

Studente (Cognome Nome): \_\_\_\_\_

Matricola: \_\_\_\_\_

Corso di Informatica  
Corso di Laurea in Ingegneria Gestionale  
a.a. 2006-07  
Secondo Compitino – 21 Dicembre 2006

*Si noti che le soluzioni ai quesiti saranno considerate valide solo se il materiale consegnato includerà anche lo svolgimento. Tale foglio deve essere consegnato insieme allo svolgimento.*

## Quesito 1

Una libreria è rappresentata da un array di libri, identificati dal titolo e dalla quantità di libri disponibili per ciascun titolo.

Un preventivo inviato da un fornitore è rappresentato da un array di libri, identificati dal titolo e dalla quantità proposta.

**a) Scrivere un metodo Java che verifichi che un preventivo sia valido.** Il metodo deve avere il seguente prototipo:

```
public static boolean is_valid(String [] lib_titolo, int [] lib_quantità, String [] preventivo_titolo, int [] preventivo_quantità)
```

e deve restituire true se:

1) tutti i titoli del preventivo sono presenti in libreria

2) la quantità per ciascun elemento del preventivo è  $>0$  e tale che sommata alla quantità disponibile in libreria dia un numero  $\leq 100$  (ossia, in libreria, dopo aver effettuato l'acquisto, per ogni titolo non possono essere presenti più di 100 libri). Prerequisiti del metodo: gli array non sono null e le quantità sono  $\geq 0$ .

**b) Scrivere un metodo Java che aggiorna l'elenco dei libri presenti in libreria in relazione all'accettazione del preventivo ricevuto, incrementando il numero delle copie disponibili della quantità proposta.** Il metodo deve avere il seguente prototipo:

```
public static void aggiorna(String [] lib_titolo, int [] lib_quantità, String [] preventivo_titolo, int [] preventivo_quantità)
```

Prerequisiti del metodo: gli array non sono null e il preventivo è valido.

**c) Scrivere un metodo Java che calcola il numero di elementi non validi all'interno di un preventivo.** Il metodo deve avere il seguente prototipo:

```
public static int non_validi(String [] lib_titolo, int [] lib_quantità, String [] preventivo_titolo, int [] preventivo_quantità)
```

Prerequisiti del metodo: gli array non sono null.

**d) Scrivere un programma main di test che**

- Inizializza la libreria con i seguenti valori:

Titolo	Disponibilità
Analisi I	4
C++	2
Visual Basic	7
Visio	1

- Crea un preventivo non valido.
- Se il preventivo non è valido, crei da questo, un preventivo valido eliminando gli elementi che lo rendono non valido. Suggerimento: Quanti sono gli elementi presenti nel nuovo preventivo (che deve essere valido)?

Esempio: se il preventivo è il seguente:

Titolo	Quantità
C#	4
Visual Basic	2
C++	99

Il preventivo valido derivato da questo è

Titolo	Quantità
Visual Basic	2

- Aggiorna l'elenco dei libri presenti in libreria
- Stampa il nuovo elenco con un formato analogo alla tabella di sopra

Note:

- E' possibile risolvere l'esercizio anche usando le classi. In tal caso, modificare in modo coerente l'intestazione delle funzioni.
- E' possibile scrivere funzioni aggiuntive per risolvere dei compiti richiesti in più punti del programma.

## Quesito 2

Si rappresenti il diagramma di flusso relativo al seguente metodo Java:

```
public static void f(int [] v) {
    int i=0;
    int N = v.length-1;
    do {
        if (v[i]>=3)
            v[i]=-v[i];
        i++;
    }
    while (i<N);
}
```

## Quesito 3

Si determini l'output prodotto dall'esecuzione del seguente programma Java, in cui il corpo del metodo f e' stato omesso per brevità, essendo identico al metodo del quesito precedente.

```
public static void main(String[] args) {
    int [] a = new int [] {1,2,5,10};
    System.out.println(a[1]+" "+a[2]+" "+a[3]);
    f(a);
    System.out.println(a[1]+" "+a[2]+" "+a[3]);
}
```

## Quesito 1: soluzione.

Note: il quesito può essere risolto in svariati modi. Qui ne viene proposto uno. Per una soluzione che fa un uso estensivo dei metodi, vedi il testo e la soluzione del quesito D, dove cambiano solo i nomi dei parametri e l'inizializzazione, ma gli algoritmi sono identici.

```
public class testoB {

    public static boolean is_valid(
        String[] lib_titolo,
        int[] lib_quantita,
        String[] preventivo_titolo,
        int[] preventivo_quantita) {
        // prima soluzione: si contano i numeri degli elementi del
preventivo
        // che sono validi.
        // se il numero è uguale alla lunghezza del preventivo, il
preventivo è valido
        boolean valido = false; // l'inizializzazione non è imp.
        int num_validi = 0; // numero di preventivi validi
        int i; // indice per ciclare sui preventivi
        int j; // indice per ciclare sui libri
        for (i = 0; i < preventivo_titolo.length; i++) {
            for (j = 0; j < lib_titolo.length; j++) {
                // cerca un titolo del preventivo
                // presente in libreria
                if (preventivo_titolo[i].equals(lib_titolo[j]))
                    // condizione di validità
                    // nota:non testo il caso di lib_quantita<0
                    // perchè è dato come prerequisito
                    if (preventivo_quantita[i] + lib_quantita[j]<=100)
valida
                        num_validi++; //la riga del preventivo è

            } // end_for_j
        } // end_for_i

        // test di validità:
        // il numeo di righe valide deve
        // essere uguale alla lunghezza del preventivo
        if (num_validi == preventivo_titolo.length)
            valido = true;
        else
            valido = false;

        return (valido);
    } //end_is_valid

    // seconda soluzione suggerita durante l'esame
    // un preventivo è valido
    // se il numero di entry non valide è pari a 0
    public static boolean is_valid1(
        String[] lib_titolo,
        int[] lib_quantita,
        String[] preventivo_titolo,
        int[] preventivo_quantita) {

        boolean valido;
        if (non_validi(lib_titolo,
            lib_quantita,
            preventivo_titolo,
```

```

        preventivo_quantita)
        == 0)
        valido = true;
    else
        valido = false;

    return valido;
}

public static void aggiorna(
    String[] lib_titolo,
    int[] lib_quantita,
    String[] preventivo_titolo,
    int[] preventivo_quantita) {
    // per ogni titolo del preventivo,
    // lo cerco in libreria
    // ed aggiornno le relative quantità
    //nota:non occorrono controlli
    // perchè il preventivo è valido
    int i; // contatore sui preventivi
    int j; // contatore sui libri
    for (i = 0; i < preventivo_titolo.length; i++) {
        for (j = 0; j < lib_titolo.length; j++) {
            if (preventivo_titolo[i].equals(lib_titolo[j]))
                // aggiornno la quantità'
                lib_quantita[j] += preventivo_quantita[i];
        } // end_for_j
    } // end_for_i
} //end_aggiorna

public static int non_validi(
    String[] lib_titolo,
    int[] lib_quantita,
    String[] preventivo_titolo,
    int[] preventivo_quantita) {
    // si contano i numeri degli elementi del preventivo
    // che sono validi.
    // il numero di elementi non validi è la lunghezza dell'array
    // dei preventivi - il numero di entry non valide
    int non_validi = 0; // numero di righe non valide
    int num_validi = 0; // numero di preventivi validi
    int i; // indice per ciclare sui preventivi
    int j; // indice per ciclare sui libri
    for (i = 0; i < preventivo_titolo.length; i++) {
        for (j = 0; j < lib_titolo.length; j++) {
            // cerca un titolo del preventivo
            // presente in libreria
            if (preventivo_titolo[i].equals(lib_titolo[j]))
                // condizione di validità
                if ((preventivo_quantita[i] >= 0)
                    && (preventivo_quantita[i] +
lib_quantita[j]<= 100))
                        num_validi++; //la riga del preventivo è
valida

        } // end_for_j
    } // end_for_i

    non_validi = preventivo_titolo.length - num_validi;

    return (non_validi);
} //end_non_validi

```

```

public static void main(String[] args) {

    // inizializzazione libreria
    String[] lib_titolo =
        new String[] { "Analisi I", "C++", "Visual Basic", "Visio" };
    int[] lib_quantita = new int[] { 4, 2, 7, 1 };
    // creazione preventivo non valido
    // (quello suggerito nel testo)
    String[] preventivo_titolo = new String[] { "C#", "Visual Basic",
"C++" };
    int[] preventivo_quantita = new int[] { 4, 2, 99 };

    boolean valido =
        is_valid(lib_titolo, lib_quantita, preventivo_titolo,
preventivo_quantita);

    if (!valido) {
        // creo un preventivo temporaneo fatto da due array paralleli.
        // la dimensione è pari alla dimensione del vecchio
        // preventivo - il numero di entry non valide
        int elem_non_validi =
            non_validi(
                lib_titolo,
                lib_quantita,
                preventivo_titolo,
                preventivo_quantita);
        int lunghezza = preventivo_titolo.length - elem_non_validi;
        String[] tmp_prev_titolo = new String[lunghezza];
        int[] tmp_prev_quantita = new int[lunghezza];

        // tutti gli elementi validi li ricopio negli array fittizi
        int i, j; // due indici per scorrere i libri e i preventivi
        int k = 0; // indice dell'elemento inserito nell'array
temporaneo
        for (i = 0; i < preventivo_titolo.length; i++)
            for (j = 0; j < lib_titolo.length; j++) {
                // condizioni di validita di una entry
                if (preventivo_titolo[i].equals(lib_titolo[j])) {
                    if ((preventivo_quantita[i] >= 0)
                        &&
(preventivo_quantita[i]+lib_quantita[j]<=100)) {
prevenitivo
                            // l'entry è valida, la ricopio nel
                            // temporaneo
                            tmp_prev_titolo[k] =
prevenitivo_titolo[i];
                            tmp_prev_quantita[k] =
prevenitivo_quantita[i];
                            k++;
                        } // end_if preventivo_quantita
                    } // end_if preventivo_titolo
                } // end_for_j

        // ora butto via il vecchio preventivo e prendo il preventivo
valido
        preventivo_titolo = tmp_prev_titolo;
        preventivo_quantita = tmp_prev_quantita;

    } // end_if_not_valido

    // aggiorno il catalogo
    aggiorna(lib_titolo, lib_quantita, preventivo_titolo,
preventivo_quantita);
}

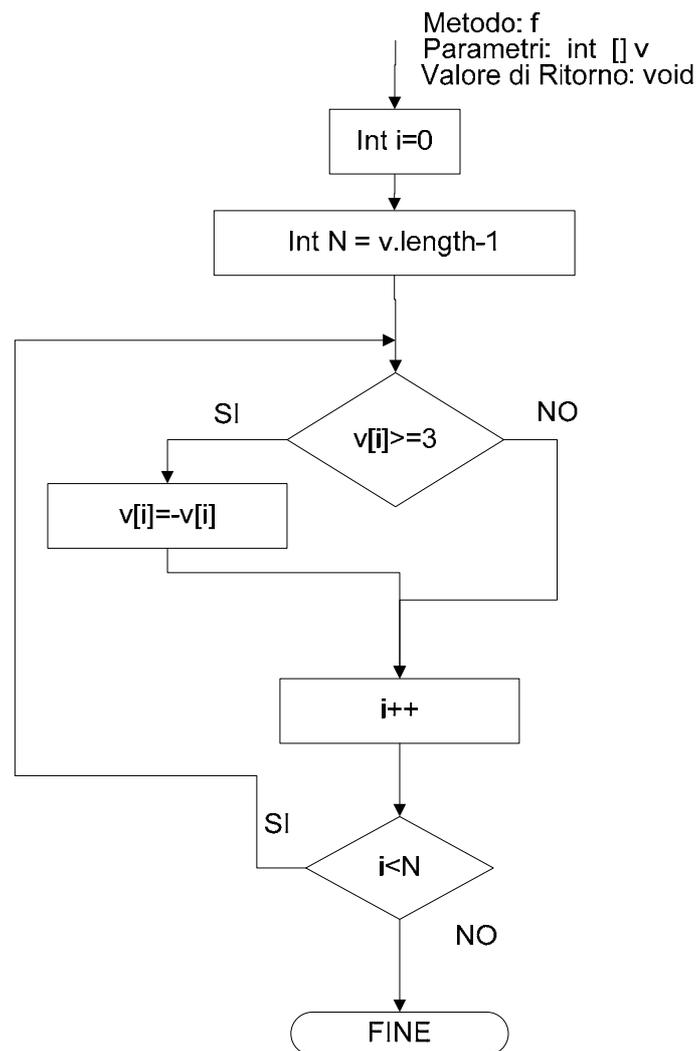
```

```

// stampo il catalogo
System.out.println(" Titolo\tQuantità");
for (int i = 0; i < lib_titolo.length; i++) {
    System.out.println(" " + lib_titolo[i] + "\t" +
lib_quantita[i]);
}
}
}

```

### Quesito 2: soluzione.



### Quesito 3: soluzione.

Output del programma:

```

2 5 10
2 -5 10

```

Per la giustificazione, si ricorre alla evoluzione dello stato del programma, identificando i contenuti dell'area locale ed area globale e seguendo la loro evoluzione. Attenzione alla inizializzazione di N in f: quanti cicli fa il for?