Palestra di algoritmi - day 11

31/01/2023

Ricorsione

Ricorsione

Risolvere un problema dividendolo in sottoproblemi.

Ogni sottoproblema:

- ha la struttura identica al problema originale
- è più piccolo di quello originale

Esempio 1 - sommatoria

Vogliamo sommare i numeri da 1 a n.

Se per esempio n=5 vogliamo fare 1+2+3+4+5.

Si può risolvere con un semplice ciclo for, lo so, ma dobbiamo partire da qualcosa di semplice per arrivare a casi più complessi in cui serve la ricorsione per forza.

Esempio 1 - sommatoria

Possiamo dividere il problema in:

- parto da n (5 nel nostro caso) e lo aggiungo alla mia variabile somma
- risolvo il problema sui numeri rimasti, fin quando non finisco i numeri

La ricorsione in codice

Ma come si fa tutto ciò in codice?

In c++ possiamo scrivere funzioni che chiamano sé stesse.

Esempio:

```
void stampaCiao(){
	cout << “ciao\n”;
	stampaCiao();
}
```

Questa funzione stampa ciao all’infinito, perchè continua a chiamare sé stessa senza mai fermarsi.

La ricorsione - punti chiave

Per scrivere una funzione ricorsiva quindi:

- dobbiamo risolvere un problema che può essere scomposto in problemi più piccoli
- ci devono essere dei “casi base”, ossia delle condizioni per cui la ricorsione si ferma, senza continuare all’infinito

Nel nostro esempio, se partiamo dal 5, la ricorsione si deve fermare a 0, perchè non c’è più nulla da sommare, se non 0.

Se partissimo da 0 invece dovrebbe fermarsi una volta che arriva a n, perché non vogliamo sommare i numeri successivi.

Esempio 1 - sommatoria - codice

```
int sommatoria(int n) {
	if(n<=0) { //ho finito i numeri da sommare
		return 0;
	} else{
		return n+sommatoria(n-1); //ritorno n + la somma di tutti gli altri numeri
	}
}
```

Nel main: cout << sommatoria(5);

Esempio 2 - fattoriale

```
int fattoriale(long long int n){
        if(n<=1) return 1;
        else return n*fattoriale(n-1);
}
```

Nel main: cout << fattoriale(10)

Praticamente uguale alla sommatoria

Esempio 3 - Fibonacci

La serie di Fibonacci è una serie di numeri in cui i primi due sono 1, mentre tutti gli altri sono dati dalla somma dei due numeri precedenti.

In forma di funzione:
Fibonacci(1) = 1
Fibonacci(2) = 1
Fibonacci(n) = Fibonacci(n-1) + Fibonacci(n-1)

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89,144, 233, 377, 610, 987,1597, 2584, 4181, …

Esempio 3 - Fibonacci - codice

```
int fibonacci(int n) {
	if(n==1 || n==2) {
		return 1;
	} else if(n>2) {
		return fibonacci(n-1) + fibonacci(n-2);
	}
}
```

Esercizio

Scrivere una funzione ricorsiva che stampi tutte le possibili stringhe di numeri binari di lunghezza n.

Esempio:

n = 3

000
001
010
011
100
101
110
111

Più difficile

Riscrivere il programma di prima, ma stavolta avete in input due numeri n0 e n1. Le stringhe binarie che stampate possono avere al massimo n0 zeri consecutivi e n1 uni consecutivi.

Esempio: n=6, n0=2, n1=3

Il programma deve stampare (non per forza in quest'ordine):

```
100100 010100 110100 001100 101100 011100 010010
110010 001010 101010 011010 111010 100110 010110
110110 001110 101110 001001 101001 011001 111001
100101 010101 110101 001101 101101 011101 010011
110011 001011 101011 011011 111011 100111 010111
110111
```

Matrici

Matrici

Abbiamo visto gli array, cioè una sequenza di elementi tutti dello stesso tipo.

Cosa succede se gli elementi dell'array sono a loro volta array?

Otteniamo una matrice!

Dimensions	Example	Terminology
1	0 1 2	Vector
2	0 1 2 3 4 5 6 7 8	Matrix

Codice

Dichiarazione:

int matrix[RIGHE][COLONNE]

Accedere a un elemento:

int x = matrix[i][j]
Dove 0<=i<RIGHE e 0<=j<COLONNE

Codice

Inizializzazione:

int matrix[3][3] = {{1,2,3}, {4,5,6}, {7,8,9}}
NB: se inizializziamo in questo modo, si può omettere il numero di righe.
Es: int matrix[][3] = {{1,2,3}, {4,5,6}, {7,8,9}}

Inizializzazione da tastiera (o da file con “in”, come sappiamo fare):

```
int matrix[RIGHE][COLONNE];
for(int i=0;i<RIGHE;i++){
	for(int j=0;j<COLONNE;j++){
		cin >> matrix[i][j];
	}
}
```

NB: similmente, con questo doppio ciclo, possiamo stampare tutta la matrice, fare operazioni su tutti gli elementi, eccetera…

Esercizio (semplice)

Data in input una matrice nxm, determinare qual è la riga con la somma maggiore.

Esempio:

4 3
13 34 21
17 22 33
14 55 56
23 24 25

In questo caso dobbiamo dare in output 2 (ricordiamo che gli indici partono da 0!)

Vi sono già dati su github il template con l'input e il file "input.txt"

Potete anche provare a farlo uguale ma per le colonne.

Matrice come mappa

Vedi https://training.olinfo.it/#/task/mappa/statement