Ans5) public class SpecialMatrixStorage {

**//(a) Diagonal Matrix: Only n elements (arr[i][i])**

static class DiagonalMatrix {

int n;

int[] arr;

DiagonalMatrix(int n) {

this.n = n;

arr = new int[n];

}

void set(int i, int j, int val) {

if (i == j) arr[i] = val;

}

int get(int i, int j) {

return (i == j) ? arr[i] : 0;

}

}

**// (b) Tri-diagonal Matrix: Only 3n-2 elements**

static class TriDiagonalMatrix {

int n;

int[] arr;

TriDiagonalMatrix(int n) {

this.n = n;

arr = new int[3 \* n - 2];

}

void set(int i, int j, int val) {

if (Math.abs(i - j) > 1) return;

if (i - j == 1) arr[i - 1] = val; // lower diagonal

else if (i == j) arr[n - 1 + i] = val; // main diagonal

else arr[2 \* n - 1 + i] = val; // upper diagonal

}

int get(int i, int j) {

if (Math.abs(i - j) > 1) return 0;

if (i - j == 1) return arr[i - 1];

else if (i == j) return arr[n - 1 + i];

else return arr[2 \* n - 1 + i];

}

}

**// (c) Lower Triangular Matrix: n(n+1)/2 elements**

static class LowerTriangularMatrix {

int n;

int[] arr;

LowerTriangularMatrix(int n) {

this.n = n;

arr = new int[n \* (n + 1) / 2];

}

void set(int i, int j, int val) {

if (i >= j) arr[i \* (i + 1) / 2 + j] = val;

}

int get(int i, int j) {

return (i >= j) ? arr[i \* (i + 1) / 2 + j] : 0;

}

}

**// (d) Upper Triangular Matrix: n(n+1)/2 elements**

static class UpperTriangularMatrix {

int n;

int[] arr;

UpperTriangularMatrix(int n) {

this.n = n;

arr = new int[n \* (n + 1) / 2];

}

void set(int i, int j, int val) {

if (i <= j) arr[i \* n - (i \* (i - 1)) / 2 + (j - i)] = val;

}

int get(int i, int j) {

return (i <= j) ? arr[i \* n - (i \* (i - 1)) / 2 + (j - i)] : 0;

}

}

**// (e) Symmetric Matrix: store either lower or upper triangle**

static class SymmetricMatrix {

int n;

int[] arr;

SymmetricMatrix(int n) {

this.n = n;

arr = new int[n \* (n + 1) / 2];

}

void set(int i, int j, int val) {

if (i >= j) arr[i \* (i + 1) / 2 + j] = val;

else arr[j \* (j + 1) / 2 + i] = val;

}

int get(int i, int j) {

return (i >= j) ? arr[i \* (i + 1) / 2 + j] : arr[j \* (j + 1) / 2 + i];

}

}