```java
/* TemaEsame_Schema_Inventato4: Esercizio 6 */

public class Scommesse {
    /*@ spec_public @*/ int[] budget;

    // INVARIANTI:
    // 1) budget ha una dimensione pari al numero di partite (5)
    //@ public invariant budget.length==5;
    // 2) la partita termina quando uno dei budget raggiunge il
    // valore 1, quindi sicuramente budget[i]>=1 per ogni i
    //@ public invariant (\forall int i; 0<=i && i<=4; budget[i]>=1);


    // POSTCONDIZIONI:
    // 1) Il budget relativo ad ogni partita è stato
    //    inizializzato al valore 5
    //@ ensures (\forall int i; 0<=i && i<5; budget[i]==5);
    public Scommesse() {
        budget = new int[] {5,5,5,5,5};

        //JML_1: violo INVARIANTE 1)
        //  budget = new int[] {5,5,5,5,5,10};

        //JML_2: violo INVARIANTE 2)
        //  budget = new int[] {0,0,0,0,0};
    }

    // funzione da asmeta:
    // " function check50 =
    // (budget(1)+budget(2)+budget(3)+budget(4)+budget(5)>=50) "
    // POSTCONDIZIONI:
    // 1) il valore ritornato è true se la somma dei budget è >=50
    //@ ensures (\result == true ==> (budget[0]+budget[1]+budget[2]+budget[3]+budget[4]) >=
50);
    // 2) il valore ritornato è false se la somma dei budget non è >=5
    //@ ensures (\result == false ==> ! ( (budget[0]+budget[1]+budget[2]+budget[3]+budget[4])
>= 50) );
    /*@ pure @*/
    public boolean check50() {
        return (budget[0]+budget[1]+budget[2]+budget[3]+budget[4]) >= 50;

        // JML_3: violo postcondizione 1)
        // return (budget[0]+budget[1]+budget[2]+budget[3]+budget[4]) < 50;

    }

    // funzione da asmeta:
    // "function check1 =
    // ( forall $b in NumPartita with budget($b) > 1 )"
    // POSTCONDIZIONI:
    // 1) Se tutti i valori di ogni posizione dell'array budget sono >1, il risultato
restituito è true
    //@ ensures (\result == true ==> ((budget[0]>1) && (budget[1]>1) && (budget[2]>1) &&
(budget[3])>1 && (budget[4])>1 ));
    // 2) Il risultato è false se non vale la condizione precedente
    //@ ensures (\result == false ==> !((budget[0]>1) && (budget[1]>1) && (budget[2]>1) &&
(budget[3])>1 && (budget[4])>1 ));
```

```
53      /*@ pure @*/
54      public boolean check1() {
55          return (budget[0]>1) && (budget[1]>1) && (budget[2]>1) && (budget[3])>1 &&
  (budget[4])>1;
56
57          //JML_4: violo postcondizione 1)
58          // return (budget[0]>7) && (budget[1]>1) && (budget[2]>1) && (budget[3])>1 &&
  (budget[4])>1;
59
60      }
61
62      // PRECONDIZIONI:
63      // 1) Scommesse e risultati possono contenere solo i seguenti valori: 0=PARI, UNO = 1,
  DUE=2
64      //@ requires (\forall int i; 0<=i && i<5; 0<= scommesse[i] && scommesse[i]<=2 && 0<=
  risultati[i] && risultati[i]<=2);
65      // 2) Scommesse e risultati devono avere dimensione 5
66      //@ requires scommesse.length==5 && risultati.length==5;
67      // POSTCONDIZIONI:
68      // Tutte le prossime postcondizioni valgono solamente se la scomessa può essere fatta,
  quindi saranno
69      // tutte precedute da "!this.check50() && this.check1()"
70      // 1) Se vinco la scommessa in posizione i, allora il budget in posizione i aumenta di 1
71      //@ ensures (!\old(this.check50()) && \old(this.check1())) ==> (\forall int i; 0<=i &&
  i<budget.length; ( ( scommesse[i]==risultati[i] ) ==> ( budget[i]==\old(budget[i])+1) ) );
72      // 2) Se perdo la scommessa in posizione i, ma il risultato della partita è pari allora il
  budget in i diminuisce di 1
73      //@ ensures (!\old(this.check50()) && \old(this.check1())) ==> (\forall int i; 0<=i &&
  i<budget.length; ( (!(scommesse[i]==risultati[i])&&risultati[i]==0) ==> (
  budget[i]==\old(budget[i])-1) ) );
74      // 3) Se perdo la scommessa in posizione i e il risultato non è pari, allora il budget in
  i diminuisce di 2
75      //@ ensures (!\old(this.check50()) && \old(this.check1())) ==> (\forall int i; 0<=i &&
  i<budget.length; ( (!(scommesse[i]==risultati[i])&& !(risultati[i]==0)) ==> (
  budget[i]==\old(budget[i])-2) ) );
76      public void giocaScommessa(int[] scommesse, int[] risultati) {
77          // la scommessa può essere fatta solo se non sono stati raggiunti
78          // 50€ in totale (!check50()) e se il budget in ogni posizione è
79          // maggiore di 1 (check1())
80          if ( !this.check50() && this.check1()) {
81
82              // SE FACCIO IL RAC CON QUESTO LOOP INVARIANT RESTITUISCE:
83              // "A catastrophic JML internal error occurred.  Please report the bug with as
  much information as you can.
84              // Reason: MISMATCHED BLOCKS"
85  //          /*@ loop_invariant
86  //            0<=i && i<=5
87  //            &&
88  //            ( ( \forall int j; 0<=j && j<i; ( (scommesse[j]==risultati[j]) ==> (
  budget[j]==\old(budget[j])+1 ) ) ) )
89  //            &&
90  //            ( \forall int j; 0<=j && j<i; ( !(scommesse[j]==risultati[j]) && risultati[j]==0
  ) ==> ( budget[j]==\old(budget[j])-1 ))
91  //            &&
92  //            ( \forall int j; 0<=j && j<i; ( !(scommesse[j]==risultati[j]) && !
  (risultati[j]==0) ) ==> ( budget[j]==\old(budget[j])-2 ))
93  //           ;
```

```
94 //           @*/
95             for (int i=0; i<5; i++) {
96                 // se ho indovinato vinco 1€
97                 if(scommesse[i]==risultati[i])
98                     budget[i]+=1;
99                 // se non ho indovinato eseguo l'else
100                else {
101                    // se il risultato effettivo PARI=0 e ho sbagliato
102                    // allora perdo solamente 1€
103                    if(risultati[i]==0)
104                        budget[i]-=1;
105                    // altrimenti perdo 2€
106                    else
107                        budget[i]-=2;
108                }
109            }
110        }
111    }
112
113    public static void main(String[] args) {
114
115        /*NB: qui di seguito inserisco le violazioni dei
116              contratti. In particolare utilizzerò:
117              JML_Numero = per indicare nel codice della classe ciò che deve
118              essere decommentato per poter poi violare il contratto nel main
119              JML_Check = per indicare il codice del main che chiama il pezzo di
120              codice della classe che fa scattare la violazione del contratto
121        */
122
123        //JML_1_check:
124        // Scommesse s = new Scommesse();
125 //      "18: JML invariant is false on leaving method Scommesse.Scommesse()"
126 //      //@ public invariant budget.length==5;
127 //                ^
128        //JML_2_check:
129 //      18: JML invariant is false on leaving method Scommesse.Scommesse()
130 //      //@ public invariant (\forall int i; 0<=i && i<=4; budget[i]>=1);
131 //                ^
132
133        //JML_3_check:
134        // s.giocaScommessa(new int[] {1,1,1,1,1}, new int[]{2,1,0,1,1});
135 //      37: JML postcondition is false
136 //      // //@ ensures (\result == true ==>
    (budget[0]+budget[1]+budget[2]+budget[3]+budget[4]) >= 50);
137
138
139        //JML_4_check:
140        // s.giocaScommessa(new int[] {1,1,1,1,1}, new int[]{2,1,0,1,1});
141 //      54: JML postcondition is false
142 //      //@ ensures (\result == false ==> !((budget[0]>1) && (budget[1]>1) && (budget[2]>1) &&
    (budget[3])>1 && (budget[4])>1 ));
143 //          ^
144 //
145
146        // TEST DI DEBUG GENERALE
147        /*
148         * 1 - perdo 2€
```

```
149          * 2 - vinco 1€
150          * 3 - perdo 1€
151          * 4 - vinco 1€
152          * 5 - vinco 1€
153          */
154         Scommesse s = new Scommesse();
155         s.giocaScommessa(new int[] {1,1,1,1,1}, new int[]{2,1,0,1,1});
156         System.out.println("budget[0]: " + s.budget[0]); // 3
157         System.out.println("budget[1]: " + s.budget[1]); // 6
158         System.out.println("budget[2]: " + s.budget[2]); // 5
159         System.out.println("budget[3]: " + s.budget[3]); // 6
160         System.out.println("budget[4]: " + s.budget[4]); // 6
161         System.out.println();
162
163         // ora vinco sempre fino a quando la partita si conclude
164         s.giocaScommessa(new int[] {1,1,1,1,1}, new int[]{1,1,1,1,1});
165         System.out.println("budget[0]: " + s.budget[0]);
166         System.out.println("budget[1]: " + s.budget[1]);
167         System.out.println("budget[2]: " + s.budget[2]);
168         System.out.println("budget[3]: " + s.budget[3]);
169         System.out.println("budget[4]: " + s.budget[4]);
170         System.out.println();
171
172         s.giocaScommessa(new int[] {1,1,1,1,1}, new int[]{1,1,1,1,1});
173         System.out.println("budget[0]: " + s.budget[0]);
174         System.out.println("budget[1]: " + s.budget[1]);
175         System.out.println("budget[2]: " + s.budget[2]);
176         System.out.println("budget[3]: " + s.budget[3]);
177         System.out.println("budget[4]: " + s.budget[4]);
178         System.out.println();
179
180         s.giocaScommessa(new int[] {1,1,1,1,1}, new int[]{1,1,1,1,1});
181         System.out.println("budget[0]: " + s.budget[0]);
182         System.out.println("budget[1]: " + s.budget[1]);
183         System.out.println("budget[2]: " + s.budget[2]);
184         System.out.println("budget[3]: " + s.budget[3]);
185         System.out.println("budget[4]: " + s.budget[4]);
186         System.out.println();
187
188         s.giocaScommessa(new int[] {1,1,1,1,1}, new int[]{1,1,1,1,1});
189         System.out.println("budget[0]: " + s.budget[0]);
190         System.out.println("budget[1]: " + s.budget[1]);
191         System.out.println("budget[2]: " + s.budget[2]);
192         System.out.println("budget[3]: " + s.budget[3]);
193         System.out.println("budget[4]: " + s.budget[4]);
194         System.out.println();
195
196         s.giocaScommessa(new int[] {1,1,1,1,1}, new int[]{1,1,1,1,1});
197         System.out.println("budget[0]: " + s.budget[0]);
198         System.out.println("budget[1]: " + s.budget[1]);
199         System.out.println("budget[2]: " + s.budget[2]);
200         System.out.println("budget[3]: " + s.budget[3]);
201         System.out.println("budget[4]: " + s.budget[4]);
202         System.out.println();
203
204         // ora il budget è 50 e i valori non dovrebbero più aumentare
205         s.giocaScommessa(new int[] {1,1,1,1,1}, new int[]{1,1,1,1,1});
```

Scommesse.java

```
206         System.out.println("budget[0]: " + s.budget[0]);
207         System.out.println("budget[1]: " + s.budget[1]);
208         System.out.println("budget[2]: " + s.budget[2]);
209         System.out.println("budget[3]: " + s.budget[3]);
210         System.out.println("budget[4]: " + s.budget[4]);
211         System.out.println();
212     }
213
214 }
215
```