Scheda di lavoro sul debugging

Insegnamento di Didattica dell'informatica

19 maggio 2020

1 Controllo su un array

1.1 Il problema

Scrivere un algoritmo che riceve un array A di dimensione N (gli indici dell'array vanno da 1 a N), e restituisce `true` se tutti gli elementi dell'array sono uguali, e `false` se non sono tutti uguali.

1.2 Le soluzioni proposte

Soluzione 1

```
all-equals(A, N){
    ok ← true
    for i from 1 to N do
        if (A[i] ≠ A [i + 1]) then
            ok ← false
    return ok
}
```

Soluzione 2

```
all-equals(A, N){
    ok ← true
    for i from 1 to N-1 do
        if (A[i] ≠ A [i + 1]) then
            ok ← false
        else
            ok ← true
    return ok
}
```

Soluzione 3

```
all-equals(A, N){
    ok ← true
    for i from 1 to N-1 by 2 do
         if (A[i] ≠ A [i + 1]) then
             ok ← false
   return ok
}
```

Soluzione 4

```
all-equals(A, N){
    ok ← true
    for i from 1 to N-1 by 2 do
        if (A[i] ≠ A [i + 1]) then
            ok ← false
    for i from 2 to N-1 by 2 do
        if (A[i] ≠ A [i + 1]) then
            ok ← false
    return ok
}
```

Soluzione 5

```
all-equals(A, N){
    ok ← true
    for i from 1 to N-1 do
        if (A[i] ≠ A [i + 1]) then
            ok ← false
    return ok
}
```

Soluzione 6

```
all-equals(A, N){
    count ← 0
    for i from 2 to N do
        if (A[i] = A [i + 1]) then
            count ← count + 1
    if (count = N) then
        return true
   else
        return false
}
```

1.3 Esercizio

Lavorando a coppie, per ciascuna delle soluzioni:

- determinate se è corretta o no
- se non è corretta, individuate gli errori

2 Condizione per anno bisestile

2.1 Il problema

Scrivere un metodo (funzione) che riceve come argomento un intero corrispondente a un anno e restituisce `true` se l'anno corrispondente è bisestile, `false` altrimenti. Un anno è bisestile se soddisfa una delle seguenti condizioni:

- è un anno non secolare divisibile per 4 (ad es. 1964);
- è un anno secolare divisibile per 400 (ad es. 2000).

Il simbolo % è il simbolo dell'operatore di modulo (resto della divisione intera).
Il simbolo ! è il simbolo dell'operatore di negazione logica (NOT), e != è il simbolo per $\neq$.

2.2 Le soluzioni proposte

Soluzione 1

```
isBissextile(anno) {
   if(anno % 4 = 0) {
         return true;
   }
   else if(anno % 400 = 0) {
      return true;
   }
   return false;
}
```

Soluzione 2

```
isBissextile(anno) {
   bisest <- false;
   if (anno % 4 = 0 OR anno % 400 = 0)
     bisest <- true;
   return bisest;
}
```

Soluzione 3

```
isBissextile( anno){
   if(anno%100 != 0){
      if(anno%4 = 0)
         return true;
   }
   else {
      if(anno%400 = 0)
         return true;
   }
   return false;
}
```

Soluzione 4

```
isBissextile( anno){
   if(anno%100 != 0)
      if(anno%4 = 0)
         return true;
   else
      if(anno%400 = 0)
         return true;
   return false;
}
```

Soluzione 5

```
isBissextile(anno){
   if (anno%100 = 0){
      if (anno%400 = 0)
         return true;
   }
   if (anno%100 != 0){
      if (anno%4 = 0)
         return true;
   }
   return false;
}
```

Soluzione 6

```
isBissextile(anno){
   if(anno%4 = 0)
      return true;
   else if(anno%4 = 0 AND anno%100 = 0)
      return true;
   else return false;
}
```

Soluzione 7

```
isBissextile(anno) {
   if (anno % 100 = 0 AND anno % 400 = 0)
      return true;
   else if (anno % 4 = 0)
      return true;
   else
      return false;
}
```

Soluzione 8

```
isBissextile(anno){
   if((anno%4 = 0 AND anno%100 != 0) OR anno%400 = 0)
      return true;
   else
      return false;
}
```

2.3 Esercizio

Lavorando a coppie, per ciascuna delle soluzioni determinate se è corretta o no e, se non è corretta, individuate gli errori.

3 Tecniche per facilitare l'individuazione degli errori in un programma

Ecco una breve rassegna di possibili tecniche da proporre:

a. istruzioni di print per visualizzare il contenuto di una o più variabili

b. istruzioni di print per verificare il flusso di esecuzione (scoprire in quale caso del controllo condizionale ci si trova o quante iterazioni di un ciclo sono state fatte)

c. rappresentazione grafica dello stato della memoria (es. di un array o di un albero)

a	b	c	d	e

d. inserimento di input critici (input su cui il programma funziona / non funziona correttamente)

e. esecuzione passo-passo con carta e penna (tabella con i valori delle variabili per ogni riga di un blocco di codice):

linea di codice	var 1	var 2	...

f. studio di invarianti (condizioni che, in presenza di un comportamento iterativo come un ciclo o una chiamata ricorsiva, rimangono vere prima, durante e al termine delle varie iterazioni)

g. asserzioni (condizioni che devono essere sempre vere se il programma è corretto, e che bloccano la sua esecuzione altrimenti)

4 Consegna

A gruppi di quattro (due coppie del lavoro precedente), per ogni tecnica (Sez. 3), individuate le soluzioni per il problema sull'array e quelle per l'anno bisestile per le quali la proporreste.

tecnica	soluzioni
a.	
b.	
c.	
d.	
e.	
f.	
g.	

Scegliete almeno tre fra le tecniche viste e per ciascuna
- scegliete una procedura da correggere per la quale la proporreste,
- motivate perché vi sembra un approccio adatto,
- illustrate come immaginate si possa svolgere l'interazione con lo studente (facendo riferimento alle slide sul feedback).