COMPITO DI LABORATORIO DI PROGRAMMAZIONE 2001-2002

17 Luglio 2002

Soluzione degli Esercizi

ESERCIZIO 1. Si consideri il seguente frammento di codice in linguaggio Java:

```
int i=0, j=0;
int[][] a=b;
boolean trovato=false;
while(i<a.length && !trovato) {
   j=0;
   while(j<a[0].length && !trovato) {
      if ( a[i][j]==0 )
         trovato=true;
      j++;
   }
   i++;
}
```

Determinare il valore finale delle variabili `i`, `j` e `trovato` nei seguenti casi:
1) `b` è un array bidimensionale di 2 righe e 2 colonne. Inoltre,
`b[0][0]=1, b[0][1]=2, b[1][0]=3, b[1][1]=4` ;
2) `b` è un array bidimensionale di 3 righe e 2 colonne. Inoltre,
`b[0][0]=1, b[0][1]=1, b[1][0]=1, b[1][1]=0, b[2][0]=0, b[2][1]=0`.

Soluzione

1) `i=2, j=2, trovato=false;`
2) `i=2, j=2, trovato=true`.

ESERCIZIO 2. Si consideri il seguente frammento di codice in linguaggio Java:

```
String s=stringa, c='''';
for(int i=0; i<s.length(); i++) {
   if(s.charAt(i)=='a' || s.charAt(i)=='b')
      c=c+s.charAt(i);
}
```

Determinare il valore finale della variabile `c` nei seguenti casi:
1) `stringa=''libellula'';`
2) `stringa=''albatro'';`
3) `stringa=''anaconda'';`
4) `stringa=''balena''.`

Soluzione

1) `c=''ba'';`
2) `c=''aba'';`
3) `c=''aaa'';`
4) `c=''baa''.`

ESERCIZIO 3. Scrivere un metodo in linguaggio Java che dato un array di interi restituisca `true` se tutti i suoi elementi sono identici, e `false` altrimenti.

Soluzione

```
public static boolean uguali(int[] a) {
  boolean equal = true;
```

```
  int i = 0;

  while(i<a.length-1 && equal) {
    if(a[i]!=a[i+1])
      equal=false;
    else
      i++;
  }
  return equal;
}
```

ESERCIZIO 4. Risolvere l'esercizio 3 usando la ricorsione.

Soluzione

```
public static boolean uguali_ric(int[] a, int i) {
  if(i==a.length-1) return true;
  else {
    if(a[i]!=a[i+1])
      return false;
    else
      return uguali_ric(a,i+1);
  }
}
```

ESERCIZIO 5. Si consideri la seguente classe Java Animale. Le variabili istanza sono la specie (ad esempio, tigre), il nome (ad esempio, Simba) e l'età. I metodi sono quelli che restituiscono i valori delle variabili istanza.

```
public class Animale {
   public Animale (String specie, String nome, int eta){
      this.specie=specie;
      this.nome=nome;
      this.eta=eta;
   }
   public String get_specie() {
      return this.specie;
   }
   public String get_nome() {
      return this.nome;
   }
   public int get_eta() {
      return this.eta;
```

```
    }
    private String specie, nome;
    private int eta;
}
```

Scrivere una nuova classe Contenitore_di_animali, dove ciascun contenitore di animali è caratterizzato dalle sue dimensioni (lunghezza, larghezza e altezza) e da un elenco di animali (il cui numero massimo deve essere impostato dal costruttore della classe), che abbia i seguenti metodi:

1. un metodo che aggiunge un animale ad un contenitore di animali;
2. un metodo che restituisce l'elenco dei nomi di tutti gli animali in un contenitore di animali.

Scrivere inoltre un breve programma di prova che illustri l'uso della classe Contenitore_di_animali e dei suoi metodi.

Soluzione

```
public class Contenitore_di_animali {

  public Contenitore_di_animali (int lunghezza, int larghezza, int
     altezza, int n) {
    this.lunghezza = lunghezza;
    this.larghezza = larghezza;
    this.altezza = altezza;
    this.animali = new Animale[n];
  }

  public void add_animale(Animale a) {
    if (count < animali.length) {
      animali[count] = a;
      count++;
    }
  }

  public String[] get_nomi_animali() {
    String[] s = new String[count];
    for (int i = 0; i < count; i++)
      s[i] = animali[i].get_nome();
    return s;
  }

  private int lunghezza, larghezza, altezza;
  private Animale[] animali;
  private int count = 0;

}
```

ESERCIZIO 6. Scrivere due sottoclassi, Gabbia e Vasca, della classe Contenitore_di_animali con le seguenti caratteristiche:

1. la classe Gabbia è caratterizzata da un habitat (es:savana);

2. la classe Vasca è caratterizzata dal tipo di acqua (es:acqua dolce) e dal flusso di acqua (misurato in litri al minuto) richiesto per il ricambio della stessa.

Le due classi, oltre ad ereditare i metodi della classe Contenitore_di_animali, hanno metodi che restituiscono l'habitat (classe Gabbia) ed il tipo di acqua e il flusso richiesto (classe Vasca). Scrivere inoltre un breve programma di prova che illustri l'uso delle classi Gabbia e Vasca.

Soluzione

```
public class Gabbia extends Contenitore_di_animali {

  public Gabbia(int lun, int lar, int alt, int n, String habitat) {
    super(lun, lar, alt, n);
    this.habitat = habitat;
  }

  public String get_habitat() {
    return habitat;
  }

  private String habitat;

}

public class Vasca extends Contenitore_di_animali {

  public Vasca(int lun, int lar, int alt, int n, String tipo, double flusso) {
    super(lun, lar, alt, n);
    this.tipo = tipo;
    this.flusso = flusso;
  }

  public String get_tipo_di_acqua() {
    return tipo;
  }

  public double get_flusso() {
    return flusso;
  }

  private String tipo;
```

```
  private double flusso;

}
```

ESERCIZIO 7. Uno zoo è composto da un insieme di gabbie e vasche al cui interno sono ospitati animali. Usando le classi precedentemente definite, scrivere una semplice classe Zoo, motivando le scelte effettuate. Scrivere inoltre un breve programma di prova che illustri l'uso della classe Zoo.

Soluzione

```
public class zoo {

  public zoo(String nome, int n) {
    this.nome = nome;
    contenitori = new Contenitore_di_animali[n];
  }

  public void add_contenitore(Contenitore_di_animali c) {
    if(count<contenitori.length) {
      contenitori[count] = c;
      count++;
    }
  }

  public void add_animale(Contenitore_di_animali c, Animale a) {
    contenitori[c].add_animale(a);
  }

  private String nome;
  private Contenitore_di_animali[] contenitori;
  private int count;
}
```