1) 
```
import java.util.*;
public class Main
{
  public static void main(String[] args) {

  System.out.println("Enter the number n");

     Scanner sc= new Scanner(System.in);
int n = sc.nextInt();

for(int i=0;i<n;i++)
  {
     for(int j=0;j<n;j++)
     {
        if(i==0 || i == n-1 || j == 0 || j == n-1) // Condition for outermost character *
        System.out.print("*");

        else if((i+j)%2==0) // Condition for '+'
        System.out.print("+");

        else // Else space(' ') will be printed
        System.out.print(" ");
     }
     System.out.println(); // Breaking the line after 1 line of characters.
  }
  }
}
```

2)

```
// Import these classes from utility package
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import java.util.Scanner;

// PatternMatcher class definition
public class PatternMatch {
   // Main method
   public static void main(String[] args) {
      Scanner input = new Scanner(System.in);
      // Reading pattern from the user
      System.out.print("Enter a pattern string: ");
      String stringPattern = input.nextLine();
      // Loop is executed util user enters exit
      while(true) {
         // Reading a string
         System.out.print("Enter a string: ");
         String string = input.nextLine();

         // If user enters exit then print bye and then break the loop
         if (string.equals("exit")) {
            System.out.println("Bye");
            break;
         }
         // Creating patter object using patter
```

```
        Pattern pattern = Pattern.compile(stringPattern, Pattern.CASE_INSENSITIVE);
        // Creating matcher using patter object and input string
        Matcher matcher = pattern.matcher(string);

        // If match found in the input string
        boolean isFound = matcher.find();
        if(isFound) {
           System.out.println(stringPattern + " occurs in " + "\""+string+"\"\n");
        }
        // If match not found in the input string
        else {
           System.out.println(stringPattern + " does NOT occur in " + "\""+string+"\"\n");
        }

      }
   }

}
```

3)

```
/**
 * Personal Data.
 */
   import java.util.Date;
public class PersonalData {
   // Create variables.
   private java.util.Date birthDate;
   private String address;
   private long ssn;

   // Constructor 1.
   public PersonalData(java.util.Date birthDate,long ssn){
      this.birthDate  = birthDate;
      this.ssn        = ssn;
   }

   // Constructor 2.
   public PersonalData(int year,int month,int day ,long ssn){
      this.birthDate  = new Date(year,month,day);
      this.ssn        = ssn;
   }

   // Get birtdate.
   public Date getBirthDate(){
      return this.birthDate;
   }

   // Get address
   public String getAddress(){
      return this.address;
   }

   // Get SSN.
   public long getSSN(){
      return this.ssn;
   }

   // Set address.
   public void setAddress(String address){
```

```
        this.address    = address;
    }
}


/**
 * Student.
 */
public class Student{
    // Create variables.
    private String name;
    private long id;
    private double gpa;
    private PersonalData pd;

    // Constructor 1.
    public Student(String name, long id, double gpa, PersonalData pd){
        this.name   = name;
        this.id     = id;
        this.gpa    = gpa;
        this.pd     = pd ;
    }

    // Get student name.
    public String getName(){
        return this.name;
    }

    // Get student ID.
    public long getID(){
        return this.id;
    }

    // Get student GPA.
    public double getGPA(){
        return this.gpa;
    }

    // Get student personal data.
    public PersonalData getPersonalData(){
        return this.pd;
    }

    // Student information tostring.
    public String toString(){
        return "Name: " + this.name + " ID: " + this.id + " GPA: " + this.gpa;
    }
}




/**
 * Course.
 */
public class Course{
    // Create variables and initialize.
    private String name;
    private int capacity = 40;
    private Student[] students= new Student[capacity];
    private int numberOfStudents;
```

```
// Constructor 1.
public Course(String name){
    this.name = name;
}

// Constructor 2.
public Course(String name, int capacity){
    this.name       = name;
    this.capacity   = capacity;
}

// Get student number.
public int getNumberOfStudents(){
    return this.numberOfStudents;
}

// Get course name.
public String getCourseName(){
    return this.name;
}

// Get student.
public Student[] getStudents(){
    return this.students;
}

// Student addition to course.
public boolean addStudent(Student student){
    if(numberOfStudents < capacity){
        for(int i = 0; i < numberOfStudents; i++){
            if(student.equals(students[i]))
            return false;
        }
        students[numberOfStudents] = student;
        numberOfStudents++;
    return true;
    }
    return false;
}

// Student drop from course.
public boolean dropStudent(Student student){
    for(int i = 0; i < numberOfStudents; i++){
        if(student.equals(students[i])){
            students[i] = null;
            while(i < numberOfStudents){
                students[i] = students[i+1];
                i++;
            }
            numberOfStudents--;
            return true;
        }
    }
return false;
}

// Increase capacity of course.
public void increaseCapacity(){
    capacity = capacity + 5;
```

```
    }

    // Get best student.
    public Student getBestStudent(){
        Student beststudent = students[0];
        for(int i=1; i < numberOfStudents; i++){
            if(students[i].getGPA() > students[i-1].getGPA())
                beststudent = students[i];
        }
        return beststudent;
    }

    // Get youngest student.
    public Student getYoungestStudent(){
        Student youngestStudent = students[0];
        for(int i = 0; i < numberOfStudents - 1; i++){
            if((students[i].getPersonalData().getBirthDate()).compareTo(students[i+1].getPersonalData().getBirthDat
e()) < 0)
                youngestStudent = students[i];
            else
if(students[i].getPersonalData().getBirthDate().compareTo(students[i+1].getPersonalData().getBirthDate()) > 0)
                youngestStudent = students[i + 1];
        }
        return youngestStudent;
    }

    // Clear course.
    public void clear(){
        for(int i = 0; i < numberOfStudents; i++){
            students[i] = null;
        }
    }

    // Student list.
    public void list(){
        String result = "";
        for(int i = 0; i < numberOfStudents; i++){
            result += students[i] + "\n";
        }
        System.out.println(result);
    }

    // Course infrmation to string.
    public String toString(){
        return "Number of students " + this.numberOfStudents + "\ncapacity " + this.capacity + "\ncourse name " +
this.name;
    }
}




/**
 * Personal Data.
 */
    import java.util.Date;
public class PersonalData {
    // Create variables.
    private java.util.Date birthDate;
    private String address;
    private long ssn;
```

```
    // Constructor 1.
    public PersonalData(java.util.Date birthDate,long ssn){
        this.birthDate  = birthDate;
        this.ssn        = ssn;
    }

    // Constructor 2.
    public PersonalData(int year,int month,int day ,long ssn){
        this.birthDate  = new Date(year,month,day);
        this.ssn        = ssn;
    }

    // Get birtdate.
    public Date getBirthDate(){
        return this.birthDate;
    }

    // Get address
    public String getAddress(){
        return this.address;
    }

    // Get SSN.
    public long getSSN(){
        return this.ssn;
    }

    // Set address.
    public void setAddress(String address){
        this.address    = address;
    }
}
```

```
/**
 * Student.
 */
public class Student{
    // Create variables.
    private String name;
    private long id;
    private double gpa;
    private PersonalData pd;

    // Constructor 1.
    public Student(String name, long id, double gpa, PersonalData pd){
        this.name   = name;
        this.id     = id;
        this.gpa    = gpa;
        this.pd     = pd ;
    }

    // Get student name.
    public String getName(){
        return this.name;
    }
```

```
    // Get student ID.
    public long getID(){
        return this.id;
    }

    // Get student GPA.
    public double getGPA(){
        return this.gpa;
    }

    // Get student personal data.
    public PersonalData getPersonalData(){
        return this.pd;
    }

    // Student information tostring.
    public String toString(){
        return "Name: " + this.name + " ID: " + this.id + " GPA: " + this.gpa;
    }
}


/**
 * Course.
 */
public class Course{
    // Create variables and initialize.
    private String name;
    private int capacity = 40;
    private Student[] students= new Student[capacity];
    private int numberOfStudents;

    // Constructor 1.
    public Course(String name){
        this.name = name;
    }

    // Constructor 2.
    public Course(String name, int capacity){
        this.name       = name;
        this.capacity   = capacity;
    }

    // Get student number.
    public int getNumberOfStudents(){
        return this.numberOfStudents;
    }

    // Get course name.
    public String getCourseName(){
        return this.name;
    }

    // Get student.
    public Student[] getStudents(){
        return this.students;
    }

    // Student addition to course.
    public boolean addStudent(Student student){
```

```
        if(numberOfStudents < capacity){
            for(int i = 0; i < numberOfStudents; i++){
                if(student.equals(students[i]))
                return false;
            }
            students[numberOfStudents] = student;
            numberOfStudents++;
        return true;
        }
        return false;
    }

    // Student drop from course.
    public boolean dropStudent(Student student){
        for(int i = 0; i < numberOfStudents; i++){
            if(student.equals(students[i])){
                students[i] = null;
                while(i < numberOfStudents){
                    students[i] = students[i+1];
                    i++;
                }
                numberOfStudents--;
                return true;
            }
        }
    return false;
    }

    // Increase capacity of course.
    public void increaseCapacity(){
        capacity = capacity + 5;
    }

    // Get best student.
    public Student getBestStudent(){
        Student beststudent = students[0];
        for(int i=1; i < numberOfStudents; i++){
            if(students[i].getGPA() > students[i-1].getGPA())
                beststudent = students[i];
        }
        return beststudent;
    }

    // Get youngest student.
    public Student getYoungestStudent(){
        Student youngestStudent = students[0];
        for(int i = 0; i < numberOfStudents - 1; i++){
            if((students[i].getPersonalData().getBirthDate()).compareTo(students[i+1].getPersonalData().getBirthDat
e()) < 0)
                youngestStudent = students[i];
            else
if(students[i].getPersonalData().getBirthDate().compareTo(students[i+1].getPersonalData().getBirthDate()) > 0)
                youngestStudent = students[i + 1];
        }
        return youngestStudent;
    }

    // Clear course.
    public void clear(){
        for(int i = 0; i < numberOfStudents; i++){
```

```
            students[i] = null;
        }
    }

    // Student list.
    public void list(){
        String result = "";
        for(int i = 0; i < numberOfStudents; i++){
            result += students[i] + "\n";
        }
        System.out.println(result);
    }

    // Course infrmation to string.
    public String toString(){
        return "Number of students " + this.numberOfStudents + "\ncapacity " + this.capacity + "\ncourse name " +
this.name;
    }
}




/**
 * Test.
 */
public class StudentTest{
    public static void main(String[] args){

        // 5 students are created.
        Student student1 = new Student("Rodney McKay ",5005,3.90,new PersonalData(85,5,5,115));
        Student student2 = new Student("Daniel Jackson ",5006,2.80,new PersonalData(86,2,4,345));
        Student student3 = new Student("Samantha Carter ",5007,3.30,new PersonalData(87,7,6,123));
        Student student4 = new Student("George Hammond ",5008,2.20,new PersonalData(90,4,12,657));
        Student student5 = new Student("Jack O'Neill ",5009,2.50,new PersonalData(92,6,12,854));

        // Course CSE141 is created with a capacity of 3.
        Course course1 = new Course("CSE141",3);

        // Any 4 of the students is added to CSE141.
        course1.addStudent(student1);
        course1.addStudent(student2);
        course1.addStudent(student3);
        course1.addStudent(student4);

        // All students of CSE141 are printed on the screen.
        System.out.println("All students of course " + course1.getCourseName() + ": ");
        course1.list();

        // The capacity of CSE141 is increased.
        course1.increaseCapacity();

        // Remaining 2 students are added to CSE141.
        course1.addStudent(student4);
        course1.addStudent(student5);

        // All students of CSE141 are printed on the screen.
        System.out.println("All students of course "+course1.getCourseName() + ": ");
        course1.list();

        // Student with ID 5005 is dropped from CSE141.
```

```
        course1.dropStudent(student1);

        // All students of CSE141 are printed.
        System.out.println("All students of course "+ course1.getCourseName() + ": ");
        course1.list();

        // Number of students enrolled to CSE141 is printed.
        System.out.println(course1.getCourseName() + "'s number of students are " +
course1.getNumberOfStudents() + ".");

        // CSE141's best student's birthdate's year is printed.
        System.out.println("\nBirth year of the best student of CSE141 is : "
+course1.getBestStudent().getPersonalData().getBirthDate().getYear());

        // New course is created.
        Course course2 = new Course("CSE142");

        // All students enrolled in CSE141 are added to CSE142.
        Student[] students = course1.getStudents();
        for(int i=0; i < course1.getNumberOfStudents(); i++)
           course2.addStudent(students[i]);

        // All students of CSE141 are removed from the course.
        course1.clear();

        // Student with ID 5005 is dropped from CSE141 and result of the operation is printed on the screen.
        System.out.println("\nStudent with ID 5005 is dropped from " + course1.getCourseName() + " is " +
course1.dropStudent(student1));

        // All students of CSE142 are printed on the screen.
        System.out.println("\nAll students of "+course2.getCourseName() + ": ");
        course2.list();

        // Best studen of CSE142 is dropped from CSE142.
        course2.dropStudent(course2.getBestStudent());

        // All students of CSE142 are printed on the screen.
        System.out.println("All students of " + course2.getCourseName() + ": ");
        course2.list();

        // GPA of youngest student CSE142 is printed on the screen.
        System.out.println("The GPA of youngest student CSE142 is " + course2.getYoungestStudent().getGPA());

        // Courses CSE141 and CSE142 are printed on the screen.
        System.out.println("\nAll students of " + course1.getCourseName() + ": ");
        course1.list();
        System.out.println("\nAll students of " + course2.getCourseName() + ": ");
        course2.list();
    } // End of main method.
}
```