```
package com.company;
import java.io.IOException;
import java.util.Scanner;

class Main {
    static Scanner sc = new Scanner(System.in);
    static LinkedList ob = new LinkedList();
    public static void main(String args []) throws IOException {
        while (1>0) {
            System.out.println("\n\n----Collection of logic Programs----");
            System.out.println("-------------------------------------");
            System.out.println("1.  Word extraction from given sentence\n" +
                    "2. Decimal to Binary conversion\n" +
                    "3. Palindrome or not(Integer)\n" +
                    "4. Sum of N integers\n" +
                    "5. Add digits of given number\n" +
                    "6. Pattern search\n" +
                    "7. Binary search\n" +
                    "8. Transpose of matrix\n" +
                    "9. Selection sort\n" +
                    "10. Bubble sort\n" +
                    "11. Insertion sort\n" +
                    "12. Pyramid With Star\n" +
                    "13. Permutation on letters\n" +
                    "14. Combination Values\n" +
                    "15. Linked List Operations\n" +
                    "16. Open Terminal\n" +
                    "17. Pattern occurence\n" +
                    "18. oddeventriangle");
            int choice = sc.nextInt();
            switch (choice) {
                case 1:
                    word_extraction();
                    break;
                case 2:
                    decimal_to_binary();
                    break;
                case 3:
                    palindrome();
                    break;
                case 4:
                    addition_of_n_integers();
                    break;
                case 5:
                    add_digits_of_numbers();
                    break;
                case 6:
                    pattern_search();
                    break;
                case 7:
                    binary_search();
                    break;
```

```
            case 8:
               transpose_of_matrix();
               break;
            case 9:
               selection_sort();
               break;
            case 10:
               bubble_sort();
               break;
            case 11:
               insertion_sort();
               break;
            case 12:
               pyramid_with_star();
               break;
            case 13:
               permutaion_on_letters();
               break;
            case 14:
               combination_values();
               break;
            case 15:
               ob.linked_list();
               break;
            case 16:
               open_terminal();
               break;
            case 17:
               occurence();
               break;
            case 18:
               oddeventriangle();
               break;
            default:
               System.out.println("\nSorry...! Wrong entry----");
         }
      }
   }
   static void word_extraction(){
      System.out.println("Enter the sentence you need to extract:");
      String str = sc.nextLine();
      str = sc.nextLine();
      String [] words = str.split(" ");
      for (String word : words){
         System.out.println(word);
      }
   }
   static void decimal_to_binary(){
      System.out.println("Enter Any Decimal Number :");
      int input_decimal_num = sc.nextInt();
      String binary_string = " ";
      while(input_decimal_num > 0){
```

```
        binary_string = input_decimal_num%2 + binary_string;
        input_decimal_num = input_decimal_num/2;
    }
    System.out.println("Conversion of decimal to binary is : " + binary_string);
}
static void palindrome(){
    int rev = 0, rem,actual_num,temp_num;
    System.out.println("Enter any number :");
    actual_num = sc.nextInt();
    temp_num = actual_num;

    while(temp_num > 0){
        rem = temp_num % 10;
        rev = rev*10+rem;
        temp_num = temp_num/10;
    }
    if(rev == actual_num)
        System.out.println("Given number is palindrome");
    else
        System.out.println("Given number is not palindrome");
}
static void addition_of_n_integers(){
    System.out.println("\nEnter the limit of numbers:");
    int number_limit = sc.nextInt();
    int sum = 0;
    for (int i = 0;i<=number_limit;i++){
        sum = sum+i;
    }
    System.out.println("\nSum of the Numbers up to given limit is: "+sum);
}
static void add_digits_of_numbers(){
    System.out.println("\nEnter the number:");
    int input_number = sc.nextInt();
    int sum = 0;
    while (input_number>0){
        int rem = input_number%10;
        sum = sum+rem;
        input_number = input_number/10;
    }
    System.out.println("\nSum of the digits is:"+sum);
}
static void pattern_search(){
    Scanner ob = new Scanner(System.in);
    int i, j=0, l=1, flag=0;
    String pattern, string;
    System.out.println("Enter the Pattern:");
    pattern = ob.nextLine().toLowerCase();
    System.out.println("Enter the String:");
    string = ob.nextLine().toLowerCase();

    try {
        for (i = 0; i < string.length(); i++) {
```

```
            if (pattern.charAt(j) == string.charAt(i)) {
                j++;
                l++;
            } else {
                l = 1;
                j = 0;
            }
            if (l == pattern.length()) {
                flag = 1;
                break;
            }
        }
    }catch (Exception e){}
    if (flag==1)
        System.out.println("Pattern Found");
    else
        System.out.println("Pattern not found");
}
static void binary_search(){
    int limit, key, mid, low=0, high, position = 0, i, flag = 0;
    System.out.println("Enter the limit of numbers:");
    limit = sc.nextInt();
    high = limit-1;
    System.out.println("Enter the Numbers up to given limit:");
    int numbers_array [] = new int[limit];
    for (i = 0; i<limit; i++){
        numbers_array[i] = sc.nextInt();
    }
    System.out.println("Enter the Key element to search:");
    key = sc.nextInt();

    mid = (low + high)/2;
    while (low < high){
        if (key  == numbers_array[mid]){
            position = mid + 1;
            flag = 1;
            break;
        }
        else if (key > numbers_array[mid]) {
            low = mid + 1;
            mid = (low + high) / 2;
        }else if (key < numbers_array[mid]){
            high = mid-1;
            mid = (low + high)/2;
        }
    }
    if (flag == 1)
        System.out.println("\nElement found at the position:"+position);
    else
        System.out.println("\nElement not found in the array");
}
static void transpose_of_matrix(){
```

```
    int rowd, columnd, i, j;
    System.out.println("Enter the dimension of matrix(row and column)::");
    rowd = sc.nextInt();
    columnd = sc.nextInt();
    int matrix_array [][] = new int[rowd][columnd], matrix_t_array [][] = new int[columnd][rowd];
    System.out.println("Enter the values for the matrix in row wise::");
    for (i =0; i<rowd; i++){
        for (j=0; j<columnd;j++){
            matrix_array[i][j] = sc.nextInt();
        }
    }

    for (i=0;i<columnd;i++){
        for (j=0;j<rowd;j++){
            matrix_t_array[i][j] = matrix_array[j][i];
        }
    }
    System.out.println("Transpose of the given matrix is::");
    for (i =0; i<columnd; i++){
        for (j=0; j<rowd;j++){
            System.out.print(matrix_t_array[i][j]);
            System.out.print("\t");
        }
        System.out.println("");
    }
}
static void selection_sort(){
    int limit, l=0, min, i, j, mi=0, temp;
    System.out.println("Enter the limit of numbers to sort::");
    limit = sc.nextInt();
    int array[] = new int[limit];
    System.out.println("Enter the numbers upto given limit::");
    for (i =0;i<limit;i++){
        array[i] = sc.nextInt();
    }
    System.out.println("sorted list is::");
    while (l<limit) {
        min = array[l];
        j = l + 1;
        while (j < limit) {
            if (min > array[j]) {
                mi = j;
                min = array[j];
                temp = array[l];
                array[l] = min;
                array[mi] = temp;
            }
            j++;
        }
        l++;
    }
    for (i =0;i<limit;i++){
```

```
        System.out.println(array[i]);
    }
}
static void bubble_sort(){
    int limit, i, j, temp;
    System.out.println("Enter the limit of numbers to sort::");
    limit = sc.nextInt();
    int array[] = new int[limit];
    System.out.println("Enter the numbers upto given limit::");
    for (i =0;i<limit;i++){
        array[i] = sc.nextInt();
    }
    System.out.println("sorted list is::");
    for (i =0;i<limit-1;i++){
        for (j=0;j<limit-i-1;j++){
            if (array[j]>array[j+1]){
                temp = array[j];
                array[j] = array[j+1];
                array[j+1] = temp;
            }
        }
    }
    for (i =0;i<limit;i++){
        System.out.println(array[i]);
    }
}
static void insertion_sort(){
    int limit, i, j, temp;
    System.out.println("Enter the limit of numbers to sort::");
    limit = sc.nextInt();
    int array[] = new int[limit];
    System.out.println("Enter the numbers upto given limit::");
    for (i =0;i<limit;i++){
        array[i] = sc.nextInt();
    }
    System.out.println("sorted list is::");

}
static void pyramid_with_star(){
    int limit, k = 0, j;
    System.out.println("Enter the limit of raws::");
    limit = sc.nextInt();
    for (int i = 1; i <= limit; i++) {
        if (i % 2 != 0) {
            for (j = k + 1; j < k + i; j++)
                System.out.print(j + "*");
            System.out.println(j++);
            k = j;
        } else {
            k = k + i - 1;
            for (j = k; j > k - i + 1; j--)
                System.out.print(j + "*");
```

```
            System.out.println(j);
            }
        }
    }
    static void permutaion_on_letters() {
        // Create an alphabet to work with
        String alphabet;
        System.out.println("Enter the length::");
        int length = sc.nextInt();
        alphabet =
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890!@#$%^&*()-=_+";
        // Find all possible combinations of this alphabet in the string size of 3
        possibleStrings(length,alphabet,"");
    }
    static void possibleStrings(int length,String alphabet, String curr) {
        // If the current string has reached it's maximum length
        if(curr.length() == length) {
            System.out.println(curr);
            // Else add each letter from the alphabet to new strings and process these new strings again
        } else {
            for(int i = 0; i < alphabet.length(); i++) {
                String oldCurr = curr;
                curr += alphabet.charAt(i);
                possibleStrings(length,alphabet,curr);
                curr = oldCurr;
            }
        }
    }
    static void combination_values(){
        int a, b, c, x, y, z, d, sum, i, j, k, id=0;
        System.out.println("Enter the values for a,b,c,d of (ax+by+cz=d)::");
        a = sc.nextInt();
        b = sc.nextInt();
        c = sc.nextInt();
        d = sc.nextInt();
        System.out.println("Combination values are");
        System.out.println("----------------------");
        for (i =0; i<=d; i++){
            x = i;
            for (j =0; j<=d; j++) {
                y = j;
                for (k = 0; k <= d; k++) {
                    z = k;
                    sum = a * x + b * y + c * z;
                    if (sum == d) {
                        id++;
                        System.out.println(id+":"+x+","+y+","+z);
                    }
                }
            }
        }
    }
```

```
static void open_terminal() throws IOException {
    String command= "/usr/bin/gnome-terminal";
    Runtime rt = Runtime.getRuntime();
    Process pr = rt.exec(command);
}
static void occurence(){
    Scanner ob = new Scanner(System.in);
    int i, j=0, l=1, c=0;
    String pattern, string;
    System.out.println("Enter the Pattern:");
    pattern = ob.nextLine().toLowerCase();
    System.out.println("Enter the String:");
    string = ob.nextLine().toLowerCase();

    try {
        for (i = 0; i < string.length(); i++) {
            if (pattern.charAt(j) == string.charAt(i)) {
                j++;
                l++;
                if(l == pattern.length()){
                    c++;
                    l = 1;
                    j = 0;
                }
            } else {
                l = 1;
                j = 0;
            }
        }
    }catch (Exception e){}
    if (c==0)
        System.out.println("Pattern not Found");
    else
        System.out.println("Pattern occurence :: "+c);
}
static void oddeventriangle(){
    int row_limit, temp = 1, l, new_v;
    System.out.println("Enter the number of Rows::");
    row_limit = sc.nextInt();
    for (int i = 1; i <= row_limit;i++){
        if(i%2==0){
            l = 0;
            while (l<i) {
                temp++;
                l++;
            }
            new_v = temp;
            for (int j = 0;j < i;j++){
                System.out.print(temp);
                temp--;
                if(j == i-1)
                    break;
```

```
                System.out.print("*");

            }
            temp = new_v;
        }else {
            if (i == 1) {
                System.out.print(i);
            }else {
                temp++;
                for (int k = 0;k < i;k++){
                    System.out.print(temp);
                    temp++;
                    if(k == i-1){
                        temp--;
                        break;
                    }
                    System.out.print("*");
                }
            }
        }
        System.out.println("");
    }
  }
}
```