Prime istruzioni

1. Leggere e valutare un polinomio di secondo grado

```
//Polinomio di secondo grado
import fondamenti.corejava.*;
public class Polinomio2{
      public static void main (String args[])
      {
            System.out.println ("Inserisci i coefficienti interi del polinomio
di secondo grado");
            int a = Console.readInt("a =");
            int b = Console.readInt("b =");
            int c = Console.readInt("c =");
            System.out.println ();
            System.out.println ("p(x) = "+a+"x^2 + "+b+"x + "+c);
            System.out.println ();
            double x = Console.readDouble("x =");
            double valore = a*x*x + b*x + c;
            System.out.println ("p("+x+") = "+valore);
      }
}
```

2. Identificazione dell'ora del giorno a partire da un numero di secondi letti dalla mezzanotte.

```
   /*
   Legge un numero di secondi a partire dalla mezzanotte e converte i secondi
   letti in ore, minuti e secondi - Versione con divisione e sottrazione
   */
public class OrarioA{
      public static void main(String args[]){
            final int orePerGiorno=24; //numero di ore in un giorno
            final int minutiPerOra=60; //numero di minuti in un'ora
            final int secondiPerOra=minutiPerOra*60;
                              //60*60=3600 numero di secondi in un'ora
            int secondiLetti=Console.readInt("Inserire i secondi dalla
mezzanotte: ");
            int ore=secondiLetti/secondiPerOra;
            int minuti=(secondiLetti-(ore*secondiPerOra))/minutiPerOra;
            int secondi=(secondiLetti-(ore*secondiPerOra)-
(minuti*minutiPerOra));
            System.out.println("L'orario e' "+ore+":"+minuti+":"+secondi);
      }
}

      /*
      Legge un numero di secondi a partire dalla mezzanotte e converte i secondi
      letti in ore, minuti e secondi - Versione con divisione e modulo;
      */
public class OrarioB{
public static void main(String args[]){
      final int orePerGiorno=24; //numero di ore in un giorno
      final int minutiPerOra=60; //numero di minuti in un'ora
      final int secondiPerOra=minutiPerOra*60;
                        //60*60=3600 numero di secondi in un'ora
      int secondi=Console.readInt("Inserire i secondi dalla mezzanotte: ");
      int ore=secondi/secondiPerOra;
      secondi=secondi%secondiPerOra;
      int minuti=secondi/minutiPerOra;
      secondi=secondi%minutiPerOra;
```

```
      System.out.println("L'orario e' "+ore+":"+minuti+":"+secondi);
      }
}
```

3. Leggere cinque numeri e calcolare la media e la somma.

```
// Versione con 6 variabili.
public class MediaA{
      public static void main(String args[]){
            double n1,n2,n3,n4,n5;
            n1=Console.readDouble("N1 :");
            n2=Console.readDouble("N2 :");
            n3=Console.readDouble("N3 :");
            n4=Console.readDouble("N4 :");
            n5=Console.readDouble("N5 :");
            double somma=n1+n2+n3+n4+n5;
            System.out.println("La somma e' "+somma+", la media e' "+somma/5);
      }
}

//Versione con una variabile
public class MediaB{
      public static void main(String args[]){
            double somma;
            somma=0;
            somma+=Console.readDouble("N1 :");
            somma+=Console.readDouble("N2 :");
            somma+=Console.readDouble("N3 :");
            somma+=Console.readDouble("N4 :");
            somma+=Console.readDouble("N5 :");
            System.out.println("La somma e' "+somma+", la media e' "+somma/5);
      }
}
```

4. Calcolo delle soluzioni di una equazione di secondo grado

```
public class Equazione2{
      public static void main (String args[]){
            int a=Console.readInt("Inserire il valore di a");
            int b=Console.readInt("Inserire il valore di b");
            int c=Console.readInt("Inserire il valore di c");
            int delta=b*b-4*a*c;
            System.out.println("Soluzioni dell'equazione "+a+"x^2 +"+b+"x +"+c+
      " = 0");
            if (delta<0)
                  System.out.println("Non esistono soluzioni reali.");
            else if (delta==0){
                  double x=-b/(2.0*a);
                  System.out.println("Soluzioni coincidenti x1=x2="+x);
                  }
            else{
                  double x1=(-b+Math.sqrt(delta))/(2*a);
                  double x2=(-b-Math.sqrt(delta))/(2*a);
                  System.out.println("Soluzioni reali e distinte x1="+x1+
      " x2="+x2);
                  }
            }
}
```

5. Calcolare il massimo di tre numeri

```
public class MaxDi3NumeriA {
public static void main(String[] args) {
```

```
	int a, b, c;
	a=Console.readInt("Inserisci un intero a> ");
	b=Console.readInt("Inserisci un intero b> ");
	c=Console.readInt("Inserisci un intero c> ");
	if (a>b)
		if (a>c)
			System.out.println("il massimo e' a="+a);
		else
			System.out.println("il massimo e' c="+c);
	else
		if (b>c)
			System.out.println("il massimo e' b="+b);
		else
			System.out.println("il massimo e' c="+c);
	}
}

// calcola il massimo di tre numeri

public class MaxDi3NumeriB {
public static void main(String[] args) {
	int a, b, c;
	a=Console.readInt("Inserisci un intero a> ");
	b=Console.readInt("Inserisci un intero b> ");
	c=Console.readInt("Inserisci un intero c> ");

	int max=a;  // supponiamo che il massimo sia a
	if (b>max) max=b; // aggiorniamo il massimo
	if (c>max) max=c; // aggiorniamo il massimo

	System.out.println("Il massimo tra "+a+" ; "+b+" e "+c+" e' ="+max);
	}
}

// calcola il massimo di tre numeri
//             Versione con condizioni composte.
public class MaxDi3NumeriC {
public static void main(String[] args) {
	int a, b, c;
	a=Console.readInt("Inserisci un intero a> ");
	b=Console.readInt("Inserisci un intero b> ");
	c=Console.readInt("Inserisci un intero c> ");
	if (a>b && a>c)
			System.out.println("Il massimo e' a="+a);
	else
	   if (b>a && b>c)
			System.out.println("Il massimo e' b="+b);
		else
			System.out.println("Il massimo e' c="+c);
	}
}
```

6. Classificare un triangolo date le misure dei suoi lati

```
/*
Legge tre numeri interi che rappresentano i lati di un triangolo e stampa un
messaggio che dice se il triangolo è equilatero, isoscele oppure scaleno.
Versione con if innestati.
*/
public class TriangoloA {
	public static void main(String[] args) {
		// legge il valore dei tre lati
		System.out.println("Scrivi la lunghezza dei lati di un triangolo ");
		int a = Console.readInt("a=");
```

```
            int b = Console.readInt("b=");
            int c = Console.readInt("c=");
            System.out.println();
            System.out.print("Il triangolo e' ");
            // confronta i lati e stampa il messaggio
            if (a==b) // equilatero oppure isoscele
                  if (b==c) // equilatero
                        System.out.println("equilatero");
                  else // isoscele
                        System.out.println("isoscele");
            else // isoscele oppure scaleno
                  if (b==c) // isoscele
                        System.out.println("isoscele");
                  else // isoscele oppure scaleno
                        if (a==c) // isoscele
                              System.out.println("isoscele");
                        else // scaleno
                              System.out.println("scaleno");
      }
}

/**
* Classificazione di un triangolo date le misure dei suoi lati
* Versione con contatore dei lati uguali.
*/
public class TriangoloB {
      public static void main(String[] args) {
            int uguali; // numero di coppie di lati uguali
            // legge il valore dei tre lati
            System.out.println("Scrivi la lunghezza dei lati di un triangolo ");
            int a = Console.readInt("a=");
            int b = Console.readInt("b=");
            int c = Console.readInt("b=");
            System.out.print("Il triangolo e' ");
            // calcola il numero di coppie di lati uguali
            uguali = 0;
            // confronta i lati
            if (a==b)
                  uguali++;
            if (a==c)
                  uguali++;
            if (b==c)
                  uguali++;
            // stampa il messaggio
            if (uguali==0)
                  System.out.println("scaleno");
            else if (uguali==1)
                        System.out.println("isoscele");
                  else // uguali vale sicuramente 3
                        System.out.println("equilatero");
            }
      }

/**
* Classificazione di un triangolo date le misure dei suoi lati
* Versione con condizioni composte.
*/
public class TriangoloC {
      public static void main(String[] args) {
            /* legge i tre numeri */
            System.out.println("Scrivi la lunghezza dei lati di un triangolo ");
            int a = Console.readInt("a=");
            int b = Console.readInt("b=");
```

```
            int c = Console.readInt("c=");
            System.out.print("Il triangolo e' ");
            /* calcola il tipo di triangolo,
             confrontando i lati, e stampa il messaggio */
            if (a==b && a==c)
                  System.out.println("equilatero");
            else if (a==b || a==c || b==c)
                        System.out.println("isoscele");
                  else
                        System.out.println("scaleno");
      }//main
}//class
```

Istruzioni di iterazione

7. Esempi di utilizzo dell'indice del for

```
public class EsempiDiFor {
public static void main (String args[])
{// nota   "\t" serve per tabulare  "\n" per andare da capo
      int n=Console.readInt("Inserisci numero");

      System.out.println("stampa dei numeri da 0 a n-1");
      for(int i=0; i<n; i++)
            System.out.println(i);

      System.out.println("stampa dei primi n-1 numeri pari");
      for(int i=0; i<n; i++)
            System.out.println(i+"\t"+(i*2));

      System.out.println("stampa dei primi n-1 numeri dispari");
      for(int i=0; i<n; i++)
            System.out.println(i+"\t"+(i*2+1));

      //stampa la tabellina di m
      System.out.println("stampa la tabellina di un numero");
      int m=Console.readInt("Inserisci numero");
      for(int i=1; i<10; i++)
            System.out.println(i+"*"+m+"="+(i*m));

      System.out.println("stampa dei numeri da n-1 a 0");
      for(int i=0; i<n; i++)
            System.out.println(i+"\t"+(n-1-i));

      System.out.println("stampa dei numeri da n-1 a 0");
      for(int i=n-1; i>=0; i--)
            System.out.println( (n-1-i)+"\t"+i);

      //Numerazioni
      System.out.println("stampa delle numerazioni a passo a passo partendo da
inizio fino a n");
      n=Console.readInt("Inserisci n");
      int inizio=Console.readInt("Inserisci inizio");
      int passo=Console.readInt("Inserisci passo");
      for(int i=inizio ;i<n;i+=passo)
      {
            System.out.print( i +"\t");
      }

      System.out.println("\ncome prima solo che mette 5 numeri a riga");
      int contaNumeri=1;
```

```
      for(int i=inizio ;i<n;i+=passo)
      {
            System.out.print( i +"\t");
            if( contaNumeri % 5==0  ) System.out.println();
            contaNumeri++;// conta i giri del for non usa l'indice i
      }
      System.out.println("\ncome prima");
      for(int i=0 ;i<n/passo;i++)
            {
                  System.out.print( i*passo+inizio +"\t");
                  if( (i+1)% 5==0  ) System.out.println();
            }
       }
}
```

8. Dati in input il numero di esami sostenuti con i rispettivi voti, calcolarne la media

```
public class MediaEsami{
      public static void main (String args[]){
            int numeroEsami = Console.readInt ("Numero esami: ");
            int totale = 0;
            for (int i = 1; i <= numeroEsami; i++)
                        totale += Console.readInt ("Voto esame n. "+i+": ");
                        //conversione di tipo esplicita
             float media = (float)totale/numeroEsami;
             System.out.println ("media = "+media);
      }
}
```

9. Calcolare la somma dei multipli di x fino a 100

```
public class SommaMultipli {
public static void main (String args[])
{
      int multiplo,somma;
      int i;
      multiplo=Console.readInt("multiplo=");
      somma=0;
      for (i=multiplo;i<=100;i=i+multiplo)
            somma=somma+i;
      System.out.println("la somma dei multipli e' "+somma);
      }
}
```

10. Dati in input N numeri, calcolarne la media.

```
public class StampaMedia {
      public static void main (String args[])
      {
            int n, num;
            int i;
            int somma;
            somma=0;
            n=Console.readInt("Quanti sono i numeri ?");
            i=0;
            while (i<n)
                  {
                     num=Console.readInt("Dammi un numero>");
                     somma+=num;
                     i++;
                  }
            float media = (float)somma/n; //conversione di tipo esplicita
```

```
            System.out.println ("media = "+media);
      }
}
```

11. Leggere interi positivi terminati da un numero negativo e contare il numero di pari e il numero di dispari.

```
public class ContaPariEDispari {
      public static void main (String args[])
      {
            int numero,contaPari,contaDispari;
            contaPari=0;
            contaDispari=0;
            numero=Console.readInt("Inserisci numero (negativo per finire) ");
            while (numero>0)
                  {
                        if (numero%2 == 0)
                              contaPari++;
                        else
                              contaDispari++;
                        numero=Console.readInt("Inserisci numero (negativo per
            finire) ");
                  } // fine while
            System.out.println(" num. pari= "+contaPari+" num. dispari=
      "+contaDispari);
      }
}
```

12. Legge N numeri e segnala la presenza di almeno un numero pari

```
public class almenoUno {
      public static void main (String args[])
      {
            int n=Console.readInt("n= ");
            boolean trovato=false;
            for (int i=0;i<n;i++){
                  int num=Console.readInt("numero= ");
                  if(num%2==0) trovato=true;// e' pari almeno uno
                  //trovato=trovato || num%2==0;//questa e' una alternativa
all'istruzione precedente
            }
            if(trovato) System.out.println("C'e' almeno un numero pari");
            }
}
```

13. Legge N numeri e verifica se sono tutti pari

```
public class tuttiPari {
      public static void main (String args[])
      {
            int n=Console.readInt("n= ");
            boolean ePari=true;
            for (int i=0;i<n;i++){
                  int num=Console.readInt("numero= ");
                  if(!(num%2==0)) ePari=false;// ne ho trovato uno dispari
                  //ePari=ePari && num%2==0;//questa e' una alternativa
all'istruzione precedente
            }
            if(ePari) System.out.println("Sono tutti pari");
            }
}
```

14. Verifica se una sequenza di numeri positivi, introdotta come input da tastiera e terminata da un numero >=0, è in ordine strettamente crescente

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

            boolean crescente=true;
            int prec=0;
            int n=Console.readInt("Numero: ");
            while (n>0){
                  if (n>prec)
                        prec=n;
                  else{

                        crescente=false;
                        break;
                  }
                  n=Console.readInt("Numero: ");
            }
            if (crescente)
                  System.out.println("Sequenza crescente");
            else
                  System.out.println("Sequenza non crescente");
      System.out.println();
      }
}
```

15. Trovare la stringa più lunga in una sequenza di stringhe terminata da una stringa nulla.

```
public class MaxStringa {
      public static void main (String args[])
      {
            int max=0;
            String maxS=”“;
            String s;

            s=Console.readString("Inserisci una stringa ( INVIO per finire) ");
            while (s.length()>0)
                  {
                        if (s.length()>max)
                              {
                                    max=s.length();
                                    maxS=s;
                              }
                        s=Console.readString("Inserisci una stringa ( INVIO per
            finire) ");
                  }// fine while
            System.out.println(" La stringa massima e’: "+maxS+" di lunghezza
      “+max);
      }
}
```

16. Calcolare la somma dei primi n numeri naturali

```
public class StampaSomma {
public static void main (String args[])
{
      int n;
      int i;
```

```
      int somma;
      somma=0;
      n=Console.readInt("Fino a che numero?");
      i=1;
      while (i<=n)
      {
            somma=somma+i;
            i++;
            System.out.println("La somma parziale e'="+somma);
      }
      System.out.println("La somma e'="+somma);
      }
}
```

17. Calcolare il fattoriale di un numero.

```
public class Fattoriale {
public static void main (String args[])
      {
      int numero, tmp, fatt;
      numero=Console.readInt("Dammi il Numero");
      if (numero==0)
         fatt=1;
      else
         {
            tmp = numero ;
            fatt=1;
            for (tmp=numero;tmp>1; tmp--)
               fatt=fatt*tmp;
         }
      System.out.println("il numero "+numero+" ha fattoriale "+fatt);
      }
}
```

18. Calcolare il fattoriale di un numero (con verifica dell'overflow):

```
public class Fattoriale2{
  public static void main (String args[])
  {
      int n = Console.readInt("n=");
      if (n < 0)
            System.out.println("Impossibile calcolare il fattoriale di un numero
negativo");
      else
            {
            int fattoriale = 1;
            boolean ok = true;
            for (int i = 2; i <= n && ok; i++)
            {
                  if (Integer.MAX_VALUE/fattoriale < i)
                        {
                        System.out.println("Il fattoriale e' troppo grande");
                        ok = false;
                        }
                  fattoriale *= i; // fattoriale = fattoriale * i
            }
            if (ok) System.out.println(n+"! = "+fattoriale);
            }
  }
}
```

19. Verificare se un numero è primo

```
public class Primi {
public static void main (String args[])
{
      int possibilePrimo,divisore;
      boolean ePrimo;
      possibilePrimo= Console.readInt("Numero=");
      divisore=possibilePrimo / 2;
      ePrimo=true; // ipotesi ottimistica il numero è primo
      while (divisore>1 && ePrimo)
            {
                  if (possibilePrimo % divisore == 0)
                        ePrimo=false;
                  divisore--;
            }
      if (ePrimo)
            System.out.println("Il numero "+possibilePrimo+" e' primo");
      else
            System.out.println("Il numero "+possibilePrimo+" non e' primo");
      }
}
```

20. Calcolare il numero di cifre di un intero (do ... while).

```
public class ContaCifre {
public static void main (String args[])
      {
            int numero,tmp,numcifre;
            numero=Console.readInt("Inserisci numero ");
            tmp = numero;
            numcifre=0;
            do
            {
              tmp=tmp/10;
              numcifre++;
            }
            while (tmp!=0);
            System.out.println("Il numero "+numero+" ha "+numcifre+" cifre.");
      }
}
```

21. Data in input una data come giorno, mese ed anno, calcolare il numero di giorni trascorsi dall'inizio dell'anno

```
public class GiorniTrascorsi{
      public static void main (String args[])
      {
                  int giorno = Console.readInt ("giorno: ");
             int mese = Console.readInt ("mese: ");
             int anno = Console.readInt ("anno: ");
             int giorniTrascorsi = 0;

             for (int m = 1; m < mese; m++){
                        switch (m){
                        case 4:
                        case 6:
                        case 9:
                        case 11:
                                giorniTrascorsi += 30;
                                break;
                        case 2:
                               if (anno%400==0 || (anno%4==0 && !(anno%100==0)))
                                          giorniTrascorsi += 29;//anno bisestile
```

```
                        else
                                giorniTrascorsi += 28;
                        break;
                  default:
                        giorniTrascorsi += 31;
            }
      }
      giorniTrascorsi += giorno;
      System.out.println("Il "+giorno+"/"+mese+"/"+anno+" è il
      "+giorniTrascorsi+"° giorno dell'anno "+anno);
   }
}
```

22. Dato in input un numero n, calcolare i primi n numeri della serie di Fibonacci (ogni numero è dato dalla somma dei due precedenti: 1,1,2,3,5,8,13,21,…)

```
public class fibonacci{
   public static void main (String [] args){
      int n=Console.readInt("Inserisci n");
      int a=1; //contiene il valore del terzultimo numero
      int b=1;//contiene il valore del penultimo numero
      int f;
      System.out.println("I primi "+n+" numeri di Fibonacci sono:");
      for (int i=1;i<=n;i++){
         if (i<=2)
            f=1;
         else
         {
            f=a+b;
            a=b;
            b=f;
         }
         System.out.print(" "+f);
      }
   }
}
```

Esercizi con For annidati

23. Stampare la tabellina

```
public class Tabellina{
   public static void main(String [] args){
      int n=10;
      System.out.println("\t\t\t\tT a b e l l i n a\n\n");
      for (int i=1;i<=n;i++){
         for (int j=1;j<=n;j++)
            System.out.print(i*j+"\t" );
         System.out.println();
      }
   }
}
```

24. Dato in input un numero n visualizzare in output un quadrato di asterischi di lato n
Es: Input: n=4 Output:

```
****
****
****
****
```

```
public class QuadratoPieno{
```

```
      public static void main(String [] args){
         int n=Console.readInt("Inserisci la misura del lato del quadrato: ");
            for (int i=1;i<=n;i++){
                  for (int j=1;j<=n;j++)
                        System.out.print("*");
                  System.out.println();
            }
      }
}
```

25. Dato in input un numero n visualizzare in output un quadrato vuoto di asterischi di lato n
Es: Input: n=4 Output:

```
****
*  *
*  *
****
```

```
public class QuadratoVuoto{
      public static void main(String [] args){
            int n=Console.readInt("Inserisci la misura del lato del quadrato:");
            for (int i=1;i<=n;i++){
                  for (int j=1;j<=n;j++){
                        if ((i==1) || (j==1) ||(j==n) ||(i==n))
                              System.out.print("*");
                        else
                              System.out.print(" ");
                  }
                  System.out.println();
            }
      }
}
```

26. Dato in input un numero n visualizzare in output un triangolo di asterischi di lato n
Es: Input: n=4 Output:

```
*
**
***
****
```

```
public class TriangoloPieno{
      public static void main(String [] args){
            int n=Console.readInt("Inserisci la misura del lato del quadrato:");
            for (int i=1;i<=n;i++){
                  for (int j=1;j<=i;j++)
                        System.out.print("*");
                  System.out.println();
            }
      }
}
```

Esercizi con i vettori

27. Esempi di stampe di elementi di un vettore

```
public class EsempiDiStampeDiVettori {
public static void main (String args[])
{
// nota   "\t" serve per tabulare

      int n=Console.readInt("Dimensione del vettore: ");
      int v[]=new int[n];
      for (int i=0;i<n;i++)
            v[i]=Console.readInt(" Elemento "+(i+1)+": ");
```

```
      System.out.println("Elementi del vettore");
      for(int i=0; i<n; i++)
            System.out.println("v["+i+"]="+v[i]);


      System.out.println("Elementi del vettore di posizione pari");
      for(int i=0; i<n/2; i++)
            System.out.println("v["+i+"]="+v[i*2]);


      System.out.println("Elementi del vettore di posizione dispari");
      for(int i=1; i<n; i+=2)
            System.out.println("v["+i+"]="+v[i]);

      System.out.println("Elementi del vettore dell'ultimo al primo");
      for(int i=n-1; i>=0; i--)
            System.out.println("v["+i+"]="+v[i]);

      System.out.println("Elementi consecutivi del vettore");
      for(int i=0; i<n-1; i++)
            System.out.println("v["+i+"]="+v[i]+"\t"+"v["+(i+1)+"]="+v[i+1]);

      System.out.println("Elementi simmetrici del vettore");
      for(int i=0; i<n/2; i++)

      System.out.println("v["+i+"]="+v[i]+"\t"+"v["+(n/2+i)+"]="+v[n/2+i]);
  }
}
```

28. Calcolo della media degli esami (usare i vettori)

```
public class MediaEsami2
{
  public static void main (String args[])
  {
    int numeroEsami = Console.readInt ("Numero esami: ");

    int esami[] = new int[numeroEsami];

    for (int i = 0; i < esami.length; i++)
      esami[i] = Console.readInt ("Voto esame n. "+(i+1)+": ");

    int totale = 0;

    for (int i = 0; i < esami.length; i++)
      totale += esami[i];

    float media = (float)totale/numeroEsami;

    System.out.println ("media = "+media);
  }
}
```

29. Calcolo e Visualizzazione di una tavola pitagorica (con vettori)

```
public class TavolaPitagorica{
      public static void main(String args[]){
            int vettore[];
```

```
            int n=Console.readInt("Quale Tavola pitagorica vuoi visualizzare:");

            int tavola1[]=new int[10];
            int tavolaPitagorica[]=new int[10];

            for (int i=0;i<tavola1.length;i++)
                  tavola1[i]=i+1;

            for (int i=0;i<tavola1.length;i++)
                  tavolaPitagorica[i]=tavola1[i]*n;

            System.out.print("La tavola pitagorica e': [");
            for (int i=0;i<tavolaPitagorica.length;i++)
                  System.out.print(tavolaPitagorica[i]+" ");
            System.out.println("]");

      }
}
```

30. Calcolo della somma e del prodotto scalare di due vettori:

```
   public class OperazioniVettori{

      public static void main(String args[]){

            //dichiarazione dei vettori
            int v1[],v2[],somma[];
            int n=Console.readInt("Numero di elementi di entrambi i vettori:");

            //allocazione dei vettori di n elementi ciascuno
            v1=new int[n];
            v2=new int[n];
            somma=new int[n];

            //ciclo di lettura da input del 1° vettore
            for(int i=0;i<n;i++)
                  v1[i]=Console.readInt("Inserire elemento "+i+" di V1");

            //ciclo di lettura da input del 2° vettore
            for(int i=0;i<n;i++)
                  v2[i]=Console.readInt("Inserire elemento "+i+" di V2");

            //calcolo della somma
            System.out.print("Il vettore somma V1+V2 e' :");
            for(int i=0;i<n;i++){
                  somma[i]=v1[i]+v2[i];
                  System.out.print(somma[i]+" ");
            }//for
            System.out.println("");

            //calcolo del prodotto scalare
            int ps=0;
            for(int i=0;i<n;i++)
                  ps+=v1[i]*v2[i];
            System.out.println("Il prodotto scalare di V1 e V2 e': "+ps);
      }//main
}//class
```

31. VERIFICA Se un vettore ha almeno un elemento pari, varie versioni.

```
//VERIFICA Se un vettore ha almeno un elemento pari,
// varie versioni
```

```
public class AlmentoUnoPari {
	public static void main(String args[]){
		// dichiara e leggi il vettore
		int vettore[];
		int n=Console.readInt("Dimensione del vettore: ");
		vettore=new int[n];
		for (int i=0;i<n;i++)
			vettore[i]=Console.readInt(" Elemento "+(i+1)+": ");
/*
	  // verifica   non ottimizzata
		boolean trovatoPari=false;
		for (int i=0;i<vettore.length;i++)
			if (vettore[i]%2==0)  trovatoPari=true;
*/
/*
	  // verifica
		boolean trovatoPari=false;
		for (int i=0;i<vettore.length && !trovatoPari ;i++)
			if (vettore[i]%2==0)  trovatoPari=true;
*/
/*
	  // verifica
		boolean trovatoPari=false;
		for (int i=0;i<vettore.length;i++)
			if (vettore[i]%2==0) { trovatoPari=true;
                                                        break ;}
		// stampa risultato
		if (trovatoPari)
			System.out.println("Il vettore ha almeno un elemento pari.");
		else
			System.out.println("Il vettore non ha alcun elemento pari.");
*/
/*
	  // verifica
		int i
		for (i=0;i<vettore.length;i++)
			if (vettore[i]%2==0)break ;
*/
		// verifica
		int i
		for (i=0;i<vettore.length && !(vettore[i]%2==0;i++);

		// stampa risultato
		if ( i<vettore.length )
			System.out.println("Il vettore ha almeno un elemento pari.");
		else
			System.out.println("Il vettore non ha alcun elemento pari.");
	}
}
```

32. VERIFICA Se un vettore ha tutti gli elementi pari, varie versioni

```
//VERIFICA Se un vettore ha tutti gli elementi pari,
public class TuttiPari {
	public static void main(String args[]){
		// dichiara e leggi il vettore
		int vettore[];
		int n=Console.readInt("Dimensione del vettore: ");
		vettore=new int[n];
		for (int i=0;i<n;i++)
			vettore[i]=Console.readInt(" Elemento "+(i+1)+": ");
```

```
        // verifica  non ottimizzata
            boolean ePari=true;
            for (int i=0;i<vettore.length;i++)
                  if (!(vettore[i]%2==0) )  ePari=false;
/*
        // verifica
            boolean ePari=true;
            for (int i=0;i<vettore.length && ePari ;i++)
                  if (vettore[i]%2!=0)  trovatoPari=false;
*/
/*
        // verifica
            boolean ePari=true;
            for (int i=0;i<vettore.length;i++)
                  if (vettore[i]%2!=0) { trovatoPari=false;break ;}
*/
            // stampa risultato
            if (ePari)
                  System.out.println("Il vettore ha tutti gli elementi pari.");
            else
                  System.out.println("Il vettore non ha tutti gli elementi
pari.");

/*
        // verifica
            int i
            for (i=0;i<vettore.length;i++)
                  if (vettore[i]%2!=0)break ;
*/
/*
            // verifica
            int i
            for (i=0;i<vettore.length && vettore[i]%2==0;i++);

            // stampa risultato
            if ( i==vettore.length )
                  System.out.println("Il vettore ha tutti gli elementi pari.");
            else
                  System.out.println("Il vettore non ha tutti gli elementi
pari.");
*/
   }
}
```

Esercizi proposti - Verificare

1. Vettore Ordinato V[i]<V[i+1] con i=0..n-2
2. Vettore Simmetrico V[i]==V[n-1-i] con i=0..n/2
3. Prima metà uguale alla seconda metà V[i]==V[n/2+i] con i=0..n/2
4. Vettore con elementi pari in posizione pari e dispari in posizione dispari

(I%2==0 && V[i]%2==0) || (I%2==1 && V[i]%2==1)

oppure

i%2==V[i]%2 ovvero hanno lo stesso resto nella divisione per 2

5. Vettore di elementi uguali tra loro V[i]==V[0] con i=1..n-1

oppure

V[i]==V[i+1] con i=0..n-2

33. VERIFICA DELL'ORDINAMENTO DI UN VETTORE, Versione con l'uso di una variabile booleana

```
public class VettoreOrdinato {
      public static void main(String args[]){
            int vettore[];

            int n=Console.readInt("Dimensione del vettore: ");
            vettore=new int[n];
            for (int i=0;i<n;i++)
                  vettore[i]=Console.readInt(" Elemento "+(i+1)+": ");

            boolean ordinato=true;
            for (int i=1;i<vettore.length && ordinato;i++)
                  if (!(vettore[i-1]<=vettore[i]))
                        ordinato=false;

            if (ordinato)
                  System.out.println("Il vettore e' ordinato.");
            else
                  System.out.println("Il vettore e' disordinato.");

      }
}
```

34. Verifica dell'ordinamento di un vettore. Versione con l'uso dell'istruzione break.

```
public class VettoreOrdinato2 {
      public static void main(String args[]){
            int vettore[];

            int n=Console.readInt("Dimensione del vettore: ");
            vettore=new int[n];
            for (int i=0;i<n;i++)
                  vettore[i]=Console.readInt(" Elemento "+(i+1)+": ");

            int i;
            for (i=0;i<vettore.length-1;i++)
                  if (!(vettore[i]<=vettore[i+1]))
                        break;

            if (i<vettore.length-1)
                  System.out.println("Il vettore e' disordinato.");
            else
                  System.out.println("Il vettore e' ordinato.");

      }
}
```

35. Da esercizio 2 del compito 15 dicembre 2005

```
/* Si scriva un programma AzzeraCoppie che riceve in ingresso un vettore di
interi v,
```

```
e restituisce un vettore di interi w ottenuto ponendo a zero le coppie di
elementi consecutivi uguali.
Ad   esempio, se v = [ 7, 3, 3, 4, 1, 8, 6, 6, 6, 5] ,
il  vettore restituito è w = [7, 0, 0, 4, 1, 8, 0, 0, 0, 5].
*/
public class AzzeraCoppie{
      public static void main(String args []){

            int [] v={7, 3, 3, 4, 1, 8, 6, 6, 6, 5};
            // faccio una copia w del vettore v
            int n=v.length;
        int [] w= new int[n];
            for(int i=0; i<n;i++)
                  w[i]=v[i];
            // pongo a zero le coppie di elementi consecutivi uguali
        if(v[0]==v[1]) w[0]=0;// ipotizzo che il vettore ha almeno due elementi

        for (int i=1;i<n-1;i++)
            if(v[i]==v[i-1] || v[i]==v[i+1]) w[i]=0;

            if(v[n-1]==v[n-2]) w[n-1]=0;

            for(int i=0; i<w.length;i++)
                  System.out.println(w[i]);
      }
}
```

36. Da esercizio 2 del 12 settembre 2005

```
/*Si scriva un metodo GeneraVettore che riceve in ingresso tre vettori di interi
della
stessa dimensione pari V1,  V2 e V3 , e restituisce un vettore di boolean V4 in
cui
la generica posizione i è così ottenuta:
·     V4[i] = true       se V1[i] è maggiore di tutti gli elementi della prima
metà V2
ed è minore di V3[i];
·     V4[i] = false     altrimenti
Ad   esempio, se    V1= [ 20 , 18 , 4, 27 ] ,    V2= [ 10 , 9 , 7, 4]   e
V3= [ 32 , 3 , 7, 40]  il  vettore   restituito  è
V4 = [ true, false, false, true].
*/
public class GeneraVettore{
      public static void main(String args []){

            //i tre vettori;
            int [] v1={20 , 18 , 4, 27};
            int [] v2={10 , 9 , 7, 4};
            int [] v3={ 32 , 3 , 7, 40};
          // in nuovo vettore di booleani
            int n=v1.length;
        boolean [] v4= new boolean[n];


        for (int i=0;i<n;i++){
            // inizio V1[i] è maggiore di TUTTI gli elementi della prima metà V2
                  boolean eMaggioreDiTutti=true;
                  for(int k=0;k<n/2;k++)
                        if( v1[i]<=v2[k]){eMaggioreDiTutti=false;break;}
                  // fine TUTTI

                  v4[i]=eMaggioreDiTutti && (v1[i]<v3[i]);
                  }
```

```
            for (int i=0;i<n;i++)
                  System.out.println(v4[i]);
      }
}
```

37. Copia gli elementi pari di un vettore in un altro vettore (Due versioni)

```
/*    Copia gli elementi pari di un vettore in un altro vettore  Versione 1
*/
public class copiaPari
{
  public static void main (String args[])
  {
      int n = Console.readInt ("n= ");
      int a[] = new int[n];
      //leggiamo il vettore
      for (int i = 0; i < a.length; i++)
        a[i] = Console.readInt ("a["+i+"]=");
      //stampiano il vettore
      for (int i = 0; i < a.length; i++)
        System.out.print("a["+i+"]="+a[i]+"  ");
      System.out.println ();
      //contiamo gli elementi pari
      int contaPari=0;
      for (int i = 0; i < a.length; i++)
        if (a[i] %2==0) contaPari++;
      // creiamo un nuovo vettore di dimensione opportuna
      int pari[]=new int[contaPari];
      // copiamo i pari nel nuovo vettore
      int k=0;
      for (int i = 0; i < a.length; i++)
        if (a[i]%2 ==0){
                  pari[k]= a[i];
                  k++;
            }
      //stampiano il vettore dei pari
      for (int i = 0; i < pari.length; i++)
            System.out.print("pari["+i+"]="+pari[i]+"  ");
      System.out.println ();
  }
}

/*    Versione n.2  */
public class copiaPari2
{
  public static void main (String args[])
  {
      int n = Console.readInt ("n= ");
      int a[] = new int[n];

      //leggiamo il vettore
      for (int i = 0; i < a.length; i++)
        a[i] = Console.readInt ("a["+i+"]=");

      //stampiano il vettore
      for (int i = 0; i < a.length; i++)
        System.out.print("a["+i+"]="+a[i]+"  ");
      System.out.println ();

      // creiamo un nuovo vettore di dimensione massima
      int temp[]=new int[a.length];
```

```
      // prendiamo i pari nel nuovo vettore
      int k=0;
      for (int i = 0; i < a.length; i++)
        if (a[i]%2 ==0){
                  temp[k]= a[i];
                  k++;
             }
      // creiamo un nuovo vettore di dimensione apportuna
      int pari[]=new int[k];

      // travasiamo
      for (int i = 0; i < pari.length; i++)
            pari[i]=temp[i];

      //stampiano il vettore dei pari
      for (int i = 0; i < pari.length; i++)
            System.out.print("pari["+i+"]="+pari[i]+"  ");
      System.out.println ();
  }
}
```

38. CALCOLO DEL MASSIMO; SUBMASSIMO, MINIMO DI UN VETTORE

```
public class MassimiMinimi{

  public static void main(String args[]){

    //dichiarazione del vettore
    int v[];
    int n=Console.readInt("Numero di elementi del vettore:");
    //allocazione del vettore di n elementi
    v=new int[n];

    //ciclo di lettura da input del vettore
    for(int i=0;i<n;i++)
      v[i]=Console.readInt("Inserire elemento "+i);

    //calcolo di massimi e minimi
    int massimo=v[0];
    int s_massimo=Integer.MIN_VALUE;
    int minimo=v[0];
    for(int i=1;i<n;i++){
      if (v[i]>massimo){//occorre aggiornare massimo e sub-massimo
        s_massimo=massimo;
        massimo=v[i];
      }
      else if (v[i]>s_massimo && v[i]<massimo)//aggiorno solo il sub-massimo
        s_massimo=v[i];
      if (v[i]<minimo)
        minimo=v[i];
    }//for

    System.out.println("Il massimo è: "+massimo);

    if(s_massimo==Integer.MIN_VALUE)
      System.out.println("Il sub-massimo non è definito per questo vettore");
    else
      System.out.println("Il sub-massimo è: "+s_massimo);
    System.out.println("Il minimo è: "+minimo);
  }//main
}//class
```

39. Verifica della simmetria ed inversione di un vettore

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

		//dichiarazione del vettore
		int v[];
		int n=Console.readInt("Numero di elementi del vettore:");
		//allocazione del vettore di n elementi
		v=new int[n];
		//ciclo di lettura da input del vettore
		for(int i=0;i<n;i++)
			v[i]=Console.readInt("Inserire elemento "+i);
		boolean simmetrico=true;
		for (int k=0;k<v.length/2 && simmetrico;k++)
			if(v[k]!=v[v.length-k-1])
				simmetrico=false;
		if (simmetrico)
			System.out.println("Il vettore è simmetrico e non ha bisogno
di essere invertito.");
		else{
			//inversione del vettore
			System.out.println("Il vettore non e' simmetrico.");
			System.out.println("Il vettore invertito e': ");
			int i=0;
			while(i<n/2){
				int temp;
				temp=v[i];
				v[i]=v[v.length-i-1];
				v[v.length-i-1]=temp;
				i++;
			}
			i=0;
			do {
				System.out.print(v[i]+" ");
				i++;
				}
			while(i<n);
		}//else
	}//main
}//class
```

Esercizi con i metodi

40. Dato in input un numero N (numero di elementi) scrivere tre metodi che, a partire dal parametro N, leggano da input N interi, N double ed N Stringhe e restituiscano tre vettori contenenti gli elementi letti. Scrivere poi tre metodi, senza parametri, che stampino a video i tre differenti vettori ed utilizzare questi metodi nel main (esempio di ridefinizione metodi con lo stesso nome).

```
public class LeggiNVettori
{
	public static int [] leggiInt(int N)
	{
		int [] vettore = new int[N];
		for (int i = 0; i < N; i++)
			vettore[i] = Console.readInt("Inserisci l'intero numero " + (i
+ 1) + ": ");
		return vettore;
```

```
	}
	public static double [] leggiDouble(int N)
	{
		double [] vettore = new double[N];
		for (int i = 0; i < N; i++)
			vettore[i] = Console.readDouble("Inserisci il double numero "
+ (i + 1) + ": ");
		return vettore;
	}
	public static String [] leggiString(int N)
	{
		String [] vettore = new String[N];
		for (int i = 0; i < N; i++)
			vettore[i] = Console.readString("Inserisci la stringa numero "
+ (i + 1) + ": ");
		return vettore;
	}
	public static void stampaInt(final int [] vettore)
	{
		for (int i = 0; i < vettore.length; i++)
			System.out.print(vettore[i] + " ");
		System.out.println();

	}
	public static void stampaDouble(final double [] vettore)
	{
		for (int i = 0; i < vettore.length; i++)
			System.out.print(vettore[i] + " ");
		System.out.println();

	}
	public static void stampaString(final String [] vettore)
	{
		for (int i = 0; i < vettore.length; i++)
			System.out.print(vettore[i] + " ");
		System.out.println();

	}
	public static void main (String args[])
	{
		// Numero di elementi nei vettori
		int dimensioneVettore = Console.readInt ("Numero di elementi da
leggere: ");
		int [] vettoreInteri = leggiInt(dimensioneVettore);
		double [] vettoreDouble = leggiDouble(dimensioneVettore);
		String [] vettoreString = leggiString(dimensioneVettore);
		stampaInt(vettoreInteri);
		stampaDouble(vettoreDouble);
		stampaString(vettoreString);
	}
}
```

41. Scrivere, ed usare nel main, una funzione che calcoli il prodotto scalare di due vettori di interi ed una funzione che ne calcoli la somma.

```
public class OperazioniVettoriali
{
	public static int prodottoScalare (int [] vettore1, int [] vettore2)
	{
		int risultato = 0;
		for (int i = 0; i < vettore1.length; i++)
			risultato = risultato + vettore1[i]*vettore2[i];
		return risultato;
```

```
	}
	public static int [] somma (int [] vettore1, int [] vettore2)
	{
		int [] risultato = new int[vettore1.length];
		for (int i = 0; i < vettore1.length; i++)
			risultato[i] =  vettore1[i] + vettore2[i];
		return risultato;
	}

	public static void main (String args[])
	{
		int [] vettore1 = {1, 5, 7, 1, 8, 2};
		int [] vettore2 = {1, 8, 0, 1, -3, 1};
		int prodotto = prodottoScalare(vettore1, vettore2);
		int [] sommaVettori = somma (vettore1, vettore2);

	}

}
```

42. Scrivere, ed usare nel main, un metodo che riceva in input un vettore di interi ed una valore intero di soglia e restituisca in output un vettore con tutti gli elementi del vettore di valore minore o uguale a quelli della soglia data. Nella stessa classe scrivere un metodo che riceva in input due vettori di uguale dimensione e restituisca un terzo vettore i cui elementi siano gli elementi del primo vettore uguali agli elementi del secondo vettore.

```
public class SottoinsiemiVettoriali
{
	public static int [] vettoreSoglia (int [] vettore, int soglia)
	{
		int dimensioneSoglia = 0;
		for (int i = 0; i < vettore.length; i++)
			if (vettore[i] <= soglia)
				dimensioneSoglia++;
		int [] risultato = new int [dimensioneSoglia];
		int k = 0;
		for (int i = 0; i < vettore.length; i++)
			if (vettore[i] <= soglia)
			{
				risultato[k] = vettore[i];
				k++;
			}
		return risultato;
	}
	public static int [] vettoreUguaglianze (int [] vettore1, int [] vettore2)
	{
		int dimensioneRisultato = 0;
		for (int i = 0; i < vettore1.length; i++)
			if (vettore1[i] == vettore2[i])
				dimensioneRisultato ++;
		int [] risultato = new int [dimensioneRisultato];
		int k = 0;
		for (int i = 0; i < vettore1.length; i++)
			if (vettore1[i] == vettore2[i])
			{
				risultato[k] = vettore1[i];
				k++;
			}
		return risultato;
	}
```

```
	public static void main (String args[])
	{
		int soglia = 12;
		int [] vettore1 = {1, 35, 34, -2, 11, 2, -3};
		int [] vettore2 = {1, 3, 34, -2, 9, 2, -8};
		int [] soglie = vettoreSoglia(vettore1, soglia);
		int [] uguaglianze = vettoreUguaglianze(vettore1, vettore2);
	}
}
```

43. Scrivere un metodo che riceva in ingresso due vettori di interi (non necessariamente di uguale dimensione) e restituisca true se tutti gli elementi del primo vettore (fino alla fine del più piccolo dei vettori) sono strettamente maggiori di quelli del secondo vettore.

```
public class ControlloVettori
{
	public static boolean controlla(int [] vettore1, int [] vettore2)
	{
		if (vettore1.length < vettore2.length)
			for(int i = 0; i < vettore1.length; i++)
			{
				if (vettore1[i] <= vettore2[i])
					return false;
			}
		else
			for(int i = 0; i < vettore2.length; i++)
			{
				if (vettore1[i] <= vettore2[i])
					return false;
			}
		return true;
	}
	public static void main (String args[])
	{
		int [] vettore1 = {1, 35, 34, -2, 11, 2, -3};
		int [] vettore2 = {1, 3, 34, -2, 9};
		System.out.println(controlla(vettore1, vettore2));
	}
}
```

44. VERIFICA DELL'ORDINAMENTO DI UN VETTORE

```
public class VettoreOrdinatoMetodo {

	public static boolean eOrdinato(final int []v){

		for (int i=1;i<v.length;i++)
			if (!(v[i-1]<=v[i])) return false;
		return true;
	}

	public static void main(String args[]){
		int vettore[];

		int n=Console.readInt("Dimensione del vettore: ");
		vettore=new int[n];
		for (int i=0;i<n;i++)
			vettore[i]=Console.readInt(" Elemento "+(i+1)+": ");

		if (eOrdinato(vettore))
			System.out.println("Il vettore e' ordinato.");
		else
```

```
                System.out.println("Il vettore e' disordinato.");
        }
}
```

45.Verifica della simmetria ed inversione di un vettore

```
public class SimmetriaInversioneMetodi{
        public static boolean eSimmetrico(int [] vett)
        {
                for (int k=0;k<vett.length/2;k++)
                        if(vett[k]!=vett[vett.length-k-1]) return false;
                return true;
        }
        //inverto lo stesso vettore
        public static void inverti(int [] v)
        {
                int temp;
                for (int i=0;i<v.length/2;i++){
                        // facciamo uno scambio
                        temp=v[i];
                        v[i]=v[v.length-i-1];
                        v[v.length-i-1]=temp;
                }//for
        }//inverti

        //restituisco un vettore invertito
        public static int [] invertiBis(int [] v)
                {
                        int [] vett=new int[v.length];
                        for (int i=0;i<v.length;i++){
                                // copio dall'ultimo
                                vett[i]=v[v.length-i-1];
                                }
                        return vett;
        } //inverti

        public static void stampa(final int[] v, String messaggio)
        {
                System.out.println(messaggio);
                for(int i=0;i<v.length;i++)
                        System.out.println(v[i]);
                }
        public static void main(String args[]){

                //dichiarazione del vettore
                int v[];
                int n=Console.readInt("Numero di elementi del vettore:");
                //allocazione del vettore di n elementi
                v=new int[n];
                //ciclo di lettura da input del vettore
                for(int i=0;i<n;i++)
                        v[i]=Console.readInt("Inserire elemento "+i);

                if (eSimmetrico(v))
                        System.out.println("Il vettore e'simmetrico e non ha bisogno
di essere invertito.");
                else{
                        //inversione del vettore
                        System.out.println("Il vettore non e' simmetrico.");
                        inverti(v);
                        stampa(v,"Il vettore invertito e': ");
                        int [] v2=invertiBis(v);
                        stampa(v2,"Il vettore riinvertito e'");
```

```
            }//else
      } //main
} //class
```

46.Copia gli elementi pari di un vettore in un altro vettore

```
public class copiaPariMetodi
{
      public static void stampa(final int[] v)
      {
            System.out.print("[  ");
            for (int i = 0; i < v.length; i++)
                  System.out.print(v[i]+"  ");
            System.out.println ("]");
      }

      public static int[] copiaPari(final int[] a)
      {
            //contiamo gli elementi pari
            int contaPari=0;
            for (int i = 0; i < a.length; i++)
                  if (a[i] %2==0) contaPari++;

            // creiamo un nuovo vettore di dimensione opportuna
            int pari[]=new int[contaPari];

            // copiamo i pari nel nuovo vettore
            int k=0;
            for (int i = 0; i < a.length; i++)
                  if (a[i]%2 ==0){
                        pari[k]= a[i];
                        k++;
                   }
            return pari;
      }

      public static void main (String args[])
        {
            int n = Console.readInt ("n= ");
            int a[] = new int[n];
            //leggiamo il vettore
            for (int i = 0; i < a.length; i++)
              a[i] = Console.readInt ("a["+i+"]=");
            //stampiano il vettore
            stampa(a);
            //copiamo i pari
            int altroVettore[];
            altroVettore=copiaPari(a);
            //stampiano il vettore dei pari
            stampa(altroVettore);
        }
}
```

Esercizi con le matrici

47. Stampe di elementi su matrici (triangoli, righe, colonne, diagonali)

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

            int n=Console.readInt("n= ");
            int A[][] = new int[n][n];

            int i,j;
            //legge A
            for (i=0;i<n;i++)
```

```
                for(j=0;j<n;j++)
                        A[i][j]=Console.readInt("A["+i+","+j+"]: ");

        //stampa matrice
                System.out.println("Matrice letta:");
        for (i = 0; i < n; i++){
                        for (j = 0; j < n; j++)
                        System.out.print("A["+i+","+j+"]="+A[i][j]+"  ");
                System.out.println ();
                }

        //Scrive Triangolo inferiore
                System.out.println("Triangolo inferiore:");
                for (i = 0; i < n; i++){
                        for (j = 0; j <= i; j++)
                                System.out.print(A[i][j]+" ");
                        System.out.println ();
                }

        //Scrive Triangolo superiore
                System.out.println("Triangolo superiore:");
                for (i = 0; i < n; i++){
                        for (j = i; j < n; j++)
                                System.out.print(A[i][j]+" ");
                        System.out.println ();
                        }

        //Scrive Matrice trasposta
        System.out.println("Trasposta:");
        for (i = 0; i < n; i++){
                        for (j = 0; j < n; j++)
                                System.out.print (A[j][i]+" ");
                        System.out.println ();
                }

                //Scrive gli elementi di una riga
                int riga=Console.readInt("Dammi una riga: ");
        System.out.println("Elementi sulla riga n."+riga+ " :");
        for (i = 0; i < n; i++)
                        System.out.print (A[riga][i]+" ");
                System.out.println ();

                //Scrive gli elementi di una colonna
                int colonna=Console.readInt("Dammi una colonna: ");
        System.out.println("Elementi sulla colonna n."+colonna+ " :");
        for (i = 0; i < n; i++)
                        System.out.println (A[i][colonna]);

        //Scrive elementi sulla diagonale principale
        System.out.println("Diagonale principale:");
        for (i = 0; i < n; i++)
                        System.out.print (A[i][i]+" ");
                System.out.println ();


        //Scrive elementi sulla diagonale principale
                System.out.println("Diagonale secondaria:");
                for (i = 0; i < n; i++)
                        System.out.print (A[i][n-1-i]+" ");
                System.out.println ();
        }
}
```

48. Copia la zeta in un vettore

```
   /*
      copia dalla fine all'inizio gli elementi della matrice che si
      trovano sulla zeta formata dalla prima riga,
      diagonale secondaria e ultima riga
   */
   public class CopiaZeta{
      public static void main (String args[]){
                  int n=Console.readInt("n= ");
                  int A[][] = new int[n][n];

                  int i,j;
                  //legge A
                  for (i=0;i<n;i++)
                        for(j=0;j<n;j++)
                              A[i][j]=Console.readInt("A["+i+","+j+"]: ");

                  //stampa matrice
                  System.out.println("Matrice letta:");
                  for (i = 0; i < n; i++){
                        for (j = 0; j < n; j++)
                        System.out.print("A["+i+","+j+"]="+A[i][j]+"  ");
                  System.out.println ();
                  }
                  int v[]=new int[A.length*3-2];
                  int k=0;
                  //Copia gli elementi dell'ultima riga da destra a sinistra
                  for (j = n-1; j >=0; j--){
                        v[k]=A[n-1][j];
                        k++;
                  }
                  //copia elementi della diagonale secondaria da sotto a sopra
                  for (i = n-2; i >=1; i--){
                        v[k]=A[i][n-1-i];
                        k++;
                  }
                  //Copia gli elementi della prima riga destra a sinistra
                  for (j = n-1; j >=0; j--){
                        v[k]=A[0][j];
                        k++;
                  }
                  // stampa vettore
                  for(i=0;i<v.length;i++)
                        System.out.print("V["+i+"]="+v[i]+"  ");
      }
}
```

49. Date in input due matrici A e B, entrambe di dimensioni 3x4 calcolare: C=A*BT+D dove D è ottenuta estraendo le prime 3 righe e colonne di A.

```
public class opMatrici{
      public static void main (String args[]){
            final int n=3;
            final int m=4;
            int A[][] = new int[n][m];
            int B[][] = new int[n][m];
            int D[][] = new int[n][n];
            int C[][] = new int[n][n];
            int i,j;
            //legge A
            for (i=0;i<n;i++)
                  for(j=0;j<m;j++)
                        A[i][j]=Console.readInt("A["+i+","+j+"]: ");
```

```
            //legge B
            for (i=0;i<n;i++)
                  for(j=0;j<m;j++)
                        B[i][j]=Console.readInt("B["+i+","+j+"]: ");

            //estrae  D
            for (i=0;i<n;i++)
                  for(j=0;j<n;j++)
                        D[i][j]=A[i][j];
            //calcola trasposta di B
            int BT[][] = new int[m][n];
            for (i = 0; i < m; i++)
                        for (j = 0; j < n; j++)
                        BT[i][j] = B[j][i];
                  //moltiplica A*BT
            int P[][] = new int[n][n];
            for (i = 0; i < n; i++)
                        for (j = 0; j < n; j++){
                        int somma = 0;
                        for (int k = 0; k < m; k++)
                                    somma += A[i][k] * BT[k][j];
                        P[i][j] = somma;
                        }
            //calcola C
            for (i = 0; i < n; i++)
                        for (j = 0; j < n; j++)
                        C[i][j] = P[i][j] + D[i][j];
            //Scrive trasposta
            System.out.println("Trasposta:");
            for (i = 0; i < m; i++){
                        for (j = 0; j < n; j++)
                              System.out.print (BT[i][j]+" ");
                        System.out.println ();
            }
            //Scrive prodotto
            System.out.println("Prodotto:");
            for (i = 0; i < n; i++){
                        for (j = 0; j < n; j++)
                              System.out.print (P[i][j]+" ");
                        System.out.println ();
            }
            //Scrive C
            System.out.println("Risultato:");
            for (i = 0; i < n; i++){
                        for (j = 0; j < n; j++)
                              System.out.print(C[i][j]+" ");
                        System.out.println ();
            }
      }
  }
```

50. Controlla se almeno una riga di una matrice A contiene solo zeri

```
   /* Controlla se almeno una riga di una matrice A contiene solo zeri
   Traccia 29-aprile 2005 traccia C esercizio 3
   */
   public class controllaRighe{
      public static void main (String args[]){
                  int A[][] = {{2,3,0,2,1},
                                    {0,0,0,0,0},
                                    {1,3,0,0,3},
                                    {1,-1,6,-2,0},
                                    {3,2,4,0,1}
```

```
                                    };
                  int i,j;
                  int n=A.length,m=A[0].length;
                  //stampa matrice
                  System.out.println("Matrice letta:");
                  for (i = 0; i < n; i++){
                        for (j = 0; j < m; j++)
                        System.out.print("A["+i+","+j+"]="+A[i][j]+"  ");
                  System.out.println ();
                  }
               // fuori ho almeno uno, dentro ho un tutti
               //inizio almeno uno
               boolean trovataRiga=false;
               for(i=0;i<n;i++){
                        // inizio tutti
                        boolean tuttiZero=true;
                        for (j=0;j<m;j++)
                              if(A[i][j]!=0){tuttiZero=false;break;}
                        // fine tutti
                        if (tuttiZero) {trovataRiga=true;break;}
                  }
                  //fine almeno uno
                  if(trovataRiga)
                        System.out.println("La matrice contiene almeno una riga
   di tutti zeri");
     }
    }
```

51. Realizza alcune operazioni di base su matrici

```
public class MatriciOperazioni {
/* lettura di una matrice r x c*/
      public static int[][] letturaMatrice(int r, int c){
            int [][] matrice=new int [r][c];
            for (int i=0;i<r;i++)
                  for(int j=0;j<c;j++)
                        matrice[i][j]=Console.readInt("Elemento ("+i+","+j+")
");
            return matrice;
      }
      /* stampa di una matrice di qualsiasi dimensione*/
      public static void stampaMatrice(int [][] matrice){
            for (int i=0;i<matrice.length;i++){
                  for(int j=0;j<matrice[0].length;j++)
                        System.out.print(matrice[i][j]+" ");
                  System.out.println("");
            }
      }
      /* Calcola la somma degli elementi dell'array di array a. */
      public static int sommaDiretta(int[][] a) {
            int s=0; // somma degli elementi di a
            for (int i=0; i<a.length; i++)
            for (int j=0; j<a[0].length; j++)
                  s += a[i][j];
            return s;
      }
      /* Calcola la somma degli elementi dell'array a. */
      public static int somma(int[] a) {
            int s=0; // somma degli elementi di a
            for (int i=0; i<a.length; i++)
                  s += a[i];
            return s;
      }
```

```
	/* Calcola la somma degli elementi dell'array di array a. */
	public static int somma(int[][] a) {
		int s=0; // somma degli elementi di a
		for (int i=0; i<a.length; i++)
			s += somma(a[i]);
		return s;
	}
	/* Calcola la somma di due matrici di dimensioni identiche*/
	public static int[][] sommaMatrici(int[][] matrice1,int[][] matrice2){
		int n=matrice1.length;
		int m=matrice1[0].length;
		int[][] risultato=new int[n][m];
		for (int i=0;i<n;i++)
			for (int j=0;j<m;j++)
				risultato[i][j]=matrice1[i][j]+matrice2[i][j];
		return risultato;
	}
	public static void main(String[] args) {
		int [][] m1=letturaMatrice(2,2);
		System.out.println("Somma diretta degli elementi della prima
matrice"+sommaDiretta(m1));
		System.out.println("Somma per righe degli elementi della prima
matrice"+somma(m1));
		System.out.println();
		int [][] m2=letturaMatrice(2,2);
		System.out.println("Matrice 1");
		stampaMatrice(m1);
		System.out.println("Matrice 2");
		stampaMatrice(m2);
		int [][] somma=sommaMatrici(m1,m2);
		System.out.println("Matrice Somma");
		stampaMatrice(somma);
	}//main
}//class
```

52. Esercizio 3 (Esame di Fondamenti di Informatica - Data: 18 Aprile 2002– Traccia A)

Si realizzi una classe Matrice per rappresentare matrici di interi e si implementi i seguenti metodi:

- un metodo *leggi* che effettui la lettura da input degli elementi di una matrice; il metodo restituisce la matrice letta.
- un metodo *stampa* che effettui la stampa su output degli elementi di una matrice formattati per righe e colonne;
- un metodo *strisce* che verifichi se una matrice è **a strisce orizzontali** considerando la definizione sotto riportata;
- un metodo *sumax* che calcoli il **valore massimo tra le somme delle righe** di una matrice a strisce orizzontali;
- il metodo *main* che utilizzi i metodi precedenti per leggere una matrice, verifica se è a strisce orizzontali e in questo caso stampa la matrice e il massimo tra le somme delle righe.

Definizione: Sia **A** una matrice di interi; **A** è detta **a strisce orizzontali** se le righe di indice **dispari** presentano valori **tutti nulli**, mentre le **altre** righe presentano valori **tutti non nulli.**

```
public class Matrice18Aprile2002
	{ public static int[][] leggi ()
		{
		int n = Console.readInt ("numero di righe: ");
		int m = Console.readInt ("numero di colonne: ");
		int mat[][] = new int[n][m];
		for (int i = 0; i < mat.length; i++)
		for (int j = 0; j < mat[0].length; j++)
		mat[i][j] = Console.readInt ("M["+i+"]["+j+"]=");
		return mat;
```

```
	}
	public static void stampa (int mat[][])
	{
		for (int i = 0; i < mat.length; i++)
			{
			for (int j = 0; j < mat[0].length; j++)
			System.out.print (mat[i][j]+"\t");
			System.out.println();
		}
	}
	public static boolean strisce (int mat[][])
	{
		boolean verificato = true;
		for (int i = 0; i < mat.length && verificato; i++)
			for (int j = 0; j < mat[0].length && verificato; j++)
				if ((i%2==0 && mat[i][j]==0)||(i%2!=0 && mat[i][j]!=0))
					verificato = false;
		return verificato;
	}
	public static int sumax (int mat[][])
	{
		/* si assume che la matrice sia a strisce
		orizzontali secondo la definizione */
		int max = 0;
		for (int i = 0; i < mat.length; i+=2)
			{
			int sommaRiga = mat[i][0];
			for (int j = 1; j < mat[0].length; j++)
				sommaRiga+=mat[i][j];
			if (sommaRiga > max)
			max = sommaRiga;
		}
		return max;
	}
	public static void main (String args[])
		{
		int m[][] = leggi();
		if (strisce(m))
			{
			System.out.println ("Stampa della matrice:");
			stampa (m);
		System.out.println ("Massimo tra le somme delle righe: "+sumax(m));
			}
	}
}
```

53. Esercizio 2 - Da una traccia d’esame

Si progetti una classe Esercizio2 per elaborare matrici di interi.

In particolare bisogna estrarre da una matrice rettangolare M di dimensione 4 x 12 tre sottomatrici quadrate S, T, V di dimensione 4 x 4:

M = | S | T | V |

e calcolare la matrice R = V * St + T , in cui St è la matrice trasposta di S.

Si implementino quindi (almeno) i seguenti metodi:

1. leggi che effettua la lettura da input degli elementi di una matrice m ricevuta parametricamente;
2. scrivi che scrive su output il contenuto di una matrice m passata come parametro
3. moltiplica che riceve due matrici quadrate m1 ed m2, supposte della stessa dimensione, e restituisce la matrice prodotto;
4. addiziona che riceve due matrici quadrate m1 ed m2, supposte della stessa dimensione, e restituisce la matrice somma;
5. trasponi che riceve una matrice m e restituisce la sua trasposta;

6. estrai che riceve la matrice rettangolare m ed un intero i e restituisce una delle matrici quadrate S, T, V a seconda del valore assunto da i (per esempio per i = 0 la matrice estratta sarà la S, per i = 1 la matrice estratta sarà la T e per i = 2 la matrice estratta sarà la V).
7. main che, utilizzando i metodi precedenti, provvede a effettuare l'estrazione di S, T e V, il calcolo di R e, infine, la visualizzazione del suo contenuto.

```
public class EsercizioOperMatrici
{
	public static void main (String args[])
	{
		int M[][] = new int[4][12];
		leggi (M);
		int S[][] = estrai (M,0);
		int T[][] = estrai (M,1);
		int V[][] = estrai (M,2);
		int St[][] = trasponi (S);
		int P[][] = moltiplica (V,St);
		int R[][] = addiziona (P,T);
		scrivi (R);
	}
	public static void leggi (int m[][])
	{
		for (int i = 0; i < m.length; i++)
			for (int j = 0; j < m[0].length; j++)
				m[i][j] = Console.readInt ("m["+i+"]["+j+"]=");
	}
	public static void scrivi (int m[][])
	{
		for (int i = 0; i < m.length; i++)
			{
				for (int j = 0; j < m[0].length; j++)
					System.out.print (m[i][j]+" ");
				System.out.println ();
			}
	}
	public static int[][] moltiplica (int m1[][], int m2[][])
	{
		int dim = m1.length;
		int p[][] = new int[dim][dim];
		for (int i = 0; i < dim; i++)
			for (int j = 0; j < dim; j++)
			{
				int somma = 0;
				for (int k = 0; k < dim; k++)
					somma += m1[i][k] * m2[k][j];
				p[i][j] = somma;
			}
		return p;
	}
	public static int[][] addiziona (int m1[][], int m2[][])
	{
		int dim = m1.length;
		int s[][] = new int[dim][dim];
		for (int i = 0; i < dim; i++)
			for (int j = 0; j < dim; j++)
				s[i][j] = m1[i][j] + m2[i][j];
		return s;
	}
	public static int[][] trasponi(int m[][])
	{
		int t[][] = new int[m[0].length][m.length];
		for (int i = 0; i < t.length; i++)
			for (int j = 0; j < t[0].length; j++)
```

```
                        t[i][j] = m[j][i];
            return t;
      }
      public static int[][] estrai(int m[][], int i)
      {
      int nRighe = m.length;
            int nColonne = m[0].length/3;
            int e[][] = new int[nRighe][nColonne];
            for (int k = 0; k < nRighe; k++)
                  for (int h = 0; h < nColonne; h++)
                        e[k][h] = m[k][h+nColonne*i];
            return e;
      }
}
```

54. Prova scritta dell'esame di Fondamenti di Informatica Data: 6 aprile 2004 TRACCIA A - Esercizio 1 - Si consideri il seguente programma:

```
public class Esercizio1A {
      public static int metodoA(int x[]) {
            int c = 0;
            for (int i = 0; i < x.length/2; i++) {
                  int j = i + x.length/2;
                  if (( x[i] % x[j] == 0 ) || ( x[j] % x[i] == 0 )) {
                        c++;
                        System.out.println (x[i]);
                  }
            }
            return c;
      }
      public static void main(String args[]) {
            int v[] = {3, 4, 8, 5, 2, 16};
            int k = metodoA(v);
            System.out.println ("k = "+k);
      }
}
   /*
   Si descriva sinteticamente la funzione svolta dal metodo metodoA e,
   in particolare, si mostri l'esecuzione e cosa viene stampato nel caso
   in esempio, in cui v = {3, 4, 8, 5, 2, 16}.
   */
```

55. Scrivere un metodo verificavett che riceve un vettore V di interi di lunghezza n, e restituisca il valore true se ogni elemento di V è maggiore o uguale della somma dei due valori precedenti (se esistono), false altrimenti. Ad esempio, se V=[1, 2, 4, 7, 13, 20, 35], il metodo restituirà true.

```
public static boolean verificavett (int[] V)
{
      boolean verifica = true;
      for (int i = 2; i < V.length && verifica; i++)
            if (V[i] < V[i-1]+V[i-2])
                  verifica = false;
      return verifica;
}
```

56. Traccia B di un compito

```
public class SoluzioneTracciaB
{
      /* Esercizio 1
                  Il metodo accetta come input un vettore di boolean;
            controlla che la lunghezza del vettore non sia dispari o che il
```

```
      vettore non contenga solo 2 elementi e nal caso in cui una di queste
      due condizioni sia vera restituisce -1.
            In caso contrario inizializza un contatore k ed una
      variabile risultato p, entrambi di tipo intero. Inizia un ciclo di
      while che scorre il vettore fino alla sua metà ed all'interno del
      ciclo controlla che per le posizioni pari il valore contenuto nel
      vettore sia uguale al suo simmetrico e nel caso questo sia vero
      incrementa la variabile p. Nell'esempio presentato il metodo
      restituisce un valore pari a 2.
*/
public static int metodo1(boolean [] vett)
{
      if ((vett.length % 2 != 0) || (vett.length < 2))
      {
            return -1;
      }
      else
      {
            int k = 0, p = 0;
            while (k < vett.length/2)
            {
                  if ((vett[k] == vett[vett.length-k-1]) && (k % 2==0))
                  {
                        p++;
                  }
                  k++;
            }
            return p;
      }
}
/* Esercizio 2
            Scrivere un metodo generaVettore che riceve come parametri di
      input due vettori di interi e restituisce un vettore di interi che
      contiene tutti gli elementi del primo vettore che sono strettamente
      maggiori di tutti gli elementi del secondo vettore a partire
      dalla posizione successiva nel secondo vettore e scorrendone poi gli
      elementi successivi.
*/
public static int [] generaVettore(int [] v1, int[] v2)
{
      int conta=0;
      for (int i = 0; i < v1.length - 1; i++)
      {
            boolean inserisci = true;
            for (int j = i + 1; j < v1.length ; j++)
            {
                  if (v1[i] <= v2[j])
                  {
                        inserisci = false;
                        break;
                  }
            }
            if (inserisci)conta++;
      }
      int [] risultato = new int [conta];
      int k=0;
            for (int i = 0; i < v1.length - 1; i++)
            {
                  boolean inserisci = true;
                  for (int j = i + 1; j < v1.length ; j++)
                  {
                        if (v1[i] <= v2[j])
                        {
```

```
                        inserisci = false;
                        break;
                    }
                }
                if (inserisci){risultato[k]=v1[i]; k++;}
        }
        return risultato;
}

/* Esercizio 3_1
        Scrivere un metodo estraiArray che riceve una matrice quadrata
    di interi e restituisce un vettore contenente una parte degli
    elementi della diagonale secondaria a partire dalla posizione in
    alto a destra fino al primo elemento che abbia valore 0.
*/
public static int[] estraiArray(int [][] A)
{
        int n=A.length;
        int conta=0;
        for (int i = n-1; i >=0  ; i --)
        {
            if(A[i][n-1-i]==0)break;
            conta++;
        }
        int [] risultato=new int [conta];
        int k=0;
        for (int i = n-1; i >=0  ; i --)
        {
            if(A[i][n-1-i]==0)break;
            risultato[k]=A[i][n-1-i];
            k++;
        }
        return risultato;
}

/* Esercizio 3_2
        Scrivere un metodo estraiMatrice che riceve in input una
    matrice quadrata ed un intero k e restituisce la sottomatrice
    inferiore destra della matrice data a partire dalla posizione(k, k).
*/
public static int[][] estraiMatrice(int [][] A, int k)
{
        int [][] risultato = new int [A.length - k][A.length - k];
        for (int i = 0; i < A.length - k; i++)
        {
            for (int j = 0; j < A.length - k; j++)
            {
                risultato[i][j] = A[k + i][k + j];
            }
        }
        return risultato;
}
/* Esercizio 3_3
        Scrivere un metodo controlla che riceve in input una matrice
    quadrata e restituisce un booleano. Il valore restituito sarà true
    se nessuno degli elementi in una colonna pari ha valore pari. False
    altrimenti.
*/
public static boolean controlla(int [][] A)
{
        for (int i = 1; i < A.length ; i += 2)
        {
            for (int j = 0; j < A.length ; j++)
```

```
                {
                      if (A[i][j] % 2 == 0)
                      {
                            return false;
                      }
                }
          }
          return true;
    }
    public static void stampa(int [][] M)
    {
          for (int i = 0; i < M.length ; i ++)
          {
                for (int j = 0; j < M[0].length ; j++)
                {
                      System.out.print(M[i][j] + " ");
                }
                System.out.println();
          }
    }
    /* Scrivere un main che utilizzi tutti i metodi sopra esposti. */
    public static void main(String [] args)
    {
          boolean [] vett = {false, false, true, false, false, true, false,
false};
          int [][] A ={{5, 2, 11, 9, 12},{9, 15, 10, 7, 1},{14, 21, 19, 2, 2},
                       {8, 0, 13, 9, 7}, {10, 17, 12, 10, 3}};
          System.out.println("Matrice A: ");
          stampa(A);
          /* int dimensione;
          dimensione = Console.readInt("Dimensione della matrice quadrata: ");
          for (int i = 0; i < dimensione; i++)
          {
                for (int j = 0; j < dimensione; j++)
                {
                      A[i][j] = Console.readInt("Valore in posizione (" + i +
", " + j + "): ");
                }
          }*/
          int n = metodo1(vett);
          System.out.println("Risultato chiamata metodo1 (esercizio 1): "+ n);
          int [] v1 = {7, 10, 9, 2};
          int [] v2 = {2, 8, 4, 1};
          int [] v3;
          v3 = generaVettore(v1, v2);
          System.out.print("Risultato chiamata generaVettore(v1, v2): ");
          for (int i = 0; i < v3.length ; i ++)
          {
                System.out.print(v3[i] + " ");
          }
          System.out.println();
          System.out.print("Risultato chiamata estraiArray(A): ");
          v3 = estraiArray(A);
          for (int i = 0; i < v3.length ; i ++)
          {
                System.out.print(v3[i] + " ");
          }
          System.out.println();
          System.out.println("Risultato chiamata estraiMatrice(A, 2): ");
          stampa(estraiMatrice(A, 2));
          System.out.println("Risultato chiamata controlla(A): "+
controlla(A));
    }
```

```
};
```

57. Traccia D di un compito

```
public class SoluzioneTracciaD
{
	/* Esercizio 1
		Il metodo accetta come input un vettore di interi ed un intero h;
		viene quindi eseguito un ciclo while che scorre tutti gli elementi
		del vettore (tramite l'indice i). Per ciascun elemento del vettore
	   si controlla che sia divisibile per il parametro h. Se questo è vero
	   nella posizione attuale del vettore viene salvato il risultato della
	   divisione del valore attuale del vettore per il parametro h. Se invece
	   è falso nella posizione attuale del vettore si salva il valore h. Per
	   ogni passo del ciclo si stampa il valore del vettore appena modificato.
	*/
	public static void metodo2(int[] vt, int h)
	{
		int i = 0;
		while (i < vt.length)
		{
			if (vt[i] % h == 0)
			{
				vt[i] = vt[i] / h;
			}
			else
			{
				vt[i] = h;
			}
			System.out.println(vt[i]);
			i = i + 1;
		}
	}
	/* Esercizio 2
			Scrivere un metodo selezionaVettore che riceve come parametri
	   di ingresso due vettori e restituisce un vettore che contiene tutti
	   gli elementi del primo vettore il cui valore è strettamente
	   inferiore alla somma degli elementi del secondo vettore a parire
	   dalla posizione successiva a quella data.
	*/
	public static int [] selezionaVettore(int [] v1, int[] v2)
	{
		int [] temp = new int [v1.length];// dimensione massima
		int k=0;
		for (int i = 0; i < v1.length; i++)
		{
			int somma = 0;
			for (int j = i + 1; j < v1.length ; j++)
				somma += v2[j];
			if (v1[i] < somma) {
					temp[k]=v1[i];
					k++;
				}
		}
		int []risultato =new int [k];
		for (k=0;k<risultato.length;k++) risultato[k]=temp[k];
		return risultato;
	}
	/* Esercizio 3_1
			Scrivere un metodo generaArray che riceve una matrice quadrata
	   di interi e restituisce in un vettore gli elementi della diagonale
	   principale a partire dall'ultimo elemento fino al primo elemento
```

```
	pari. In ogni caso restituire almeno il primo elemento, anche se la
	condizione non è verificata.
*/
public static int[] generaArray(int [][] A)
{
	int n=A.length;
	int conta=0;
	for (int i = n-1; i >=0  ; i --)
	{
		if(A[i][i]%2==0)break;
		conta++;
	}
	if (conta==0) {int []ris= new int[1];
				ris[0]=A[n-1][n-1];
				return ris;
				}
	int [] risultato=new int [conta];
	int k=0;
	for (int i = n-1; i >=0  ; i --)
	{
		if(A[i][i]==0)break;
		risultato[k]=A[i][n-1-i];
		k++;
	}
	return risultato;
}
/* Esercizio 3_2
		Scrivere un metodo generaMatrice che riceva in ingresso una
	matrice quadrata e restituisca una sottomatrice data da tutti gli
	elementi di riga e colonna dispari della matrice data.
*/
public static int[][] generaMatrice(int [][] M)
{
	int [][] risultato = new int [M.length / 2 + 1][M.length / 2 + 1];
	for (int i = 0; i < risultato.length; i++)
	{
		for (int j = 0; j < risultato.length; j++)
		{
			risultato[i][j] = M[2*i][2*j];
		}
	}
	return risultato;
}
/* Esercizio 3_3
		Scrivere un metodo valuta che riceve in ingresso una matrice
	quadrata restituisce true se tutti gli elementi posti sulle righe
	dispari sono diversi da 0, false altrimenti.
*/
public static boolean valuta(int [][] M)
{
	for (int i = 0; i < M.length ; i += 2)
	{
		for (int j = 0; j < M.length ; j++)
		{
			if (M[i][j] == 0)
			{
				return false;
			}
		}
	}
	return true;
}
public static void stampa(int [][] M)
```

```
	{
		for (int i = 0; i < M.length ; i ++)
		{
			for (int j = 0; j < M[0].length ; j++)
			{
				System.out.print(M[i][j] + " ");
			}
			System.out.println();
		}
	}
	public static void main(String [] args)
	{
		int[] vt = {15, 12, 4, 10, 27, 5};
		int[][] A = {{4, 1, 10, 8, 11},
			     {10, 16, 11, 8, 2},
			     {13, 20, 7, 1, 1},
			     {10, 2, 15, 11, 9},
			     {13, 20, 0, 10, 3}};
		System.out.println("Matrice A: ");
		stampa(A);
		/*int dimensione;
		dimensione = Console.readInt("Dimensione della matrice quadrata: ");
		for (int i = 0; i < dimensione; i++)
		{
			for (int j = 0; j < dimensione; j++)
			{
				A[i][j] = Console.readInt("Valore in posizione (" + i +
", " + j + "): ");
			}
		}
		*/
		System.out.println("Risultato chiamata metodo2 (esercizio 1): ");
		metodo2(vt, 5);
		int [] v1 = {7, 10, 3, 2};
		int [] v2 = {2, 8, 1, 4};
		int [] v3;
		v3 = selezionaVettore(v1, v2);
		System.out.print("Risultato chiamata selezionaVettore(v1, v2): ");
		for (int i = 0; i < v3.length ; i ++)
		{
			System.out.print(v3[i] + " ");
		}
		System.out.println();
		System.out.print("Risultato chiamata generaArray(A): ");
		v3 = generaArray(A);
		for (int i = 0; i < v3.length ; i ++)
		{
			System.out.print(v3[i] + " ");
		}
		System.out.println();
		System.out.println("Risultato chiamata generaMatrice(A): ");
		stampa(generaMatrice(A));
		System.out.println("Risultato chiamata valuta(A): " + valuta(A));
	}
};
```

58. TRACCIA B- 18 marzo 2004

Esercizio 2

Si scriva un metodo *creaVettore* che riceve in ingresso un array **V** di numeri interi e restituisce un vettore di interi **S** di dimensione pari alla dimensione di V, in cui l'elemento S[i] è la somma degli elementi V[i], V[i+1],…, V[V.length-1] **con valore positivo**.

Ad esempio, se **V** = [20, -11, 4, 7, -17, 5] allora **S** = [36, 16, 16, 12, 5, 5].

Esercizio 3

Si scriva una classe *ElaboraMatrici* contenente i seguenti metodi:

1. Un metodo **duplicati** che riceve una matrice di interi **M** e restituisce **true** se M presenta elementi duplicati, **false** altrimenti;
2. Un metodo **seleziona** che riceve una matrice quadrata di interi **M** e due numeri **r** e **c** e restituisce una matrice contenente **due** righe, ottenute rispettivamente invertendo la **riga r** di **M** e la **colonna c** di **M**. Si assuma che i valori di r e c siano validi per le dimensioni di M;
3. Un metodo **ripetuti** che riceve in input una matrice **M** e un intero **r** e restituisca un array di interi in cui siano memorizzati i valori presenti **esattamente r volte** in M;
4. Un metodo main che utilizzi i metodi precedenti.

Esempio: sia M la matrice riportata a destra, allora:

1. duplicati(M) → true
2. seleziona(M,1,3) →

16	21	9	17	3
-9	22	51	21	22

3. ripetuti(M,2) →

22	8	-3

M =

13	7	14	22	8
3	17	9	21	16
33	27	44	51	17
8	1	-3	22	4
2	0	17	-9	-3

```
public class SoluzioneTracciaB18032004
{
	// Esercizio 1
	public static int verifica(int [] x)
	{
		int r = 0;
		for (int i = 0; i < x.length/2 ; i++)
		{
			if (x[i] == x[x.length - i - 1])
			{
				r = r + (2*x[i]);
			}
		}
		return r;
	}
	/* Esercizio 2
			Scrivere un metodo creaVettore che riceve in ingresso un
		vettore di interi e restituisce un altro vettore di interi	che
		contiene in ogni posizione la somma di tutti gli elementi nelle
		posizioni successive del vettore in ingresso che siano strattamente
		positivi.
	*/
	public static int [] creaVettore(int [] V)
	{
		int [] risultato = new int [V.length];
		for (int i = 0; i < V.length ; i ++)
		{
			risultato[i]=0;
			for (int j = i; j < V.length ; j++)
			{
				if (V[j] > 0)
				{
```

```
                        risultato[i] += V[j];
                  }
            }
      }
      return risultato;
}
/* Esercizio 3_1
      Scrivere un metodo che riceva in ingresso una matrice di
      interi e restituisca true se nella matrice ci sono duplicati, false
      altrimenti.
*/
public static boolean duplicati(int [][] M)
{
      for (int i = 0; i < M.length ; i ++)
      {
            for (int j = 0; j < M[0].length ; j++)
            {
                  for (int k = i; k < M.length ; k++)
                  {
                        int z = j + 1;
                        if (k > i)
                        {
                              z = 0;
                        }
                        for (; z < M[0].length ; z++)
                        {
                              if ( M[i][j] == M[k][z])
                              {
                                    return true;
                              }
                        }
                  }
            }
      }
      return false;
}
/* Esercizio 3_2
      Scrivere un metodo seleziona che riceve in ingresso una
      matrice quadrata e due interi e restituisce una matrice con due
      righe date dalla riga selezionata col primo intero e dalla colonna
      selezionata col secondo ma selezionate in ordine inverso
      (dall'ultimo elemento al primo).
*/
public static int[][] seleziona(int [][] M, int r, int c)
{
      int [][] risultato = new int [2][M.length];
      for (int i = M.length - 1; i >= 0 ; i--)
      {
            risultato[0][M.length - 1 - i] = M[r][i];
            risultato[1][M.length - 1 - i] = M[i][c];
      }
      return risultato;
}
/* Esercizio 3_3
      Scrivere un metodo ripetuti che riceve in ingresso una matrice
      quadrata di interi ed un intero r. In uscita deve dare un vettore
      contenenete tutti gli elementi della matrice ripetuti un numero di
      volte pari all'intero dato.
*/
public static int[] ripetuti(int [][] M, int r)
{
      int n=M.length;
      int conta;int index=0;
```

```
	int [] temp = new int[n*n];
	for (int i = 0; i < n; i ++)
		for (int j = 0; j < n ; j++)
		{
			conta = 0;
			for (int k = 0; k < n ; k++)
				for (int z = 0; z < n; z++)
					if ( M[i][j] == M[k][z]) conta++;
			if (conta==r)
				{ temp[index]=M[i][j];
					index++;}
		}
	int [] risultato = new int[index];
	for(int i=0;i<risultato.length;i++)risultato[i]=temp[i];
	return risultato;
}
public static void stampa(int [][] M)
{
	for (int i = 0; i < M.length ; i ++)
	{
		for (int j = 0; j < M[0].length ; j++)
		{
			System.out.print(M[i][j] + " ");
		}
		System.out.println();
	}
}
public static void main(String [] args)
{
	int [][] M2;
	int [] V = {9, 2, 3, 7, 2, 30};
	int [][] M = {	{13, 7, 14, 22, 8},
				{3, 17, 9, 21, 16},
				{33, 27, 44, 51, 17},
				{8, 1, -3, 22, 4},
				{2, 0, 17, -9, -3}};
	int [][] M1 = {	{13, 7, 14, 22, 94},
				{3, 99, 9, 21, 16},
				{33, 27, 44, 51, 98},
				{8, 1, -3, 95, 4},
				{2, 0, 97, -9, 91}};
	int n = verifica(V);
	System.out.println("Verifica(V) = " + n);
	int [] C = {20, -11, 4, 7, -17, 5};
	C = creaVettore(C);
	System.out.print("creaVettore(C) = ");
	for (int i = 0; i < C.length ; i ++)
	{
		System.out.print(C[i] + " ");
	}
	System.out.println();
	System.out.println("Matrice M = ");
	stampa(M);
	System.out.println("duplicati(M) = " + duplicati(M));
	System.out.println("selezione(M, 1, 3) = ");
	stampa(seleziona(M, 1, 3));
	int [] D = ripetuti(M, 2);
	System.out.println("ripetuti(M, 2) = ");
	for (int i = 0; i < D.length ; i ++)
	{
		System.out.print(D[i] + " ");
	}
}
```

```
};
```

59. Esame di Fondamenti di Informatica Data 22 Giugno 2004 Traccia A

ESERCIZIO 2
Scrivere un metodo metodoA2 che riceva in input due vettori V1 e V2 contenenti interi positivi
e di uguale dimensione e restituisca in output un vettore W di interi contenente ogni elemento di V1 che

1. è di valore pari;
2. è multiplo degli elementi di V2 che occupano la stessa posizione e le posizioni successive.

Qualora nessuno tra gli elementi di V1 soddisfi le condizioni indicate, il vettore W sia un vettore contenente un solo elemento di valore 0.
Esempio: Se V1=[2, 11, 30, 7, 12, 6] e V2=[1, 2, 3, 10, 5, 3], il metodo restituirà W = [30, 6] perché 30 è multiplo di 3, 10,5 e 3, 6 è multiplo di 3.

```
public class Metodo22Giugno2004 {
public static int [] metodoA2(int []v1,int []v2){
            int n=v1.length;
            int temp[]=new int[n];
            int k=0;
            for (int i=0;i<n;i++){

                  if(v1[i]%2==0){
                        // controlla multiplo
                        boolean eMultiplo=true;
                        for(int j=i;j<n && eMultiplo;j++)
                         {
                              if(v1[i] % v2[j]!=0 ) eMultiplo=false;
                              }


                        if(eMultiplo) {
                              temp[k]=v1[i];
                              k++;
                              }
                     } //if
            }//for
            if(k==0){
            //nessuno tra gli elementi di V1 soddisfi le condizioni indicate
                        int[] w=new int[1];
                        w[0]=0;return w;
                        }

            int []w= new int[k];
            for(int i=0; i<k;i++)
                  w[i]=temp[i];
            return w;
      } //metodoA2

public static void main(String [] args){
      int []v1={2, 11, 30, 7, 12, 6};
      int []v2={1, 2, 3, 10, 5, 3};
      int []w=metodoA2(v1,v2);
      for(int i=0;i<w.length;i++)
            System.out.println(w[i]);

      } //main
}//class
```

60. Esame di Fondamenti di Informatica Data 22 Giugno 2004 Traccia A Es.3

```
//Scrivere una classe Matrice contenente i seguenti metodi:
public class Matrice22Giugno04{
            /* un metodo cornice che riceva in input una matrice quadrata M di
      interi e restituisca in output un vettore A contenente gli elementi della
      cornice" più esterna di M (evidenziata in figura); in A, gli elementi
      siano ordinati secondo la lettura per righe di M;
      */
      public static  int[]cornice( int m [][]){
            int n=m.length;
            int []v=new int[n+n+n-2+n-2];

            int k=0;
            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){ //sulla cornice
                              v[k]=m[i][j];
                              k++;
                        } //if
                  } //for
            return v;
            } //cornice

      public static  int[]cornice1( int m [][]){ // metodo alternativo al
precedente
            int n=m.length;
            int []v=new int[n+n+n-2+n-2];

            int k=0;
            for(int j=0; j<n;j++){
                  v[k]=m[0][j];k++;
                  }
            for(int i=1;i<n-1;i++){
                  v[k]=m[i][0];k++;
                  v[k]=m[i][n-1];k++;
                  }
            for(int j=0; j<n;j++){
                  v[k]=m[n-1][j];k++;
                  }
            return v;
            }//cornice1

            /*Un metodo rigaDelMassimo che riceva in input una matrice quadrata
      M di interi e restituisca in output un vettore B contenente la prima riga
      in cui è presente l'elemento più grande di M;
      */
      public static int[] rigaDelMax( int m [][]){
            int n=m.length;
            int []risposta=new int[m[0].length];
            int max=m[0][0];int riga=0;
            for(int i=0; i<n;i++)
                  for(int j=0;j<n;j++){
                        if(m[i][j]>max) {
                                    max=m[i][j];
                                    riga=i;
                              };
                        }
            for(int j=0;j<n;j++)
                  risposta[j]=m[riga][j];
            return risposta;
            }
```

```
	/*Un metodo matriceFiltrata che riceva in input una matrice M di
	interi e due interi h e k    e restituisca in output una matrice C
	contenente tutti gli elementi di M aventi indice di riga compreso tra h e
	k ed indice di colonna dispari; in C, gli elementi siano ordinati secondo
	la lettura per righe di M;
	*/
	public static int[][] matriceFiltrata( int m [][],int h,int k){
		int nMax=m.length;
		int mMax=m[0].length;
		int cRighe=k-h+1;
		int cColonne=mMax/2;
		int [][] c=new int [cRighe][cColonne];
		for(int i=0; i<cRighe;i++){
			int riga=h+i;
			for(int j=0;j<cColonne;j++){
				int colonna=j*2+1;System.out.print("riga "+riga+"
colonna "+colonna+"elem "+m[riga][colonna]);
				c[i][j]=m[riga][colonna];
			}
		} //for
		return c;
	} //matriceFiltrata

	public static int[][] matriceFiltrata1( int m [][],int h,int k){
					// metodo alternativo al precedente
		int nMax=m.length;
		int mMax=m[0].length;
		int cRighe=k-h+1;
		int cColonne=mMax/2;
		int [][] c=new int [cRighe][cColonne];
		int colonna;
		int riga=0;
		for(int i=h; i<=k;i++){
			colonna=0;
			for(int j=1;j<mMax;j+=2){
				c[riga][colonna]=m[i][j];
				colonna++;
			}
			riga++;// cambia la riga
		}//for
		return c;
	} //matriceFiltrata1

	public static void stampa(final int [] vettore)
	{	System.out.print("[");
		for (int i = 0; i < vettore.length-1; i++)
			System.out.print(vettore[i] + ",");
		System.out.println(vettore[vettore.length-1] + "]");

	}
	public static void stampa(final int [][]matrice)
	{
		for(int i=0; i<matrice.length;i++){
		       for(int j=0;j<matrice[0].length;j++)
				System.out.print(matrice[i][j] + " ");
		System.out.println();
		}
	}
	/* un metodo main che utilizzi i metodi precedenti
	(dichiarando le variabili necessarie a tal fine).
	*/
	public static void main(String []args){
		int m[][]=
```

```
                        {{1, 2, 3, 4 ,5},
                        {16, 1, 0 ,2, 6},
                        {15, 2, 22, 4, 7},
                        {14,3,5, 6, 8},
                        {13,4, 7,10, 9}};
                stampa(m);
                int h=1,k=3;
                int [] vett=cornice(m);//int [] vett=cornice1(m);
                stampa(vett);
                vett=rigaDelMax(m);
                stampa(vett);
                int [][] m2=matriceFiltrata(m,h,k);
                //int [][] m2=matriceFiltrata1(m,h,k);
                stampa(m2);
                }
        }//class

/*
```

61. Scrivere una classe Matrice che verifichi se una matrice quadrata ha tutte le cornici la cui somma degli elementi è pari

```
public class MatriceConcentricaPari{
        //verifica se la k-esima cornice ha somma pari
        public static  boolean pari( int a [][],int k){
                int n=a.length;
                int somma=0;
                for(int i=k; i<n-k;i++)
                        for(int j=k;j<n-k;j++)
                                if(i==k || i==n-1-k || j==k || j==n-1-k)
                                        //sulla cornice k-esima
                                        somma+=a[i][j];
                System.out.println("somma cornice "+k+"-esima="+somma);
                return somma%2==0;
                } //pari

        public static  boolean pariConcentrica( int a [][]){
                int numeroCornici=a.length/2;
                for(int k=0; k<numeroCornici;k++)
                        if(!pari(a,k)) return false;
                if(a.length%2==0) return true;
                /*
                nel caso in cui la matrice ha n dispari resta da verificare
                l'elemento al centro della matrice
                */
                return a[numeroCornici][numeroCornici]%2==0;
                }
        public static void stampa(final int [][]matrice)
        {
                for(int i=0; i<matrice.length;i++){
                        for(int j=0;j<matrice[0].length;j++)
                                System.out.print(matrice[i][j] + " ");
                System.out.println();
                }
        }
        /* un metodo main che utilizzi i metodi precedenti
        (dichiarando le variabili necessarie a tal fine).
        */
        public static void main(String []args){
                int m[][]=
                        {{1, 2, 3, 4 ,2},
                        {6, 1, 0 ,2, 6},
                        {1, 3, 2, 4, 1},
                        {1, 8, 0, 6, 2},
```

```
                        {1, 2, 3, 4, 1}};
            stampa(m);
            if(pariConcentrica(m))
                  System.out.println("La matrice e' concentrica pari");
            else System.out.println("La matrice non e' concentrica pari");
            } //main
      } //class
```

62. Compito 7-7-2005 traccia A n.2

```
/* Si scriva un metodo sommaMinori che riceve in ingresso un vettore v1, e
restituisce un vettore v2 della stessa dimensione di v1. In particolare, v2[i]
conterrà la somma degli elementi di v1 aventi indice minore di i ed il cui
valore è minore o uguale a v1[i];
se non esiste nessun elemento in v1 che soddisfa la condizione, allora v2[i]
avrà valore 0. Si noti che v2[0] avrà pertanto sempre valore 0.
Ad esempio, se v1 = [ 5, 7, 8, 5, 9, 3, 6 ], il vettore restituito è v2 = [ 0,
5, 12, 5, 25, 0, 13]. */
public class Compito7_7_05_A2{
/* esercizio 2 */
public static int[] sommaMinori(int v1[]){
      int n=v1.length;
      int [] v2=new int [n];
      // azzeriamo il vettore
      for(int i=0;i<n; i++) v2[i]=0;
      //cambiamo selettivamente
      for(int i=1; i<n; i++){
            boolean trovato=false;// per almeno uno
            //  inzio somma
            int sum=0;
            for(int j=0;j<i;j++)
                  if(v1[j]<=v1[i]) {sum+=v1[j];trovato=true;}
            //    fine somma

            if(trovato) v2[i]=sum;
      }// for
      return v2;
}

/* altra risposta esercizio 2 */
public static int[] sommaMinori2(int v1[]){
      int n=v1.length;
      int [] v2=new int [n];
      // azzeriamo il primo elemento del vettore
      v2[0]=0;
      //cambiamo selettivamente
      for(int i=1; i<n; i++){
            //  inzio somma
            v2[i]=0;
            for(int j=0;j<i;j++)
                  if(v1[j]<=v1[i]) v2[i]+=v1[j];
            //    fine somma
      } // for
      return v2;
}

  public static void main (String args[]){
        int []vett1={5,7,8,5,9,3,6};
        int n=vett1.length;
        //---------------------
        int [] vett= sommaMinori(vett1);
```

```
        for(int i=0;i<n;i++)
            System.out.println(vett[i]);
        //---------------------
        vett= sommaMinori2(vett1);
        for(int i=0;i<n;i++)
            System.out.println(vett[i]);
}
}//  class
```

63. Compito del 7-7-2005 traccia A Esercizio 3

```
/* Si realizzi una classe Esercizio3A che contenga i seguenti metodi:*/
public class Compito7_7_05_A3{
/*
Un metodo sommaMultipli che riceve una matrice di interi A ed un intero k,
e restituisce un vettore V il cui i-esimo elemento contiene la somma degli
elementi multipli di k presenti sulla i-esima colonna di A.
*/
      public static int [] sommaMultipli(int [][] A, int k){
            int n=A.length, m=A[0].length;
            int []V=new int [m];

            for(int j=0;j<m;j++){
                  int sum=0;
                  for (int i=0; i<n;i++)
                        if(A[i][j]%k==0) sum+=A[i][j];
                  V[j]=sum;
                  }
            return V;
            }

/*   semplificato
            for(int j=0;j<m;j++){
                  V[j]=0;
                  for (int i=0; i<n;i++)
                        if(A[i][j]%k==0) V[j]+=A[i][j];
                  }
*/

/*Un metodo verificaMedia che riceve una matrice di interi A avente n righe (con
n pari)
e restituisce true se e solo se nessun elemento nelle prime n/2 righe di A
ha un valore maggiore della media degli elementi presenti nelle ultime n/2 righe
di A.
*/
public static boolean verificaMedia(int A[][]){
      int n=A.length,m=A[0].length;
      // facciamo prima la media
      int sum=0;
      for(int i=n/2;i<n;i++)
            for(int j=0;j<m;j++)
                  sum+= A[i][j];
      double media=(double)sum/(n/2*m);
      // verifica
      boolean verifica=true;//vero salvo eccezione
      for(int i=0;i<n/2;i++)
            for(int j=0;j<m;j++)
                  if(A[i][j]>media) {verifica=false;break;}
      return verifica;
}
/*Un metodo creaMatrice che riceve una matrice di interi A e restituisce una
matrice M
```

```
delle stesse dimensioni di A. La i-esima riga di M sarà uguale alla i-esima
riga di A se i è pari, mentre sarà uguale all'inverso della i-esima riga di A
se i è dispari.*/
public static int[][] creaMatrice(int A[][]){
      int n=A.length,m=A[0].length;
      int [][] M=new int [n][m];
      // prima riempio le righe pari
      for(int i=0;i<n;i+=2)
            for(int j=0;j<m;j++)
                  M[i][j]=A[i][j];
      // poi le dispari
      for(int i=1;i<n;i+=2)
            for(int j=0;j<m;j++)
                  M[i][j]=A[i][m-1-j];
      return M;
}
/*Un metodo main nel quale si legge una matrice di interi, e si invocano
opportunamente i metodi definiti ai punti 1, 2 e 3.*/
public static void main (String args[]){
       int [][] M= {   {1,0,3,18,1},
                             {1,3,0,0,3},
                             {0,1,6,2,12},
                             {9,2,4,0,1}};
       int n=M.length;
       int m=M[0].length;

       //public static int [] sommaMultipli(int [][] A; int k)
       int k=3;
       int [] Vett= sommaMultipli(M,k);

       // stampa vettore
        for(int i=0;i<Vett.length;i++)System.out.println(Vett[i]);

       //public static boolean verificaMedia(int A[][])
       System.out.println(verificaMedia(M));

        // oppure
       if(verificaMedia(M))  System.out.println("la matrice verifica ...");

       //public static  int[][]   creaMatrice(int A[][])
       int[][] M2=creaMatrice(M);
       for(int i=0;i<n;i++){
                 for(int j=0;j<m;j++)
                       System.out.print(M2[i][j]+" ");
        System.out.println("");
       }
  } //main
} //  class
```

Homework

Si scriva un programma che effettua le seguenti operazioni:

1. Legge da input un array di interi **v1** di dimensione **d1** (specificata dall'utente).
2. Stampa gli elementi di **v1** su video.
3. Legge da input un array di interi **v2** di dimensione **d2** (specificata dall'utente).
4. Stampa gli elementi di **v2** su video.
5. Crea un array di interi **v3** di lunghezza **d1+d2** contenente gli elementi di **v1** seguiti dagli elementi di **v2**.
6. Stampa gli elementi di **v3** su video.
7. Crea un array di interi **v4** di lunghezza **d1+d2** contenente gli elementi di **v2** seguiti dagli elementi di **v1**.
8. Stampa gli elementi di **v4** su video.

Ad esempio se l'utente ha inserito i due array:

v1 =	**3**	**8**	**4**

v2 =	**5**	**1**	**7**	**2**	**9**

il programma dovrà creare e stampare i seguenti array:

v3 =	**3**	**8**	**4**	**5**	**1**	**7**	**2**	**9**

V4 =	**5**	**1**	**7**	**2**	**9**	**3**	**8**	**4**

Si scriva una prima soluzione nella quale tutte le istruzioni sono comprese nel solo metodo main.

Successivamente si scriva una soluzione che fa uso di metodi per:

1. Leggere un array da input.
2. Stampare un array su video.
3. Restituire la concatenazione di due array.

Esercizio 1

Si scriva un metodo *sommaElementi* che riceve come parametro una matrice di interi, e restituisce la somma dei suoi elementi. Si scriva inoltre un programma che: 1) legge una matrice *M* da input; 2) stampa la matrice *M* su video; 3) calcola la somma degli elementi di *M* utilizzando il metodo *sommaElementi* e 4) stampa su video il valore della somma così calcolata.

Esercizio 2

Si scriva un metodo massimoRiga che riceve come parametri una matrice di interi *M* ed un numero intero *r*, e restituisce il massimo tra gli elementi della riga *r* di *M*.

Esercizio 3

Si scriva un metodo *minimoColonna* che riceve come parametri una matrice di interi *M* ed un numero intero *c*, e restituisce il minimo tra gli elementi della colonna *c* di *M*.

Esercizio 4

Si scriva un metodo *puntiDiSella* che riceve come parametro una matrice di interi A, e stampa su video la posizione (indice di riga ed indice di colonna) degli elementi di A che sono contemporaneamente il massimo della propria riga ed il minimo della propria colonna. Si faccia uso dei metodi *massimoRiga* e *minimoColonna* definiti precedentemente.

Esercizio 5

Si scriva un metodo *matriceSparsa* che riceve come parametri una matrice di interi *M* ed un intero *k*, e restituisce *true* se ogni riga di *M* contiene meno di *k* elementi diversi da zero, altrimenti restituisce *false*.

Esercizio 6

Si scriva un metodo *estraiDiagonalePrincipale* che riceve come parametro una matrice quadrata di interi A, e restituisce un vettore contenente gli elementi della diagonale principale di A.

Esercizio 7

Si scriva un metodo *estraiDiagonaleSecondaria* che riceve come parametro una matrice quadrata di interi A, e restituisce un vettore contenente gli elementi della diagonale secondaria di A.

Esercizio 8

Si scriva un metodo *sommaTriangolo* che riceve come parametri una matrice quadrata di interi *M* ed un numero intero *t*, che può assumere i valori 0 o 1, e restituisce un numero intero. In particolare, se *t* vale 0, il metodo restituisce la somma degli elementi del triangolo inferiore di *M*, altrimenti restituisce la somma degli elementi del triangolo superiore di *M*.

Esercizio 9

Si scriva un metodo *sommeUguali* che riceve come parametro una matrice quadrata di interi *A*, e restituisce un booleano. In particolare, il metodo restituisce *true*, se la somma degli elementi del triangolo inferiore di *A* è uguale alla somma degli elementi del triangolo superiore di *A*, altrimenti restituisce *false*. Si faccia uso del metodo *sommaTriangolo* definito precedentemente.

Esercizio 10

Si scriva un metodo *estraiSottomatrice* che riceve come parametri una matrice di interi *M* e due numeri interi *r* e *c*, e restituisce una matrice di dimensione *r* x *c*, contenente gli elementi che si trovano nelle prime *r* righe e nelle prime *c* colonne di *M*.

Esercizio 11

Si definisca un metodo *verificaSimmetria* che riceve un array di interi ***V*** di dimensione pari e restituisce un booleano. In particolare, il metodo restituisce *true* se le coppie degli elementi di posizione opposta in ***V*** hanno tutte somma uguale. Altrimenti, il metodo restituisce *false*.
Ad esempio:

- Se ***V*** = [18, 10, 14, 11, 15, 7], il metodo restituisce *true* perché le somme degli elementi di posizione opposta sono 18+7 = **25**, 10+15 = **25**, 14+11 = **25**.
- Se ***V*** = [17, 10, 18, 1, 11, 4], il metodo restituisce *false* perché le somme degli elementi di posizione opposta sono 17+4 = **21**, 10+11 = **21**, 18+1 = **19**.

Esercizio 12

Si realizzi un metodo *elaboraVettori* che, ricevuti in ingresso due vettori **v1** e **v2** di interi restituisce *false* se i due vettori non hanno la stessa dimensione. Se entrambi hanno la stessa dimensione, il metodo restituisce *true* se la somma degli elementi di **posizione dispari** di **v1** è uguale al prodotto degli elementi con **valore pari** di **v2**.
Ad esempio, se **v1**= [1, 2, 0, 16, 4, -2] e **v2** = [1 , 2 , 4 , 7, 2, 1], il valore restituito è *true* perchè **v1**[1] + **v1**[3] +**v1**[5] = **v2**[1] * **v2**[2] * **v2**[4] = 16.

Esercizio 13

Scrivere un metodo *componiVettori* che riceve due array **V1** e **V2** di **n** interi e ritorna un vettore **V3** della stessa lunghezza in cui l'elemento di posizione **i** è pari alla somma dell'elemento in posizione **i** di **V1** e di tutti gli elementi di **V2** in posizione diversa da **i**.
Ad esempio, se **V1** = [3, 5, 11, 10], **V2** = [1, 4, 14, 6] allora il risultato sarà **V3** = [27, 26, 22, 29].

Esercizio 14

Scrivere un metodo *gestioneVettori* che riceve due array **V1** e **V2** di **n** interi e ritorna un vettore **V3** della stessa lunghezza in cui ciascun l'elemento di posizione **i** viene così calcolato**:**

- se l'indice **i** è pari**, V3[i]** sarà uguale alla somma degli elementi di **V1** con indice minore o uguale di **i ;**
- altrimenti se **i** è dispari **V3[i]** sarà uguale alla somma degli elementi di **V2** con indice maggiore o uguale di **i ;**

Ad esempio, se **V1** = [7, 5, 2, 10], **V2** = [3, 4, 11, 6] allora il risultato sarà **V3** = [7, 21, 14, 6].

Esercizio 15

Si consideri il seguente codice:

```
public static void metodo(int []a, int k)
 {
       int i=0;
       while (i< a.length) {
          if (a[i] % k == 0 )
             a[i] = a[i]/k;
          else
               a[i] = k;
          System.out.println(a[i]);
          i = i+1;
      }
 }
```

Si descriva sinteticamente la funzione svolta dal metodo e, in particolare, se ne mostri la traccia d'esecuzione nel caso in cui il valore dei parametri sia a = [15, 12, 4, 6, 27, 7] e k = 3**.** Specificare cosa viene stampato.

Prova scritta dell'esame di Fondamenti di Informatica Data: 25 marzo 2006 TRACCIA A

Esercizio 1

Si consideri il seguente programma:

```
public class Esercizio1A {
	public static int[] metodoA(int x[]) {
		int[] y = new int[x.length];
		int i;
		for (i = 0; i < x.length-2; i+=2) {
			y[i] = x[i+1];
			y[i+1] = x[i];
		}
		if (x.length % 2 != 0)
			y[i] = x[i];
		else {
			y[i] = x[i+1];
			y[i+1] = x[i];
		}
		return y;
	}
	public static void main(String args[]) {
		int v[] = {1, 5, 9, 5, 2, 3, 7};
		int k[] = metodoA(v);
		for (int i = 0; i < k.length; i++)
			System.out.print(k[i] + ",");
	}
}
```

Si descriva sinteticamente la funzione svolta dal metodo **metodoA** e, in particolare, se ne mostri l'esecuzione e si specifichi cosa viene stampato nel caso in esempio, in cui **v** = **{1, 5, 9, 5, 2, 3, 7}**.

Esercizio 2

Si scriva un metodo *costruisciVettore* che riceve in ingresso un vettore di interi **v1**, e restituisce un vettore **v2** della stessa dimensione. In particolare, il vettore **v2** è così costruito:

- **v2[i]** è pari alla media degli elementi di **v1** con indice >= di **i**, se tale media è >= di **v1[i]**;
- altrimenti **v2[i]** è pari alla differenza tra la somma degli elementi alla sinistra di **v1[i]** e la somma degli elementi alla destra di **v1[i]** (ovviamente se non c'e' nessun elemento alla destra o alla sinistra tale somma vale zero).

Ad esempio, se **v1**= [20 , 9 , 4, 8 , 2] il vettore restituito è **v2** = [-23, 6, 4, 31, 2].

Esercizio 3

Si realizzi una classe *Esercizio3A* che almeno contenga i seguenti metodi:

1. Un metodo *elaboraArray* che riceve una matrice quadrata di interi **M** di dimensione dispari, e restituisce un vettore **V** contenente gli elementi situati nelle sottomatrici quadrate (lette per righe o per colonne) che si ottengono da **M** escludendo la riga e la colonna centrale (si veda l'esempio sottostante).
2. Un metodo *verifica* che riceve una matrice di interi **M** e un intero ***p***, e restituisce un booleano. In particolare, il metodo restituisce **true** se ci sono almeno ***p*** elementi di **M** che costituiscano una serie di numeri consecutivi da ***1*** fino a ***p***, altrimenti restituisce **false**.
3. Un metodo *elaboraMatrice* che riceve una matrice di interi **M** e restituisce una matrice **A** ottenuta da **M** eliminando gli elementi che appartengono alla diagonale principale.
4. Un metodo *main* nel quale si legge una matrice quadrata di interi, e si invocano opportunamente i metodi definiti ai punti 1, 2 e 3.

Esempio:

M =

3	19	13	22	7
15	0	24	2	16
4	21	9	14	23
17	1	25	8	11
6	20	12	18	5

1. *elaboraArray* (**M**) restituisce **V** = [3,19,15,0,22,7,2,16,17,1,6,20,8,11,18,5].
2. *confronta* (**M, 6**) restituisce **true** perché ci sono all'interno della matrice 6 numeri interi consecutivi fino a 6 stesso, cioè 1,2,3,4,5,6.
3. *elaboraMatrice* (**M**) restituisce **A** =

19	13	22	7
15	24	2	16
4	21	14	23
17	1	25	11
6	20	12	18

Prova scritta dell'esame di Fondamenti di Informatica Data: 25 marzo 2006 TRACCIA B

Esercizio 1

Si consideri il seguente programma:

```
public class Esercizio1B {
	public static int[] metodoB(int x[]) {
		int[] y = new int[x.length];
		int i, n = x.length;
		for (i = 0; i < n/2; i++) {
			y[n/2+i] = x[n-1-i];
			y[n/2-i-1] = x[i];
		}
		if (x.length % 2 != 0)
			y[n/2+i] = x[n-1-i];
		return y;
	}
	public static void main(String args[]) {
		int v[] = {1, 5, 9, 5, 2, 3, 7};
		int k[] = metodoB(v);
		for (int i = 0; i < k.length; i++)
			System.out.print(k[i] + ",");
	}
}
```

Si descriva sinteticamente la funzione svolta dal metodo **metodoB** e, in particolare, se ne mostri l'esecuzione e si specifichi cosa viene se **v** = **{1, 5, 9, 5, 2, 3, 7}**.

Esercizio 2

Si scriva un metodo *generaVettore* che riceve in ingresso un vettore di interi **v1**, e restituisce un vettore **v2** della stessa dimensione. In particolare, il vettore **v2** è così costruito:

- **v2[i]** è pari alla media degli elementi di **v1** con indice <= di **i**, se tale media è <= di **v1[i]**;
- altrimenti **v2[i]** è pari alla differenza tra la somma degli elementi alla destra di **v1[i]** e la somma degli elementi alla sinistra di **v1[i]** (ovviamente se non c'e' nessun elemento alla destra o alla sinistra tale somma vale zero).

Ad esempio, se **v1**= [20 , 9 , 4, 8 , 2] il vettore restituito è **v2** = [20, -6, -19, -31, -41].

.

Esercizio 3

Si realizzi una classe *Esercizio3A* che contenga i seguenti metodi:

5. Un metodo *elaboraArray* che riceve una matrice quadrata di interi **M** di dimensione pari, e restituisce un vettore **V** contenente gli elementi situati nelle forme a "L" sottostanti (di dimensione M.length/2) che si ottengono da **M** come illustra l'esempio qui sotto.
6. Un metodo *verifica* che riceve una matrice di interi **M** e un intero ***p***, e restituisce un booleano. In particolare, il metodo restituisce **true** se ci sono almeno ***p*** elementi di **M** che costituiscano numeri multipli di ***p***, altrimenti restituisce **false**.
7. Un metodo *elaboraMatrice* che riceve una matrice di interi **M** e restituisce una matrice **A** ottenuta da **M** eliminando gli elementi che appartengono alla diagonale principale e a quella secondaria.
8. Un metodo *main* nel quale si legge una matrice quadrata di interi, e si invocano opportunamente i metodi definiti ai punti 1, 2 e 3.

Esempio:

M =

3	19	13	7
15	0	24	16
4	21	9	23
6	20	12	5

4. *elaboraArray* (**M**) restituisce **V** = [13,7,16,4,6,20].
5. *verifica* (**M, 3**) restituisce **true** perché ci sono all'interno della matrice 3 numeri interi multipli di 3 stesso, cioè 3,15,24.
6. *elaboraMatrice* (**M**) restituisce **A** =

19	13
15	16
4	23
20	12

Prova scritta dell'esame di Fondamenti di Informatica Data: 23 marzo 2006 TRACCIA C

Esercizio 1

Si consideri il seguente metodo:

```
public class Esercizio1C{
   public static boolean metodo1(int[] v1, int k)
   {
      int x=0;
      for (int i = 0; i < v1.length; i++)
         for (int j = i+1; j < v1.length;  j++)
            if (v1[j] > k)
            {v1[i] += v1[j]; x++;};
      if (x%k==0)
         return true;
      return false;
   }
   public static void main(String args[]) {
      int v[] = {3, 5, 0, 8, 9, 4, 6};
      System.out.print(metodo1(v,6));
   }
}
```

Si illustri il funzionamento del metodo *metodo1*. In particolare si mostri cosa viene stampato in output e si specifichi il contenuto dell'array v al termine dell'esecuzione del metodo1.

Esercizio 2

Si scriva un metodo *componivett* che riceve un array di interi **V1** e restituisce un array di interi **V2** contenente gli elementi **V1[i]**, con **0 < i < V1.length**, che sono multipli di tutti gli elementi che li precedono nel vettore **V1.**

Ad esempio, se **V1** = [5, 2, 10, 30, 3,90] allora il risultato sarà **V2** = [10,30,90] .

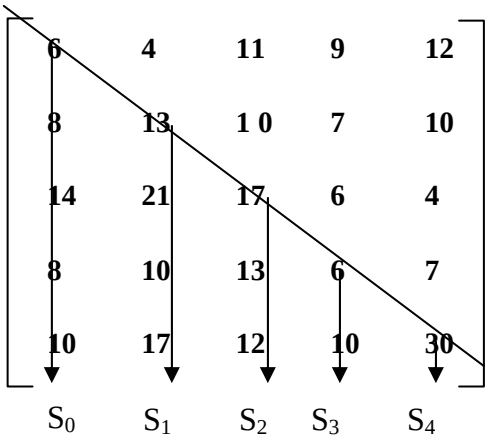
Esercizio 3

Si realizzi una classe *Matrice* per rappresentare matrici di interi che contenga almeno i seguenti metodi:

1. un metodo *controlla* che riceve una matrice quadrata di interi **M** ed un intero **K** e restituisce *true* se il numero totale di numeri primi presenti nella matrice è maggiore di **K** (per maggiore chiarezza si veda l'esempio) ;
2. un metodo *creaVettore* che riceve una matrice quadrata **M** di interi e restituisce un vettore **V** in cui l'elemento in posizione **i** è pari alla somma degli elementi appartenenti al segmento $\mathbf{S_i}$ contenente gli elementi sulla colonna i-esima a partire dall'elemento sulla diagonale (incluso) (per maggiore chiarezza si veda l'esempio) ;
3. un metodo *creaOrlata* che riceve una matrice quadrata **M** di interi e due interi **r** e **c** e restituisce una matrice **W** ottenuta da M eliminando la riga **r** e la colonna **c** (per maggiore chiarezza si veda l'esempio) ;
4. un metodo *main* che legge una matrice quadrata **M** di numeri interi e provvede, invocando opportunamente i metodi proposti, a realizzare i compiti di cui ai punti (1), (2) e (3) .

Esempio:

M=



- Il metodo c*ontrolla* invocato sulla matrice ***M*** con K=4 restituisce *true*. Infatti 7 sono i numeri primi presenti nella matrice (7>k)
- Il metodo c*reaVettore* invocato sulla matrice ***M***, restituisce il vettore ***V*** = [46, 61,42,16,30]
- Il metodo *creaOrlata*, invocato sulla matrice M con r=1 e c=2 restituisce la matrice:

Prova scritta dell'esame di Fondamenti di Informatica Data: 23 marzo 2006 TRACCIA D

Esercizio 1

Si consideri il seguente metodo:

```
public class Es1TD{
   public static int calcola (int[] v){
      int x=0; int d;
      int m=v.length/2;
      if (v.length%2==0)
         d= m;
      else
         d= m+1;
      for (int i = 0; i < m; i++)
         if (v[i] == v[i+d]) x++;
      return x;
   }
   public static void main(String args[]) {
         int v[] = {3, 5, 0, 7, 3, 5, 0};
         System.out.print(calcola(v));
   }
}
```

Si illustri il funzionamento del metodo *calcola* e si specifichi l'output ottenuto se v ={**3, 5, 0, 7, 3, 5, 0}**.

Esercizio 2

Si scriva un metodo *costruisciVettore* che riceve in ingresso un array **V** di interi positivi ed un intero **k** e crea e restituisce un array di interi **W** in cui l'elemento **W[i]** è così calcolato**:**
W[i]=a + b dove **a** è la somma degli elementi che precedono l'elemento i-esimo di **V** e **b** è il prodotto degli elementi che seguono l'elemento i-esimo di **V,** se **a + b > k**; **W[i]=a** altrimenti (Se non c'e' nessun elemento alla destra dell'elemento i-esimo si assuma **b**=0, se non c'e' nessun elemento alla sinistra dell'elemento i-esimo si assuma **a**=0).
Ad esempio, se **k=10** e **V** = [2, 3, 1, 1, 7], allora il risultato sarà **W** = [21, 2, 12, 13, 7].

Esercizio 3

Si realizzi una classe *GestMatrici* per la gestione di matrici quadrate che contenga almeno i seguenti metodi:

1. un metodo *verifica* che riceve in ingresso una matrice quadrata **M** e restituisce *true* se il prodotto degli elementi sulla diagonale principale è pari alla somma degli elementi della diagonale secondaria (per maggiore chiarezza si veda l'esempio);
2. un metodo *creaVettore* che riceve in ingresso una matrice quadrata **M** e restituisce un array **V** in cui l'elemento i-esimo è pari alla somma della riga i-esima se questa contiene almeno un numero primo, e alla somma della colonna i-esima altrimenti (per maggiore chiarezza si veda l'esempio);
3. un metodo *estrai* che riceve in ingresso una matrice quadrata **M** di interi e due interi **r** e **c** e restituisce una matrice di interi **Q**, ottenuta eliminando da **M** le righe con indice maggiore di **r** e le colonne con indice minore di **c** (per maggiore chiarezza si veda l'esempio);
4. un metodo *main* che legge una matrice quadrata **M** di numeri interi e provvede, invocando opportunamente i metodi proposti, a realizzare i compiti di cui ai punti (1), (2) e (3) .

M=

4	**4**	**4**	**9**	**10**
8	**2**	**1 0**	**7**	**10**
14	**21**	**3**	**6**	**4**
8	**16**	**4**	**1**	**4**
12	**3**	**4**	**10**	**2**

- Il metodo *verifica* invocato sulla matrice ***M*** restituisce *true*.
- Il metodo c*reaVettore* invocato sulla matrice ***M***, restituisce il vettore **V** = [46, 37,48,33,31]
- Il metodo *estrai*, invocato sulla matrice M con r=3 e c=2 restituisce la matrice:

4	9	10
10	7	10
3	6	4
4	1	4