**Appello di Programmazione e Algoritmi 1**
**10 febbraio 2022 in presenza**

**Esercizio 1**
Cosa stampa il seguente programma Java?

```
public static void main(String[] args) {
          System.out.println(calcola(-11));
          System.out.println(calcola(11));
          System.out.println(calcola(17));
          }
     static int calcola (int x){
          if (x>=0 && x<10) return x;
          if (x<0) return calcola(x+3) - x;
          return x- calcola(x-3);
         }
```

**NON SARANNO CONSIDERATE VALIDE LE RISPOSTE NON GIUSTIFICATE DAI CALCOLI EFFETTUATI PER OTTENERE IL RISULTATO**

Cosa stampa il seguente frammento di codice Java?

```
int conta=0;
for (int i=0;i<4;i++) {
          for (int j=3;j>=i;j--) {
               if ((i+j)%3!=0){
                    conta++;
               }
               else System.out.println("i: " + i + ", j: " +  j);
          }
          System.out.println("conta: " + conta);
}
```

**NON SARANNO CONSIDERATE VALIDE LE RISPOSTE NON GIUSTIFICATE DAI CALCOLI EFFETTUATI PER OTTENERE IL RISULTATI**

**Esercizio 2**
Scrivere un metodo in Java

**static boolean semprePrima (int[] a, int n, int m)**

che preso in input un array **a** di elementi interi e due interi **n** e **m**, restituisce true se nell'array **a** tutte le occorrenze di **n** precedono tutte le occorrenze di **m**.

Ad esempio: se **a** = {2, -11, 2, 4, 2, 3, -7,3, 5}, **n** è uguale a 2 e **m** è uguale a 3 il metodo dovrà restituire true, perché tutte le occorrenze del valore 2 (nell'array a) precedono tutte le occorrenze del valore 3 (nell'array a). Se **a** = {2, -11, 2, 4, 3, 2, -7,3, 5}, **n** è uguale a 2 e **m** è uguale a 3 allora il metodo dovrà restituire false, poiché c'è un'occorrenza del valore 2 (ovvero a[5]) che segue un'occorrenza del valore 3 (ovvero a[4]). Se n o m non compaiono nell'array, allora il metodo dovrà restituire true.

**Si analizzi la complessità temporale del metodo proposto** (*soluzioni con complessità temporale peggiore danno luogo a una valutazione minore*).

**Esercizio 3**

Scrivere un metodo in Java

**static int[] inserisciDopo (int[] a, int n , int m)**

che preso in input un array **a** di elementi interi, un intero **n** ed un intero **m**, restituisce un nuovo array ottenuto da **a** inserendo dopo ogni occorrenza di **n** (nell'array **a**) un'occorrenza del valore **m**. Ad esempio: se **a**={5, 7, -9, 5, 8, 5, -3}, **n** è uguale a 5 e **m** è uguale a 7, allora viene restituito l'array {5, 7, 7, -9, 5, 7, 8, 5, 7, -3} (viene messo un nuovo elemento di valore 7 dopo ogni elemento di valore 5). Altro esempio, se **a**={-1, 1, 3, 1, 5, 2}, **n** è uguale a 1 e **m** è uguale a 3, il metodo dovrà restituire {-1, 1, 3, 3, 1, 3, 5, 2}. Se **n** non compare in **a**, allora il metodo dovrà restituire un nuovo array che contiene esattamente gli stessi elementi di **a**.

**Si analizzi la complessità temporale del metodo proposto** (*soluzioni con complessità temporale peggiore danno luogo a una valutazione minore*).

**Esercizio 4**

Si consideri il seguente array di numeri interi:

**{17, 6, 16, 19,7, 21, 20, 10, 8, 21, 12}**

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)).

**[Esercizio di Laboratorio]** SOLO PER GLI STUDENTI CLEII che hanno svolto il progetto a partire dall'AA 2020/2021 In relazione al progetto di laboratorio, si implementino i seguenti due metodi:

public static int[] inserisciAlCentro(int[] a, int v)

che restituisce un nuovo array la cui lunghezza è uguale a quella dell'array a incrementata di 1, ottenuto inserendo in posizione centrale il valore v e copiando nelle altre posizione gli elementi dell'array a (se la lunghezza dell'array è dispari, si inserisca a sinistra dell'elemento centrale). Ad esempio, dato l'array {1,3,5,-2,0} ed il numero 9 restituisce l'array {1,3,9,5,-2,0}.

public static String creaStringaDoppioni(int[] a)

che restituisce una stringa composta dai valori dell'array a che occorrono ripetutamente. Ad esempio, dato l'array {1,3,1,0,7,8,1,7} restituisce la stringa "[1,1,7,1,7]".