Programmazione Procedurale

Tipi Strutturati
Tecniche Algoritmiche

versione 4.2

1

Sommario

- Alcune Tecniche Notevoli
- Calcolo di Massimi e Minimi
- Verifica di Condizioni
  - ⇨Verifica di Condizioni Locali
  - ⇨Verifica di Condizioni Globali
- Composizione di Tecniche

2

Tecniche Notevoli

- Approfondiamo alcune tecniche notevoli
  - ⇨collegate alla gestione di collezioni di valori attraverso array
- In particolare
  - ⇨ricerca del massimo e del minimo tra un insieme di valori
  - ⇨verifica di condizioni locali e globali (>>)

3

Tecniche Notevoli

- Nota
  - ⇨nel programma delle temperature vengono utilizzate varie tecniche notevoli già note
- Calcolo della media
  - ⇨viene utilizzata la stessa tecnica già vista: somma con accumulatore (la media è sempre calcolabile, perché ci sono sempre N valori)
- Conteggio delle temperature negative
  - ⇨anche in questo caso viene utilizzata la tecnica già vista: conteggio con contatore

4

```
float calcolaMedia(float arrayDati[N]){
    float somma = 0;
    for (int i = 0; i < N; i++) {
        somma += arrayDati[i];
    }
    return somma / N;
}

int contaNegativi(float arrayDati[N]){
    int conta = 0;
    for (int i = 0; i < N; i++) {
        if(arrayDati[i] < 0) {
            conta++;
        }
    }
    return conta;
}
```

5

Calcolo di Massimi e Minimi

ATTENZIONE
alle nuove
tecniche algoritmiche
notevoli

- Due nuove tecniche algoritmiche
  - ⇨ricerca del minimo e del massimo
  - ⇨si tratta di due varianti della stessa tecnica
- Ricerca del minimo: idea
  - ⇨viene cercata la posizione dell'elemento con valore minimo attraverso una funzione
  - ⇨tecnica comune utilizzando gli array
  - ⇨dalla posizione è possibile immediatamente risalire al valore

6

Calcolo di Massimi e Minimi

ATTENZIONE
ipotesi iniziale
+ verifica

- Strategia per la ricerca del minimo
  - ⇨"ipotesi e verifica"
  - ⇨all'inizio il minimo corrente è l'elemento in posizione 0 (il primo elemento) (ipotesi iniziale da verificare)
  - ⇨scandisco l'insieme con un ciclo per esaminare i restanti elementi (verifica)
  - ⇨analizzo l'array tenendo traccia della posizione del "minimo corrente" (cioè del valore minimo incontrato fino a quel punto)

7

Calcolo di Massimi e Minimi

- Strategia per la ricerca del minimo (continua)
  - ⇨nel ciclo, confronto il minimo corrente con ciascuno degli altri elementi
  - ⇨se trovo un elemento di valore minore del minimo corrente, l'elemento diventa il nuovo minimo corrente (aggiorno la posizione)
  - ⇨al termine del ciclo ho la posizione del minimo complessivo

8

>> temperaturaMinMaxMedia.cpp

Esempio: Temperature

```
int posizioneMin(float arrayDati[N]){
    int pos;
    pos = 0;
    for (int i = 1; i < N; i++) {
        if (arrayDati[i] < arrayDati[pos]){
            pos = i;
        }
    }
    return pos;
}
```

per cominciare il minimo corrente è in posizione 0

confronto il minimo corrente arrayDati[pos] con tutti gli altri elementi e se ne trovo uno minore aggiorno la posizione del minimo corrente

9

Esempio

- Esempio
  - ⇨supponiamo che l'array sia {5.1, 4.0, 7.3, 2.0, 8.9}
- Simulazione

```
pos = 0;
for (int i = 1; i < N; i++){
  if (arrayDati[i] < arrayDati[pos]){
     pos = i;
  }
}
```

0 1 2 3 4

inizialmente: pos = 0
al termine: pos = 3

i	pos	arrayDati[i]	arrayDati[pos]	nuovo valore di pos
1	0	4.0	5.1	1
2	1	7.3	4.0	1
3	1	2.0	4.0	3
4	3	8.9	2.0	3

10

Esempio: Temperature

- Annotazioni
  - ⇨la strategia funziona sia nel caso in cui ci sia un unico minimo (es: {5, 2, 7, 9}) che nel caso in cui esistono più valori uguali e pari al minimo (es: {5, 3, 7, 3, 4})
  - ⇨nel secondo caso la funzione restituisce la posizione del primo tra i valori pari al minimo
  - ⇨la strategia per la ricerca del massimo è del tutto uguale (unica differenza: la condizione dell'if nel ciclo)

11

Esempio: Temperature

```
int posizioneMax(float arrayDati[N]){
    int pos = 0;
    for (int i = 1; i < N; i++) {
        if (arrayDati[i] > arrayDati[pos]) {
            pos = i;
        }
    }
    return pos;
}
```

>> temperaturaMinMaxMedia.cpp

12

Verifica di Condizioni

- Per introdurre la verifica di condizioni
  - ⇨vediamo un ulteriore esempio
- Specifiche
  - ⇨ dati i 5 numeri estratti su una ruota del Lotto (es: Roma), acquisire dalla tastiera i dati di una giocata di 5 numeri
  - ⇨ verificare che la giocata sia corretta, ovvero: (a) i numeri siano tutti diversi; (c) i numeri siano tutti tra 1 e 90
  - ⇨ se la giocata è corretta, verificare se contiene almeno un numero superiore a 80
  - ⇨ e calcolare e stampare il punteggio della giocata rispetto ai numeri estratti

13

Verifica di Condizioni

- Nel programma sul lotto
  - ⇨alcune tecniche notevoli
  - ⇨verifica di condizioni su una collezione
- Verifica di condizioni
  - ⇨due tipologie
  - ⇨verifica di condizioni “locali”
  - ⇨verifica di condizioni “globali”
  - ⇨ne diamo due versioni, una tradizionale e una basata sulla programmazione non strutturata, generalmente più semplice

14

Verifica di Condizioni

- Condizione “locale”
  - ⇨condizione su un numero limitato di valori vicini della collezione
  - ⇨es: la giocata contiene almeno un numero superiore a 80
  - ⇨es: la giocata contiene almeno un numero superiore al successivo
  - ⇨es: la giocata contiene almeno tre numeri consecutivi minori di 10

15

Verifica di Condizioni

- Condizione “globale”
  - ⇨condizione sulla collezione nel complesso
  - ⇨es: tutti i numeri sono compresi tra 1 e 90
  - ⇨es: tutti i numeri sono diversi tra di loro
  - ⇨es: la giocata contiene tre numeri minori di 10 (in posizioni qualsiasi)

16

Verifica di Condizioni

- Differenza tra i due casi
  - ⇨la verifica di condizioni locali è algoritmicamente intuitiva
  - ⇨posso effettuarla durante la scansione
  - ⇨la verifica di condizioni globali è meno intuitiva
  - ⇨sembrerebbe che sia necessario esaminare tutti i valori contemporaneamente (?)

17

Verifica di Condizioni Locali

- Un Esempio
  - ⇨verificare che la giocata contenga almeno un numero superiore a 80
- Idea
  - ⇨analizzo con un ciclo i valori della giocata
  - ⇨non appena trovo un numero che soddisfa, mi fermo e restituisco vero, altrimenti restituisco falso
  - ⇨atteggiamento “greedy”

18

Verifica di Condizioni Locali

- La versione greedy
  - ⇨programm. non strutturata

visito per verificare se la cond. è vera

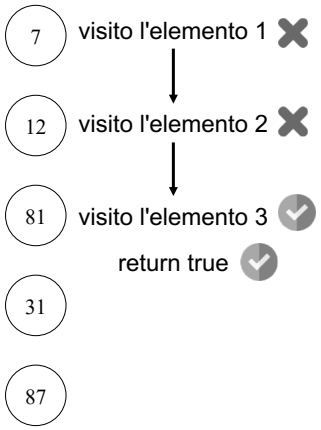
```
bool numeriAlti(int giocata[N]) {
    for (int i = 0; i < N; i++) {
      if (giocata[i] > 80) {
          return true;
      }
    }
    return false;
}
```

altrimenti decido che è falsa

19

Verifica di Condizioni Locali

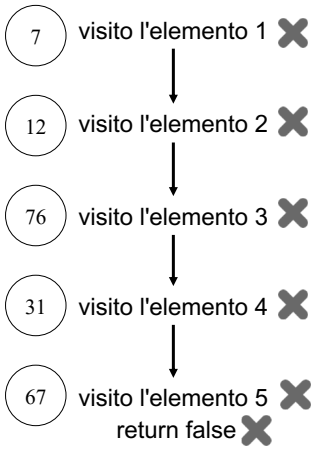
- La strategia
  - ⇨cerco un elemento di valore superiore a 80
  - ⇨caso positivo



20

Verifica di Condizioni Locali

- La strategia (cont.)
  - ⇨cerco un elemento di valore superiore a 80
  - ⇨caso negativo



21

Verifica di Condizioni Locali

- Attenzione allo sconfinamento
  - ⇨se la condizione locale richiede di ispezionare più di un elemento alla volta, è necessario fermare il ciclo prima del solito
- Esempio
  - ⇨verificare se la giocata contiene almeno una volta tre elementi superiori ad 80 consecutivi

22

Versione Non Strutturata (Semplice)

```
bool treNumeriAltiConsecutivi(int giocata[N]) {
    for (int i = 0; i < N - 2; i++) {
        if (giocata[i] > 80 &&
            giocata[i + 1] > 80 &&
            giocata[i + 2] > 80) {
              return true;
        }
    }
    return false;
}
```

const int N = 5;
int giocata[N] = {2, 34, 81, 88, 90};

i = 0 -> 2, 34, 81

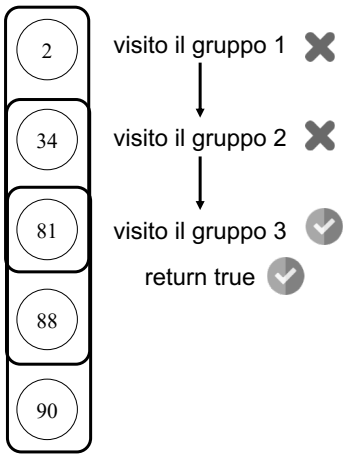
i = 1 -> 34, 81, 88

i = 2 -> 81, 88, 90 -> true

23

Verifica di Condizioni Locali

- La strategia
  - ⇨cerco tre numeri consecutivi di valore superiore a 80
  - ⇨caso positivo



24

Versione Errata

```
bool treNumeriAltiConsecutivi(int giocata[N]) {
    for (int i = 0; i < N; i++) {
        if (giocata[i] > 80 &&
            giocata[i + 1] > 80 &&
            giocata[i + 2] > 80) {
              return true;
        }
    }
    return false;
}
```

const int N = 5;
int giocata[N] = {2, 34, 56, 88, 90};

i = 0 -> 2, 34, 56

i = 1 -> 34, 56, 88

i = 2 -> 56, 88, 90

i = 3 -> 88, 90, ?

i = 4 -> 90, ?, ?

25

Verifica di Condizioni Globali

- Apparentemente più complessa
- Due possibili tecniche
- Conteggio
  - ⇨conto gli elementi che soddisfano la condizione e poi confronto il risultato con un numero di riferimento
- Riduzione a condizione locale
  - ⇨mi riduco se possibile a verificare una condizione locale equivalente

26

Verifica di Condizioni Globali

- Conteggio
  - ⇨scandisco per contare quanti elementi soddisfano la condizione
  - ⇨al termine verifico se la condizione globale è soddisfatta
- Esempi
  - ⇨es: la giocata contiene almeno 3 numeri minori di 10; conto i numeri < 10; al termine verifico se sono almeno 3
  - ⇨es: tutti i numeri sono compresi tra 1 e 90; conto i numeri tra 1 e 90 e confronto con il numero totale di numeri;

27

Verifica di Condizioni Globali

- Nota
  - ⇨questo approccio è relativamente inefficiente
  - ⇨infatti, per contare devo sempre esaminare tutti gli elementi
  - ⇨utilizzo tipicamente un ciclo for per scandire la giocata completamente
  - ⇨in molti casi questo è inutile

28

Verifica di Condizioni Globali

ATTENZIONE
alla riduzione

- Riduzione a condizione locale
  - ⇨idea: molto spesso una condizione globale è la negazione di una condizione locale
  - ⇨es: tutti i numeri sono compresi tra 1 e 90 >> nessun numero è <1 o >90
  - ⇨es: tutti i numeri sono diversi >> non esiste un numero che compare più volte nella giocata
  - ⇨posso quindi verificare la condizione locale

29

Verifica di Condizioni Globali

- Riduzione (continua)
  - ⇨devo però ragionare al contrario: per verificare se la cond. globale è soddisfatta, devo verificare se la cond. locale è violata
  - ⇨scandisco l'array, e se verifico che localmente la condizione è violata (c'è un numero <1 oppure >90), vuol dire che la condizione globale non è soddisfatta

30

Versione non Strutturata

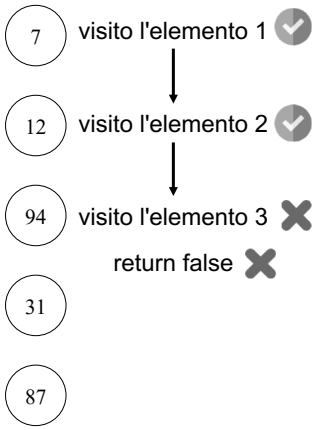
```
bool valoriAmmessi(int giocata[N]) {
  for (int i = 0; i < N; i++) {
    if (giocata[i] < 1 || giocata [i] > 90){
       return false;
    }
  }
  return true;
}
```

```
// Per confrontare con una verifica
// di condizione locale:
bool numeriAlti(int giocata[N]) {
   for (int i = 0; i < N; i++) {
     if (giocata[i] > 80) {
        return true;
     }
   }
   return false;
}
```

31

Verifica di Condizioni Globali

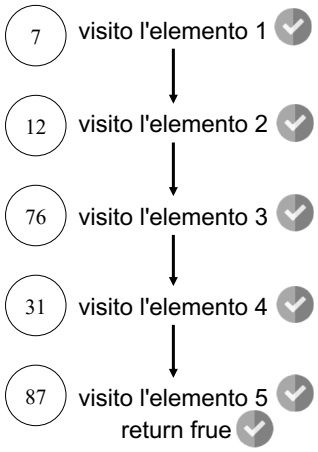
- La strategia
  - ⇨cerco un elemento che viola la condizione (NON compreso tra 1 e 90)
  - ⇨caso positivo



32

Verifica di Condizioni Globali

- La strategia (cont.)
  - ⇨cerco un elemento che viola la condizione (NON compreso tra 1 e 90)
  - ⇨caso negativo



33

Composizione di Tecniche

- Riassumiamo le principali tecniche viste
  - ⇨somma con accumulatore
  - ⇨conteggio con contatore
  - ⇨ricerca del minimo e del massimo
  - ⇨verifica di condizioni locali e globali e casi particolari (es: ricerca di un elemento)

34

Composizione di Tecniche

- Utilizzo delle tecniche
  - ⇨in fase di progetto dell'algoritmo, procedo per decomposizioni ("dall'alto in basso")
  - ⇨utilizzo sottoprogrammi opportuni per eseguire le operazioni individuate
  - ⇨ripeto il procedimento quando necessario
  - ⇨cerco però di ridurmi, quando possibile, a tecniche note, e di riutilizzarne il codice
  - ⇨in questo caso procedo "dal basso in alto"

35

Composizione di Tecniche

ATTENZIONE alla composizione delle tecniche algoritmiche

- Importante
  - ⇨questo richiede di saper riconoscere quando un sottoproblema può essere risolto componendo più tecniche algoritmiche note
- La capacità di comporre le tecniche
  - ⇨è altrettanto importante rispetto alla loro conoscenza
  - ⇨in generale, è sempre preferibile cercare una soluzione basata sulla composizione di tecniche semplici che riscrivere il codice integralmente

36

Il Gioco del Lotto: Esempio n.1

- Verifica che i numeri siano tutti diversi
  - ⇨per la soluzione posso comporre due tecniche algoritmiche notevoli
  - ⇨verifica di condizione globale per riduzione (non esiste un numero che compare più di una volta)
  - ⇨conteggio (conto il numero di occorrenze di ogni elemento della giocata)

37

Numeri Tutti Diversi

```
int contaOccorrenze (int giocata[N], int elem) {
  int conta = 0;
  for (int i = 0; i < N; i++) {
    if (giocata[i] == elem) {
       conta++;
    }
  }
  return conta;
}
```

38

Numeri Tutti Diversi

```
bool tuttiDiversi(int giocata[N]) {
  for (int i = 0; i < N; i++) {
      if(contaOccorrenze(giocata, giocata[i]) > 1){
          return false;
      }
  }
  return true;
}
// Più in dettaglio:
bool tuttiDiversi(int giocata[N]) {
  for (int i = 0; i < N; i++) {
      int numero = giocata[i];
      int compare = contaOccorrenze(giocata, numero);
      if (compare > 1){
          return false;
      }
  }
  return true;
}
```

39

Numeri Tutti Diversi

- Nota
  - ⇨il codice è scritto frammentando la soluzione in due sottoprogrammi
  - ⇨una funzione apposita per il conteggio delle occorrenze
  - ⇨avrei anche potuto evitare di separare il codice per il conteggio delle occorrenze
  - ⇨ma il risultato sarebbe stato di qualità inferiore (cicli nidificati)

40

Numeri Tutti Diversi

```
bool tuttiDiversi(int giocata[N]) {
  for (int i = 0; i < N - 1; i++) {
     for (int j = i + 1; j < N; j++) {
       if (giocata[i] == giocata[j]) {
           return false;
       }
     }
   }
  return true;
}
```

Attenzione alla gestione degli indici: POTENZIALE FONTE DI ERRORI

41

Il Gioco del Lotto: Esempio n.2

- Calcolo del punteggio della giocata
  - ⇨ date due collezioni, verificare quanti elementi della seconda sono contenuti nella prima
- Idea
  - ⇨ scandisco il secondo array per contare gli elementi contenuti nel primo
  - ⇨ dato giocata[i], lo cerco in estrazione >> per farlo ho una funzione pronta (cercaPosizione)
  - ⇨ restituisce la posizione del numero, se viene trovato, oppure il valore convenzionale -1

42

Punteggio della Giocata

```
int cercaPosizione(int array[N], int elem) {
   for(int i = 0; i < N; i++) {
      if (array[i] == elem) {
          return i;
      }
   }
    return -1;
}
```

43

Punteggio della Giocata

```
int valoriComuni(int estrazione[N], int giocata[N]){
  int conta = 0;
  for (int i = 0; i < N; i++) {
    if (cercaPosizione(estrazione, giocata[i]) != -1) {
      conta++;
    }
  }
  return conta;
}
```

verifica se l'i-esimo numero della giocata (giocata[i]) è contenuto o meno nell'array dei numeri estratti (estrazione)

44

Riassumendo

- Alcune Tecniche Notevoli
- Calcolo di Massimi e Minimi
- Verifica di Condizioni
  - ⇨Verifica di Condizioni Locali
  - ⇨Verifica di Condizioni Globali
- Composizione di Tecniche

45

Termini della Licenza

46