```java
public class Esame18febbraio{
	public static void main(String[] args) {
	//ESERCIZIO 1b
	// Cosa stampa il seguente frammento di codice Java?
	System.out.println ("STAMPA ESERCIZIO 1a");
	int conto=0;
	int[] a = {9, 3, 5, 6, 4, 5};
	int n=20, i;
	for (i=a.length-1; i>0 && conto <=2; i--) {
		for (int j=0; j<i; j++) {
			if (a[i]*a[j]<=n){
				conto++;
				System.out.println(i + " e " +j);
			}
		}
		System.out.println(conto);
	}

	//ESERCIZIO 1a
	//Cosa stampa il seguente programma Java (enigma è scritto sotto)
	System.out.println ();
	System.out.println ("STAMPA ESERCIZIO 1b");
	System.out.println(enigma(8));
	System.out.println(enigma(12));
	System.out.println(enigma(25));


// Stampa del risultato delle chiamate dei metodi ripeti
// e cancellaRip.
// Le 12 righe che seguono non vanno scritte
// Solo per farvi vedere il comportamento dei metodi.

	int []b={4, 6, -4, 3, -1} ;
	int [] c={1, 2, -5, 0, 3};
	int [] rip=ripeti(b,c);
	System.out.println ("risultato di ripeti(b,c)");
	stampaArray(rip);
	int []d={9,9,9,1,1,1,3,2,2,2,4};
	int [] canc2=cancellaRip (d, 2);
	int [] canc1=cancellaRip (d, 1);
	System.out.println ("risultato di cancellaRip(d,2)");
	stampaArray(canc2);
	System.out.println ("risultato di cancellaRip(d,1)");
	stampaArray(canc1);
	}

/*********************************************************/
	/* stampaArray non si deve scrivere
	 * nel compito. Serve per provare i metodi
	*/
```

```java
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(" }");
}


static int enigma (int x){
	if (x<=0 || x>=20) return x;
	if (x<=10) return enigma(-x);
	return x+enigma(x+5);
	}

/*Esercizio 2
Scrivere un metodo
static int[] ripeti (int[] el, int[] volte)
che, presi come parametro due array di numeri interi
della stessa lunghezza n, crea e restituisce un array che contenga,
per ogni i=0,…,n-1, volte[i] copie consecutive dell'elemento
el[i] (se volte[i] non è positivo el[i]
non viene inserito nell'array).
Se el o volte vale null, oppure se non hanno la stessa lunghezza,
viene restituito null.
Ad esempio se el={4, 6, -4, 3, -1}, e volte={1, 2, -5, 0, 3},
viene creato e restituito l'array {4, 6, 6, -1, -1, -1}.
*/

static int[] ripeti (int[] el, int[] volte){
	if (el==null || volte==null|| el.length!=volte.length)
			return null;
	if (el.length==0) return el;
	int conta=0;
	for (int i=0; i<volte.length; i++)
		if (volte[i]>0) conta=conta+volte[i];
	int []b=new int[conta];
	int k=0;
	for (int i=0; i<el.length; i++) {
		for (int j=1; j<=volte[i]; j++)
			{b[k]=el[i]; k++;}
	}
	return b;
}
```

```java
	/* Esercizio 3
	 * Scrivere un metodo in Java
	 *    static int[] cancellaRip (int[] elementi, int k)
	 * che, preso in input un array di interi a
	 * ed un intero positivo k,
	 * restituisce l'array ottenuto a partire da a eliminando
	 * tutti gli elementi in eccedenza che compaiono più di
	 * k volte consecutive, mantenendo lo
	 * stesso ordine che gli elementi avevano
	 * in a (l'array a non deve essere modificato).
	 * Ad esempio, se a={9,9,9,1,1,1,3,2,2,2,4} e
	 * k è il numero intero 2 allora cancellaRip
	 * restituirà l'array { 9,9,1,1,3,2,2,4}.
	 * Si analizzi la complessità temporale del metodo proposto.
	 */

	static int[] cancellaRip (int[] elementi, int k) {
		if (elementi==null) return null;
		if (k==0||elementi.length==0) return new int[0];
		int contael=1, conta=1, prec=elementi[0];
		for (int i=1; i<elementi.length; i++) {
			if(elementi[i]!=prec) {
				contael=1; conta++; prec=elementi[i];
				}
			else if (contael<k) { contael++; conta++;}
			}
		System.out.println (conta); //non si deve scrivere.
		int [] ris=new int [conta];
		int j=1; // per scorrere l'array ris
		contael=1; prec=elementi[0]; ris[0]=elementi[0];
		for (int i=1; i<elementi.length; i++) {
		if(elementi[i]!=prec) {
			contael=1; prec=elementi[i]; ris[j]=elementi[i]; j++;
			}
		else if (contael<k) {
			contael++; ris[j]=elementi[i]; j++;
			}
		}
			return ris;
	  }

	/*Esercizio 4
	 * Si consideri il seguente array di numeri interi:
	 * {37, 27, 25, 24, 19, 44, 70, 16, 11, 58, 62}
	 * Mostrare passo-passo l'esecuzione del merge-sort per ordinare
	 * l'array in modo non decrescente. Si descriva la complessità
	 * computazionale del merge-sort.
	*/
}
```