```
public class Example1 {

   public static void main(String[] args) {
      // Example usage:
      String inputTime = "4:10";
      String result = toAMPM(inputTime);
      System.out.println("Converted Time: " + result);
   }

   public static String toAMPM(String t24) {

         // Step 1: Find the position of the ':'
         int positionOfColon = t24.indexOf(':');

         // 1.1 Take everything before the ':' and store as hours
         int hours = Integer.parseInt(t24.substring(0, positionOfColon));

         // 1.2 Take everything after the ':' and store as minutes
         int minutes = Integer.parseInt(t24.substring(positionOfColon + 1));

         // Validate hours and minutes ranges
         if (hours < 0 || hours > 23 || minutes < 0 || minutes > 59) {
            throw new IllegalArgumentException("Invalid input format");
         }

         // Step 2: Change the value for hours and set the variable ampm
         String ampm;
         if (hours == 0) {
            // 2.1 If hours = 0, set hours to 12 and ampm to "AM"
            hours = 12;
            ampm = "AM";
         } else if (hours < 12) {
            // 2.2 If 0 < hours < 12, set ampm to "AM"
            ampm = "AM";
         } else if (hours == 12) {
            // 2.3 If hours = 12, set ampm to "PM"
            ampm = "PM";
         } else {
            // 2.4 If 12 < hours, subtract 12 from hours and set ampm to "PM"
            hours -= 12;
            ampm = "PM";
         }

         // Step 3: Construct the result string and return it
         // 3.1 If hours < 10, add a leading "0" to hours
         String formattedHours = "";
         if (hours < 10){
```

```
            formattedHours = "0" + hours;
        }
        else{
           formattedHours =  String.valueOf(hours);
        }
        // 3.2 Construct result as hours + ":" + minutes + space + ampm
        String result = formattedHours + ":" + String.format("%02d", minutes) + " " + ampm;
        // 3.3 Return result
        return result;


    }
}
```

```
public class Example2 {

   private static final String[] SMALL_NUMS = {"", "One", "Two", "Three", "Four", "Five", "Six",
"Seven", "Eight", "Nine", "Ten",
         "Eleven", "Twelve", "Thirteen", "Fourteen", "Fifteen", "Sixteen", "Seventeen", "Eighteen",
"Nineteen"};

   private static final String[] TENS = {"", "", "Twenty", "Thirty", "Forty", "Fifty", "Sixty", "Seventy",
"Eighty", "Ninety"};

   public static void main(String[] args){

      System.out.println(speakNumber(999999));

   }//end main

   public  static String speakNumber(int number){

      if (number <= 0 || number >= 1000000) {
         throw new IllegalArgumentException("Number must be between 1 and 999999");
      }

      int tens = number / 10;
      int units = number % 10;
      int hundreds = number / 100;
      int hundredsRemainder = number % 100;
      int thousands = number / 1000;
      int thousandsRemainder = number % 1000;

      if (number < 20) {
         return SMALL_NUMS[number];
      } else if (number < 100) {
         return TENS[tens] + ((units != 0) ? " " + SMALL_NUMS[units] : "");
      } else if (number < 1000) {
         return SMALL_NUMS[hundreds] + " Hundred" + ((hundredsRemainder == 0) ? " " : " "+
speakNumber(hundredsRemainder));
      } else {
         return speakNumber(thousands) + " Thousand" + ((thousandsRemainder== 0) ? " " : " "+
speakNumber(thousandsRemainder));
      }



   }
}//end class
```

```
public class Example3 {
    public static void main(String[] args) {
        String expression = "10 10 10"; // Example input
        int sum = calculateSum(expression);
        System.out.println("Sum: " + sum);
    }

    public static int calculateSum(String exp) {
        int sum = 0;
        int startPos = 0;
        int endPos = 0;

        while (endPos < exp.length()) {
            endPos = findEnd(exp, startPos);
            int number = Integer.parseInt(exp.substring(startPos, endPos));
            sum += number;
            startPos = endPos + 1;
        }

        return sum;
    }

    public static int findEnd(String exp, int searchPos) {
        while (searchPos < exp.length() && Character.isDigit(exp.charAt(searchPos))) {
            searchPos++;
        }
        return searchPos;
    }
}
```