Esercizi sulla Complessità

3 Maggio 2023

Esercizio 1.

Il problema della *ricerca in un array* può essere definito come segue: dato un array **a** di elementi di un certo tipo **T** ed un elemento **k** di tipo **T**, determinare se **a** contiene almeno un elemento uguale a **k**.
Il problema della ricerca in un array può essere risolto facilmente con una semplice scansione dell'array. Tale algoritmo viene chiamato *ricerca sequenziale*.
Se gli elementi dell'array sono ordinati (cioè gli elementi appaiono nell'array dal più piccolo al più grande) allora è possibile risolvere il problema della ricerca seguendo una diversa strategia che viene chiamata *ricerca binaria* (o *dicotomica*). Si confronta **k** con l'elemento mediano **m** dell'array: se **k=m** la ricerca termina; se **k<m** la ricerca continua allo stesso modo tra gli elementi a sinistra di **m** (che sono tutti più piccoli o uguali a **m**); se k>m la ricerca continua allo stesso modo tra gli elementi a destra di **m** (che sono tutti più grandi o uguali a **m**).
Scrivere in Java un metodo che realizza la ricerca sequenziale in un array ed un metodo che realizza la ricerca binaria. Determinare poi la complessità dei due metodi.

Esercizio 2.

Si vuole scrivere un metodo Java che dato un array **a** calcola un nuovo array **max** tale che **max[i]** è l'elemento massimo dei primi **i** elementi di **a**. I seguenti due metodi risolvono il problema con due procedimenti diversi. Valutare la complessità dei due metodi per determinare quale dei due è più efficiente.

```
public static int[] calcolaMassimi(int[] a){
	int max[] = new int[a.length];

	for(int i=0;i<max.length;i++){
		max[i]=a[0];
		for(int j=1;j<=i;j++)
			if(a[j]>max[i])
			max[i]=a[j];
	}

	return max;
}

public static int[] calcolaMassimi2(int[] a){
	int max[] = new int[a.length];

	max[0]=a[0];
	for(int i=1;i<max.length;i++){
		if(a[i]>max[i-1])
			max[i]=a[i];
		else
			max[i]=max[i-1];
	}

	return max;
}
```

Esercizio 3.

Si consideri il seguente metodo che riceve in input un intero positivo **n**. Indicare qual è la complessità asintotica di caso peggiore in funzione di **n**. Giustificare la risposta.

```
public static String binario(int n){
	String binario = “”;
	int q = n;
	while (q>0){
		binario=(q%2)+binario;
		q = q/2;
	}
	return binario;
}
```

Esercizio 4.

Si consideri il seguente metodo che riceve in input due matrici quadrate di dimensione **n** x **n** e ne calcola il prodotto. Indicare qual è la complessità asintotica di caso peggiore in funzione di **n**. Giustificare la risposta.

```
// si assume che a e b siano matrici di dimensione n x n
public double[][] prodotto(double[][] a, double[][] b){
  int n = a.length; // righe e colonne di a e b
  double[][] c = new double[n][n];
  for (int i = 0; i<n; i++){
    for (int j = 0; j<n; j++){
      double somma = 0;
      for (int k = 0; k<n; k++)
        somma += a[i][k]*b[k][j];
        c[i][j]=somma;
      }
    }
  return c;
}
```