```
import java.util.Scanner;
import java.util.*; //import all the classes from the java.util package.

public class Main
{
	public static void main(String args[])
	{
		System.out.print(max);
	}
}
System.out.println()
for next output on nextline
System.out.println( "Required Minimum Sum is " + sumOfMinAbsDifferences(arr, n));


----------traverse array----------
int arr[] = {12, 13, 1, 10, 34, 10};

arr.length
arr[i]
----------traverse array----------

----------array passing----------

getMax(arr,n)

public int(int arr[] ,int n)

----------array passing----------

----------MATHEMATICAL----------

Math.max(arr[n-1], getmax(arr, n-1))
Math.min(arr[n-1], getmin(arr, n-1))
Math.abs(arr[i] - arr[j])
Math.ceil(5.3); -> 6.0
Math.floor(5.7); -> 5.0
Math.round(5.5); -> 6
Math.pow(2, 3);
Math.sqrt(16);
Math.cbrt(27);
Math.exp(2); // e^2
----------MATHEMATICAL----------

----------TERNARY----------
return (arr[i] > max) ? arr[i] : max;

----------TERNARY----------
```

----------FUNCTION INSIDE MAIN----------

```
import java.util.Scanner;

public class Main
{
  static int getmin(int arr[], int n){

  }
  static int getmax(int arr[], int n){

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

    int arr[] = {12, 13, 1, 10, 34, 10};
    int n = arr.length;

    System.out.println(getmin(arr, n));
    System.out.println(getmax(arr, n));
   }
}
```

----------FUNCTION INSIDE MAIN----------

----------MIN MAX----------

java provides a comprehensive set of mathematical functions through the Math class, which is part of the java.lang package.

```
import java.lang.Math;

int smallest = Integer.MAX_VALUE;
int largest  = Integer.MIN_VALUE;
```

----------MIN MAX----------

----------LIBRARY FUNCTION----------

```
import java.util.Arrays;

getmin(int arr[], int n){
   // sorting the array
       Arrays.sort(arr);
  }

int[] array = {1, 2, 3, 5, 8};
int index = Arrays.binarySearch(array, 3);
```

```
// index is 2

int[] array = new int[5];
Arrays.fill(array, 7);
// array is now {7, 7, 7, 7, 7}


import java.util.Arrays;

int[] original = {1, 2, 3};
int[] copy = Arrays.copyOf(original, 5);
// copy is now {1, 2, 3, 0, 0}

import java.util.Arrays;

int[] array1 = {1, 2, 3};
int[] array2 = {1, 2, 3};
boolean areEqual = Arrays.equals(array1, array2);
// areEqual is true

import java.util.Arrays;

int[] array = {1, 2, 3};
String arrayString = Arrays.toString(array);
// arrayString is "[1, 2, 3]"
```

The java.lang.System class provides array manipulation methods.

```
int[] source = {1, 2, 3, 4, 5};
int[] destination = new int[5];
System.arraycopy(source, 0, destination, 0, 5);
// destination is now {1, 2, 3, 4, 5}



import java.util.Arrays;
import java.util.List;

String[] array = {"a", "b", "c"};
List<String> list = Arrays.asList(array);
// list is ["a", "b", "c"]

List<String> list = Arrays.asList("a", "b", "c");
String[] array = list.toArray(new String[0]);
// array is {"a", "b", "c"}
```

4. Stream API

With Java 8, the Stream API provides a more functional approach to handling arrays.

Creating Streams from Arrays

```
import java.util.Arrays;
import java.util.stream.IntStream;

int[] array = {1, 2, 3, 4, 5};
IntStream stream = Arrays.stream(array);
stream.forEach(System.out::println);
// prints 1 2 3 4 5
```

Aggregating Operations

```
import java.util.Arrays;

int[] array = {1, 2, 3, 4, 5};
int sum = Arrays.stream(array).sum();
// sum is 15
```

These functions and methods cover most common operations on arrays in Java,
making it easier to manipulate and work with array data structures.

For Reversing

If you convert the array to a list, you can use the Collections.reverse() method.

```
import java.util.Arrays;
import java.util.Collections;
import java.util.List;

public class Main {
    public static void main(String[] args) {
        Integer[] array = {1, 2, 3, 4, 5};
        List<Integer> list = Arrays.asList(array);
        Collections.reverse(list);
        array = list.toArray(new Integer[0]);
        System.out.println(Arrays.toString(array)); // Output: [5, 4, 3, 2, 1]
    }
}
```

You can use the Stream API to reverse an array.

```
import java.util.Arrays;
import java.util.stream.IntStream;

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

```
        int[] array = {1, 2, 3, 4, 5};
        int[] reversedArray = IntStream.range(0, array.length)
                                       .map(i -> array[array.length - i - 1])
                                       .toArray();
        System.out.println(Arrays.toString(reversedArray)); // Output: [5, 4, 3, 2, 1]
    }
}
```

To reverse a string in Java, you can use several methods. Here are a few common approaches:

1. Using StringBuilder
Java's StringBuilder class has a built-in method reverse() that makes this task straightforward.

```
public class ReverseString {
    public static void main(String[] args) {
        String input = "Hello, World!";
        StringBuilder sb = new StringBuilder(input);
        String reversed = sb.reverse().toString();
        System.out.println("Reversed string: " + reversed);
    }
}
```

----------STRINGS AND CHARACTER ARRAY----------

STRING TO CHARACTER ARRAY

```
public class StringToCharArray {
    public static void main(String[] args) {
        String input = "Hello, World!";
        char[] charArray = input.toCharArray();

        // Print the character array
        for (char c : charArray) {
            System.out.print(c + " ");
        }
    }
}
```

ARRAY OF INTEGERS TO STRING

```
import java.util.Arrays;

public class IntArrayToString {
    public static void main(String[] args) {
        int[] intArray = {1, 2, 3, 4, 5};
        String str = Arrays.toString(intArray);

        // Remove brackets and commas
        str = str.replaceAll("[\\[\\],]", "");

        System.out.println("String representation: " + str);
    }
}
```

STRING OF NUMBERS TO AN ARRAY OF INTEGERS

```
public class StringToIntArray {
    public static void main(String[] args) {
        String str = "1 2 3 4 5";

        // Split the string by spaces
        String[] strArray = str.split(" ");

        // Convert each substring to an integer
        int[] intArray = new int[strArray.length];
        for (int i = 0; i < strArray.length; i++) {
            intArray[i] = Integer.parseInt(strArray[i]);
        }

        // Print the integer array
        System.out.print("Integer array: ");
        for (int i : intArray) {
            System.out.print(i + " ");
        }
    }
}
```

CHARACTER ARRAY TO STRING

```
public class CharArrayToString {
    public static void main(String[] args) {
        char[] charArray = {'H', 'e', 'l', 'l', 'o', ',', ' ', 'W', 'o', 'r', 'l', 'd', '!'};
        String str = new String(charArray);

        // Print the string
        System.out.println(str);
    }
}
```

----------STRINGS AND CHARACTER ARRAY----------

----------IF ELSE- SWITCH----------

```
for (int i = 0; i < str.length(); i++) {
        char x = str.charAt(i);
        if (x == '(' || x == '[' || x == '{') {
            stack.push(x);
            continue;
        }
        if (stack.isEmpty()) return false;
        char check;
        switch (x) {
            case ')':
                check = stack.pop();
                if (check == '{' || check == '[') return false;
                break;
            case '}':
                check = stack.pop();
                if (check == '(' || check == '[') return false;
                break;
            case ']':
                check = stack.pop();
                if (check == '(' || check == '{') return false;
                break;
        }
    }
```

----------IF ELSE- SWITCH----------

##########STRINGS##########

```
String str = "Hello";

char c = 'g';

if(c=='a'||c=='e'||c=='i'||c=='o'||c=='u'||c=='A'||c=='E'||c=='I'||c=='O'||c=='U')

str1.length()

str.charAt(n - 1)

String inputStr ="prepinsta";
inputStr.indexOf(i)
inputStr.lastIndexOf(i)
```

```
-------------------------passing string to function----------------

static int countVowels(String str, int n){

}

String str = "prepinsta";
countVowels(str, str.length())

---------------------------TRAVERESE----------------------

String str = "Hello, World!";
for (int i = 0; i < str.length(); i++) {
    char ch = str.charAt(i);
    // Do something with 'ch'
    System.out.println(ch);
}

String str = "Hello, World!";
for (char ch : str.toCharArray()) {
    // Do something with 'ch'
    System.out.println(ch);
}

------------------LIBRARY FUNCTIONS----------------------------------------

isLowercaseVowel(c)
isUppercaseVowel(c)

Character.isLowerCase(c)
Character.isUpperCase(c)

boolean isDigit1 = Character.isDigit(c1);
Character.isDigit(s.charAt(i)

char c = 'a';
char uppercaseC = Character.toUpperCase(c);
char uppercaseC = Character.toLowerCase(c)

String str = "   Hello, World!   ";
String trimmedStr = str.trim();
System.out.println("Original: '" + str + "'");
System.out.println("Trimmed: '" + trimmedStr + "'");

String str = "Hello";
int len = str.length();
boolean empty = str.isEmpty();
```

```
String str = "Hello, World!";
String substring1 = str.substring(7); // "World!"
String substring2 = str.substring(0, 5); // "Hello"

REPLACE@@
public class ReplaceAllExample {
    public static void main(String[] args) {
        String str = "Hello, World! Hello, Java!";
        String replacedStr = str.replaceAll("Hello", "Hi");
        System.out.println("Original String: " + str);
        System.out.println("Replaced String: " + replacedStr);
    }
}

Original String: Hello, World! Hello, Java!
Replaced String: Hi, World! Hi, Java!


static String remVowel(String str) {
        return str.replaceAll("[aeiouAEIOU]", "");

    }

 String s = "hel1456lo56wor%^ld";
     s=s.replaceAll("[^a-zA-Z]","");

import java.util.function.Function;
import java.util.regex.MatchResult;
import java.util.regex.Pattern;

public class ReplaceAllWithFunctionExample {
    public static void main(String[] args) {
        String str = "Today is 2023-07-12. Tomorrow is 2023-07-13.";

        // Using a regular expression to match date format
        String replacedStr = str.replaceAll("\\d{4}-\\d{2}-\\d{2}", new Function<MatchResult,
String>() {
            @Override
            public String apply(MatchResult match) {
                // Replace matched date with a generic date placeholder
                return "yyyy-mm-dd";
            }
        });

        System.out.println("Original String: " + str);
        System.out.println("Replaced String: " + replacedStr);
    }
}
```

Original String: Today is 2023-07-12. Tomorrow is 2023-07-13.
Replaced String: Today is yyyy-mm-dd. Tomorrow is yyyy-mm-dd.

More examples of replaceALL

String replaceAll(String regex, String replacement)

```
public class ReplaceAllExample {
    public static void main(String[] args) {
        String str = "Today is 2023-07-12. Tomorrow is 2023-07-13.";

        // Replace all digits with 'X'
        String replacedStr = str.replaceAll("\\d", "X");

        System.out.println("Original String: " + str);
        System.out.println("Replaced String: " + replacedStr);
    }
}
```

Original String: Today is 2023-07-12. Tomorrow is 2023-07-13.
Replaced String: Today is XXXX-XX-XX. Tomorrow is XXXX-XX-XX.

Regular Expression (regex): "\\d" matches any digit (0-9).
Replacement String: "X" is used to replace each digit found in the string.
Result: All occurrences of digits in the original string are replaced with "X".

splitting and joining@@

```
public class StringSplitJoinExample {
    public static void main(String[] args) {
        // Splitting a string into an array of substrings
        String str = "apple,orange,banana";
        String[] parts = str.split(",");

        System.out.println("Splitting:");
        for (String part : parts) {
            System.out.println(part);
        }

        // Joining an array of strings into a single string
        String joined = String.join("-", "2023", "07", "12");
        System.out.println("Joined: " + joined); // Output: "2023-07-12"
    }
}
```

Splitting:
apple
orange

banana
Joined: 2023-07-12

substring@@

String substring(int beginIndex)
String substring(int beginIndex, int endIndex)

begin index - [0, length()]

```
public class SubstringExample {
    public static void main(String[] args) {
        String str = "Hello, World!";

        // Extract substring starting from index 7 (inclusive) to the end
        String substr1 = str.substring(7);
        System.out.println("Substring 1: " + substr1); // Output: "World!"
    }
}
```

```
public class SubstringExample {
    public static void main(String[] args) {
        String str = "Hello, World!";

        // Extract substring from index 7 (inclusive) to index 12 (exclusive)
        String substr2 = str.substring(7, 12);
        System.out.println("Substring 2: " + substr2); // Output: "World"
    }
}
```

Stringbuilder@@
In Java, the StringBuilder class is commonly used for appending strings because strings in Java are immutable

```
public class StringBuilderExample {
    public static void main(String[] args) {
        StringBuilder sb = new StringBuilder();

        sb.append("Hello");
        sb.append(", ");
        sb.append("World!");

        System.out.println(sb.toString()); // Output: Hello, World!
    }
}
```

Appending in a Loop

```
public class StringBuilderLoopExample {
    public static void main(String[] args) {
        StringBuilder sb = new StringBuilder();

        String[] words = {"Java", "is", "fun"};
        for (String word : words) {
            sb.append(word).append(" ");
        }

        System.out.println(sb.toString().trim()); // Output: Java is fun
    }
}


public class StringBuilderAdvancedExample {
    public static void main(String[] args) {
        StringBuilder sb = new StringBuilder("Hello, World!");

        // Insert text at position 7
        sb.insert(7, "Beautiful ");
        System.out.println(sb.toString()); // Output: Hello, Beautiful World!

        // Delete characters from index 7 to 17
        sb.delete(7, 17);
        System.out.println(sb.toString()); // Output: Hello, World!

        // Reverse the string
        sb.reverse();
        System.out.println(sb.toString()); // Output: !dlroW ,olleH
    }
}
```

This method returns the numeric value that a character represents.

```
public class GetNumericValueExample {
    public static void main(String[] args) {
        char ch1 = '9';
        char ch2 = 'A';
            char ch3 = '$';

        int numValue1 = Character.getNumericValue(ch1);
        int numValue2 = Character.getNumericValue(ch2);
            int numValue3 = Character.getNumericValue(ch3);

        System.out.println("Numeric value of " + ch1 + " is " + numValue1); // Output: Numeric
value of 9 is 9
```

```
        System.out.println("Numeric value of " + ch2 + " is " + numValue2); // Output: Numeric value of A is -1
            System.out.println("Numeric value of " + ch3 + " is " + numValue3); // Output: Numeric value of $ is -1
    }
}
```