```java
public class Esame21gennaio {
	public static void main(String[] args) {
	//ESERCIZIO 1b
	// Cosa stampa il seguente frammento di codice Java?
	System.out.println ("STAMPA ESERCIZIO 1b");
	boolean nonoutput=false;
	int n=16,i;
	for (i=0; i<9 && ! nonoutput; i++) {
		if (i*i==n){ nonoutput=true;}
		else {System.out.println ("output " + n+i);
			}
		}
	//ESERCIZIO 1a
	//Cosa stampa il seguente programma Java (prova è scritto sotto)
	System.out.println ();
	System.out.println ("STAMPA ESERCIZIO 1a");
	System.out.println(prova(2));
	System.out.println(prova(5));
	System.out.println(prova(9));
	System.out.println(prova(14));

	// Stampa il risultato di 2 chiamate al metodo puntoDiCulmine
	// LE 6 RIGHE CHE SEGUONO NON ANDAVANO SCRITTE.
	//Solo per  farvi vedere il comportamento del metodo
	System.out.println ();
	System.out.println ("DUE ESECUZIONI DI puntoDiCulmine");
	int []a={2,2,3,3,3,3};
	System.out.println(puntoDiCulmine(a));
	int []b={2,2,2,2,9};
	System.out.println(puntoDiCulmine(b));

	// Stampa l'array risultante da 1 chiamata al metodo rimpiazzaSomma
	// LE 5 RIGHE CHE SEGUONO NON ANDAVANO SCRITTE.
	//Solo per  farvi vedere il comportamento del metodo
	System.out.println ();
	System.out.println ("UNA ESECUZIONE DI rimpiazzaSomma");
	int []c={10, 20, 20, 50, 60, 90} ;
	rimpiazzaSomma (c);
	stampaArray (c);

	// Stampa l'array risultante da 1 chiamata al metodo rimpiazzaSomma2
	// LE 5 RIGHE CHE SEGUONO NON ANDAVANO SCRITTE.
	//Solo per  farvi vedere il comportamento del metodo
	System.out.println ();
	System.out.println ("UNA ESECUZIONE DI rimpiazzaSomma2");
	int []d={10, 20, 20, 50, 60, 90} ;
	rimpiazzaSomma2 (d);
	stampaArray (d);
		}
```

```java
// metodo prova nel testo del compito
static int prova (int x){
      if (x<0) return 0;
      if (x%2!=0) return prova(x-3) + x;
      return prova (x-3) - x;
         }

// IL metodo stampaArray scritto a lezione NON ANDAVA SCRITTO.
//Solo per farvi vedere il comportamento degli altri metodi
static void stampaArray (int[]a) {
      if (a==null) {
           System.out.println("Array non inizializzato");
           return;
                 }
      System.out.print("{ ");
      for (int i=0; i<a.length; i++){
            System.out.print(a[i]);
            if (i<a.length-1) System.out.print(", ");
                 }
      System.out.println(" }");
      }


/* Esercizio 2
*Un array a di n elementi interi si dice culminante
*se  n > 1 ed esiste un indice m, detto punto di culmine,
*con 0 < m < n tale che
*        a [0] = . . . = a [m - 1] < a [m] = . . . = a [n - 1]
*(ovvero, i primi m elementi sono uguali, e gli ultimi n-m
*sono anch'essi uguali e strettamente maggiori dei precedenti).
*Scrivere un metodo in Java
*     static int puntoDiCulmine (int[] a)
*che, preso come parametro un array a di numeri interi
*culminante, restituisce  il punto di culmine di a.
*Ad esempio, se a={2,2,3,3,3,3} il risultato dovrà essere 2,
*mentre se a={2,2,2,2,9}, il risultato dovrà essere 4.
*Si analizzi la complessità temporale del metodo proposto:
*tale metodo deve avere complessità temporale O(log n)
*nel caso peggiore (soluzioni con complessità temporale
*peggiore danno luogo a una valutazione minore),
*dove n è la lunghezza dell'array a.
*/

//SOLUZIONE POSSIBILE
static int puntoDiCulmine (int[] a)
  {
      return puntoDiCulmine(a,0,a.length-1);
  }
```

```java
    static int puntoDiCulmine (int[] a, int l,int r)
          {if (r==l+1) return r;
          int p=(r+l)/2;
          if (a[l]==a[p]) return puntoDiCulmine (a, p,r);
          return puntoDiCulmine (a, l,p);
          }


    /* Esercizio 3
    * Scrivere un metodo in Java
    * static void rimpiazzaSomma (int[] a)
    * che preso in input un array a di n elementi interi,
    * rimpiazza ogni elemento  dell'array con la somma
    * degli elementi che lo seguono, sempre nell'array.
    * Ad esempio, se a={10, 20, 20, 50, 60, 90} dopo
    * l'esecuzione del metodo si avrà  a={240, 220, 200, 150, 90, 0}.
    * Si analizzi la complessità temporale del metodo proposto:
    * tale metodo deve avere complessità temporale O(n)
    * (soluzioni con complessità temporale peggiore danno
    * luogo a una valutazione minore), dove n è la lunghezza dell'array a.
    */

    //PRIMA SOLUZIONE POSSIBILE
    static void rimpiazzaSomma (int[] a)
     {if (a== null|| a.length==0) return;
      int y=0;
      int x;
      for (int i=a.length-1; i>=0; i--)
          {x=a[i]; a[i]=y; y=y+x;}
     }

    //ALTRA POSSIBILE SOLUZIONE
    static void rimpiazzaSomma2 (int[] a)
     {if (a== null|| a.length==0) return;
      int somma=0;
      for (int i=1; i<a.length; i++)
          somma=somma+a[i];
      for (int i=0; i<a.length-1; i++)
          {a[i]=somma;
          somma=somma-a[i+1];}
      a[a.length-1]=0;
      }

    /*Esercizio 4
    * Si consideri il seguente array di numeri interi:
    *         {39, 27, 43, 13, 19, 82, 22, 6, 11, 28, 44}
    *  Mostrare passo-passo l'esecuzione del merge-sort per ordinare l'array
    *  in modo non decrescente. Si descriva la complessità computazionale
    * del merge-sort (anche scrivendo e risolvendo la ricorrenza T(n)).
    */
}  //CHIUSURA DELLA CLASSE
```