class occurrence {
4
5     public static void main(String[] args) {
6         String st = "Height of a person!";
7         System.out.println("String: "+st);
8         String str = st.replaceFirst("(?>H)+", "L");
9         System.out.println("String after replacing a characters first occurrence: "+str);
10     }
11 }

```

Console Output:

```
<terminated> occurrence [Java Application] D:\eclipse\workspace\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 5:53:40)
String: Height of a person!
String after replacing a characters first occurrence: Leight of a person!
```

13. Write a program to remove last occurrence of a character from string.

The screenshot shows the Eclipse IDE with a Java file named `lastOcc.java`. The code defines a package `Assignment21` and a class `lastOcc` with a `main` method. The `main` method takes a string `"Java is a programming language !"`, creates a `StringBuffer` from it, and deletes the last character. The console output shows the original string and the modified string.

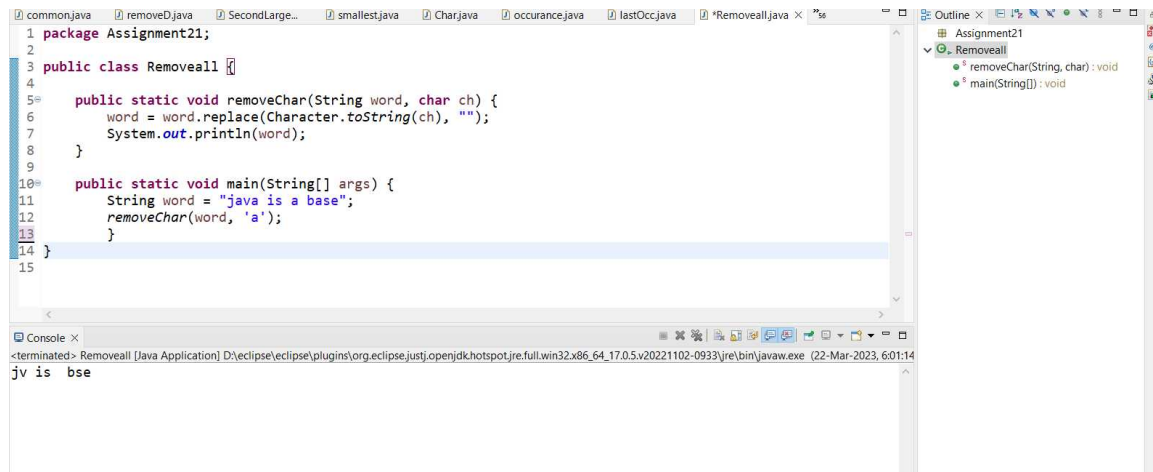
```
1 package Assignment21;
2
3 public class lastOcc {
4
5     public static void main(String[] args) {
6         String string = "Java is a programming language !";
7         StringBuffer sb= new StringBuffer(string);
8         sb.deleteCharAt(sb.length()-1);
9         System.out.println(sb);
10     }
11 }
12
```

Console Output:

```
<terminated> lastOcc [Java Application] D:\eclipse\workspace\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 5:58:11 pm)
Java is a programming language
```



14. Write a program to remove all occurrences of a character from string.

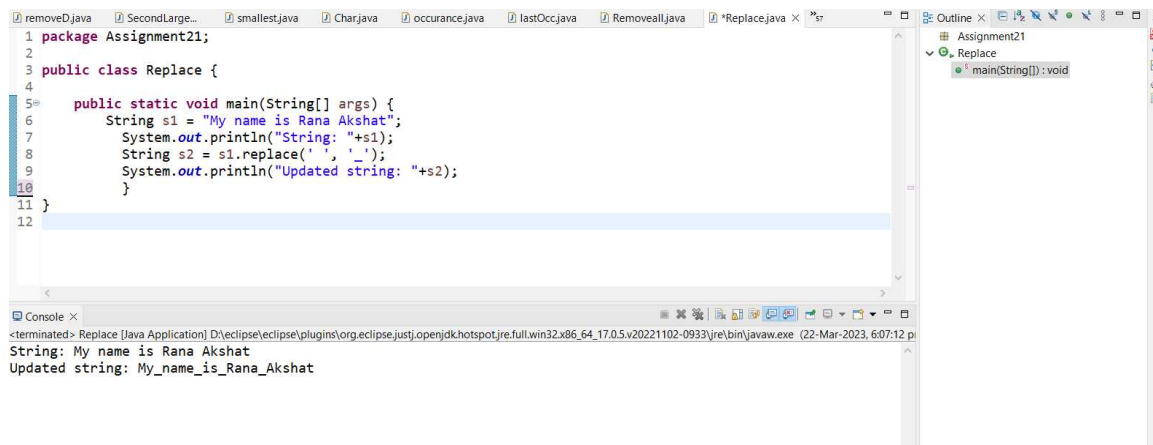


The screenshot shows an IDE with a Java file named `Removeall.java`. The code defines a package `Assignment21` and a class `Removeall`. It includes a static method `removeChar` that takes a string and a character, replaces the character with an empty string, and prints the result. The `main` method calls `removeChar` with the string "java is a base" and the character 'a'. The console output shows "jv is bse".

```
1 package Assignment21;
2
3 public class Removeall {
4
5     public static void removeChar(String word, char ch) {
6         word = word.replace(Character.toString(ch), "");
7         System.out.println(word);
8     }
9
10    public static void main(String[] args) {
11        String word = "java is a base";
12        removeChar(word, 'a');
13    }
14 }
15
```

Console Output:  
<terminated> Removeall [Java Application] D:\eclipse\plugins\org.eclipse.justi.openjdk hotspot\jre.full.win32.x86\_64\_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 6:01:14)  
jv is bse

15. Write a Java program to replace a specified character with another character.



The screenshot shows an IDE with a Java file named `Replace.java`. The code defines a package `Assignment21` and a class `Replace`. It includes a static method `main` that takes an array of strings, replaces the first space in the first string with an underscore, and prints the original and updated strings. The console output shows the original string "My name is Rana Akshat" and the updated string "My\_name\_is\_Rana\_Akshat".

```
1 package Assignment21;
2
3 public class Replace {
4
5     public static void main(String[] args) {
6         String s1 = "My name is Rana Akshat";
7         System.out.println("String: "+s1);
8         String s2 = s1.replace(' ', '_');
9         System.out.println("Updated string: "+s2);
10    }
11 }
12
```

Console Output:  
<terminated> Replace [Java Application] D:\eclipse\plugins\org.eclipse.justi.openjdk hotspot\jre.full.win32.x86\_64\_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 6:07:12 p)  
String: My name is Rana Akshat  
Updated string: My\_name\_is\_Rana\_Akshat

16. Write a program to trim leading white space characters in a string.

The screenshot shows the Eclipse IDE with a project named "Assignment21". The editor displays a Java file named "trim.java" with the following code:

```
1 package Assignment21;
2
3 public class trim {
4
5     public static void main(String[] args) {
6         String s1 = " Hello ";
7         System.out.println(s1);
8         System.out.println(s1.trim());
9     }
10 }
11
```

The console output shows the execution of the program:

```
<terminated> trim [Java Application] D:\eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-093:
Hello
Hello
```

17. Write a program to trim trailing white space characters in a string.

The screenshot shows the Eclipse IDE with a project named "Assignment21". The editor displays a Java file named "trimTrailing.java" with the following code:

```
1 package Assignment21;
2
3 public class trimTrailing {
4
5     public static void main(String[] args) {
6         String s1 = " Hello ";
7         System.out.println(s1);
8         System.out.println(s1.trim());
9
10        String s2 = "   I am here   ";
11        System.out.println(s2);
12        System.out.println(s2.trim());
13
14    }
15 }
16
```

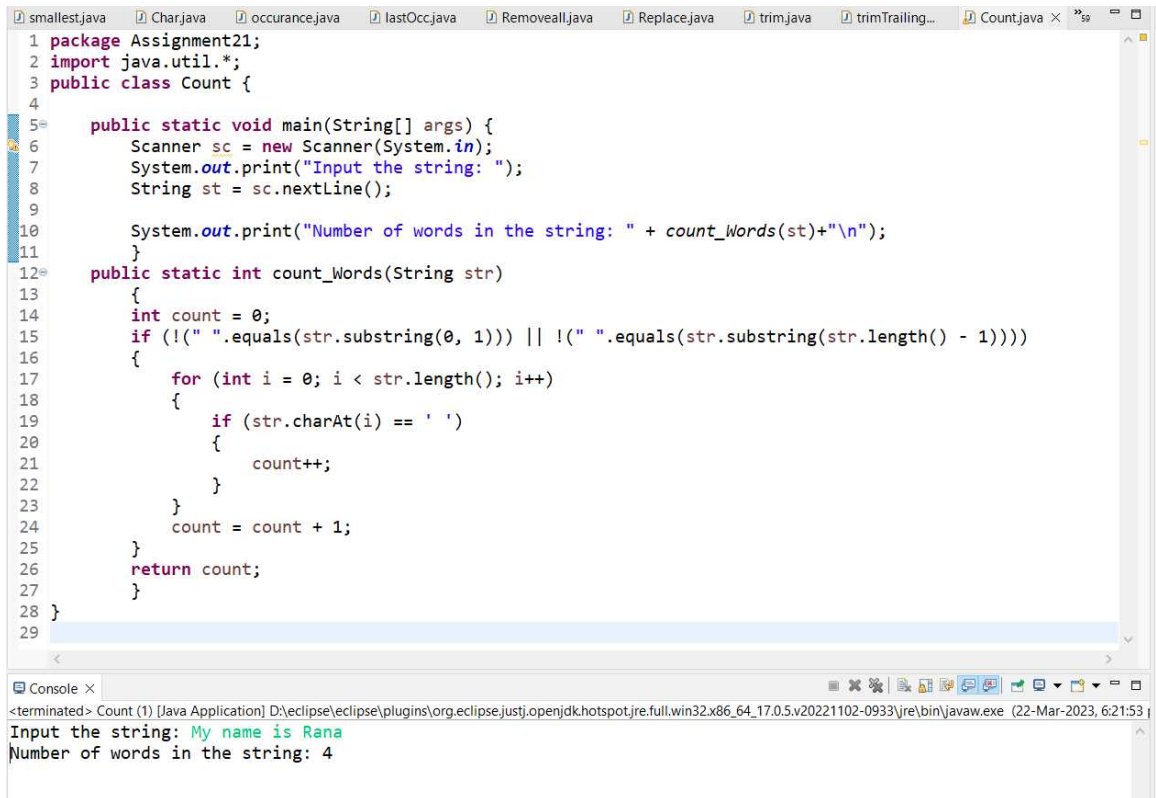
The console output shows the execution of the program:

```
<terminated> trimTrailing [Java Application] D:\eclipse\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-0933\jre\bin\javaw.exe:
Hello
Hello
   I am here
I am here
```

18. Write a program to print the word & count of words present in a given



sentence.



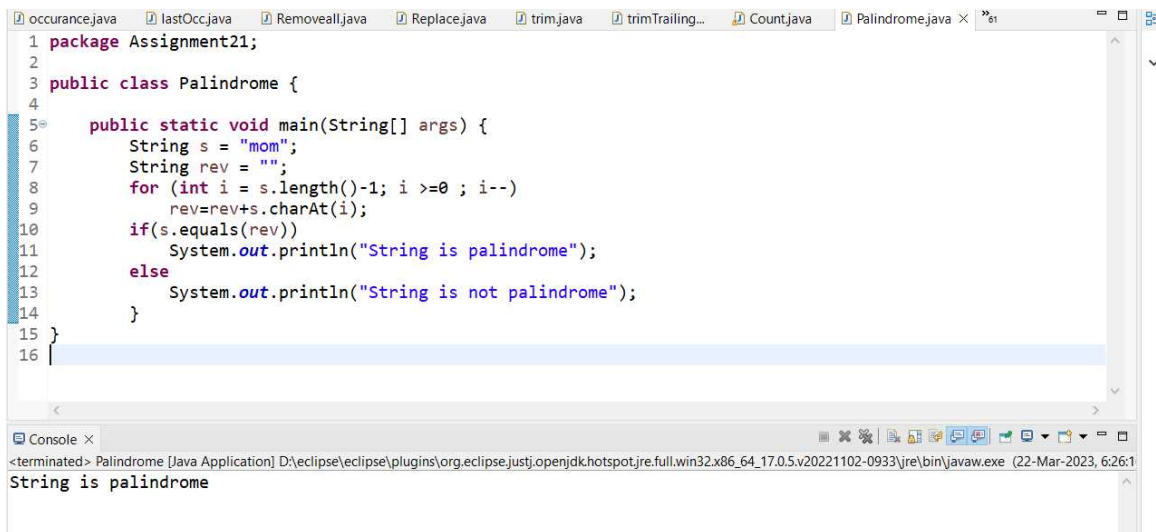
The screenshot shows an IDE with a Java file named `Count.java`. The code defines a `Count` class with a `main` method and a static `count_Words` method. The `main` method uses a `Scanner` to read a string from the user and prints the number of words in that string. The `count_Words` method counts the number of spaces in the string and adds 1 to get the word count. The console output shows the user input "My name is Rana" and the output "Number of words in the string: 4".

```
1 package Assignment21;
2 import java.util.*;
3 public class Count {
4
5     public static void main(String[] args) {
6         Scanner sc = new Scanner(System.in);
7         System.out.print("Input the string: ");
8         String st = sc.nextLine();
9
10        System.out.print("Number of words in the string: " + count_Words(st)+"\n");
11    }
12    public static int count_Words(String str)
13    {
14        int count = 0;
15        if (!(" ".equals(str.substring(0, 1))) || !(" ".equals(str.substring(str.length() - 1))))
16        {
17            for (int i = 0; i < str.length(); i++)
18            {
19                if (str.charAt(i) == ' ')
20                {
21                    count++;
22                }
23            }
24            count = count + 1;
25        }
26        return count;
27    }
28 }
29
```

Console Output:

```
<terminated> Count (1) [Java Application] D:\eclipse\ eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 6:21:53)
Input the string: My name is Rana
Number of words in the string: 4
```

19. Check whether a string entered by user is a palindrome string or not.



The screenshot shows an IDE with a Java file named `Palindrome.java`. The code defines a `Palindrome` class with a `main` method. The `main` method takes a string `s` and checks if it is a palindrome by comparing the string with its reverse. The console output shows the string "mom" and the output "String is palindrome".

```
1 package Assignment21;
2
3 public class Palindrome {
4
5     public static void main(String[] args) {
6         String s = "mom";
7         String rev = "";
8         for (int i = s.length()-1; i >=0 ; i--)
9             rev=rev+s.charAt(i);
10        if(s.equals(rev))
11            System.out.println("String is palindrome");
12        else
13            System.out.println("String is not palindrome");
14    }
15 }
16
```

Console Output:

```
<terminated> Palindrome [Java Application] D:\eclipse\ eclipse\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v20221102-0933\jre\bin\javaw.exe (22-Mar-2023, 6:26:1)
String is palindrome
```

20. What will happen if we do not override all abstract methods in subclass ?

Ans: A compile time error will be generated for each abstract method (that you don't override) saying "subclass\_name is not abstract and does not override abstract method abstractmethod\_name in classname".

21. What is the difference between Abstraction and Encapsulation ?

**Abstraction :**

Abstraction is hiding the details and implementation of the code.

Abstraction is a design level process.

Abstraction is concerned about what a class instance can do, instead of the implementation of the class.

**Encapsulation :**

Encapsulation is hiding the data and controlling the visibility of the code.

Encapsulation is an implementation level process.

Encapsulation helps in data binding and control over maintaining the transparency of the data.

22. Why abstract class has constructor even though you cannot create object ?

Ans: Abstract class has constructor even though we cannot create object because it's abstract and an object is concrete. An abstract class is sort of like a template, or an empty/partially empty structure, you have to extend it and build on it before you can use it. abstract class has a protected constructor (by default) allowing derived types to initialize it.

### 23. What is Out Of Memory Error in Exception Handling ?

Ans: Out Of Memory Error usually means that you're doing something wrong, either holding onto objects too long or trying to process too much data at a time. Sometimes, it indicates a problem that's out of your control, such as a third-party library that caches strings or an application server that doesn't clean up after deploys. And sometimes, it has nothing to do with objects on the heap.

### 24. What is the difference between throws and throw in java ?

#### **Throws :**

The throws keyword is used in the function signature. It is used when the function has some statements that can lead to exceptions.

The throws keyword can be used to declare multiple exceptions, separated by a comma. Whichever exception occurs, if matched with the declared ones, is thrown automatically then.

Syntax of throws keyword includes the class names of the Exceptions to be thrown. Syntax wise throws keyword is followed by exception class names.

throws keyword is used to propagate the checked Exceptions only.

#### **Throw :**

The throw keyword is used inside a function. It is used when it is required to throw an Exception logically.

The throw keyword is used to throw an exception explicitly. It can throw only one exception at a time.

Syntax of throw keyword includes the instance of the Exception to be thrown. Syntax wise throw keyword is followed by the instance variable.

throw keyword cannot propagate checked exceptions. It is only used to propagate the unchecked Exceptions that are not checked using the throws keyword.

25. Give some examples of Checked exceptions ?

Ans: Checked exceptions are the subclass of the Exception class. These types of exceptions need to be handled during the compile time of the program. These exceptions can be handled by the try-catch block or by using throws keyword otherwise the program will give a compilation error.

Examples are - IOException, SQLException, ClassNotFoundException, etc

26. Give some examples of Unchecked exceptions ?

Ans: Unchecked exceptions are not checked at compile time. In Java, exceptions under Error and RuntimeException classes are unchecked exceptions, everything else under throwable is checked.

Examples are - ArithmeticException, ClassCastException, NullPointerException, IllegalArgumentException, etc.

27. List the Methods in the Throwable class ?

Ans: Java Throwable class provides several methods like addSuppressed(), fillInStackTrace(), getMessage(), getStackTrace(), getSuppressed(), toString(), printStackTrace() etc.

28. What is Array Index Out Of Bounds Exception in java ?

Ans: The ArrayIndexOutOfBoundsException is a Runtime Exception thrown

only at runtime. The Java Compiler does not check for this error during the compilation of a program.

29. Will the code compile successfully? If yes, what will be the output ?

```
public abstract class A {  
    abstract void m1();  
}  
  
public class B extends A {  
    void m1(){  
        System.out.println("m1 in class B");  
    }  
}  
  
public class Test {  
    public static void main(String[] args) {  
        B b = new B();  
        b.m1();  
    }  
}
```

**Ans: Yes, Code will be compiled successfully**

**output :**

**m1 in class B**

31.Consider the below given code.

```
public abstract class A {  
    abstract void m1();  
    void m2(){  
        System.out.println("One");  
    }  
}
```

How to call m2() method in the above code ?

Ans: We will create a class B and we will extend the class A into Class B .Then we will create an object of class B and using "objectname.m2()", we can access the m2() function.