Laboratorio di algoritmi e strutture dati

Docente: Violetta Lonati

1 Esercizi introduttivi

1.1 Rovescia

Scrivete un programma che legga una sequenza di numeri interi terminata da 0 e li stampi dall'ultimo (0 escluso) al primo. Potete assumere che la sequenza contenga al più 100 numeri non nulli.

1.2 Cifre ripetute di un numero

Scrivete un programma che legga in input un numero intero n usando `scanf( "%d", &n )` e stabilisca se n contiene cifre ripetute e in caso affermativo quali.

1.3 Da base 10 a base b.

Scrivere un programma che legga una coppia di numeri interi b e n con `scanf( "%d%d", &b, &n )`, quindi memorizzi in un array e stampi la rappresentazione di n in base b. Potete assumere che il numero di cifre in base b sia sempre minore di 100.

Esempio di funzionamento:

```
Inserisci un intero b e un numero in base 10 da convertire in base b: 3 22
Il numero 22 in base 10 equivale al numero 211 in base 3.
```

1.4 Da base b a base 10

Scrivere un programma che dato un numero b (in base 10) e una sequenza s di cifre in $\{0,\dots b-1\}$ terminata da un punto, stampi il numero la cui rappresentazione in base b è data da s. Potete assumere che il numero di cifre di s sia sempre minore di 100.

Esempio di funzionamento:

```
Inserisci un intero b e un numero in base b da convertire in base 10: 3 211.
Il numero 211 in base 3 equivale al numero 22 in base 10.
```

1.5 Esami e studenti

Scrivete un programma che permetta di inserire gli esiti di 5 esami per 5 studenti e calcoli la media di ciascuno studente e la media dei voti ottenuti in ciascun esame.

1.6 Quadrato magico

Scrivete un programma che stampi un quadrato magico di dimensione $n \times n$, con n dispari. Un quadrato magico è una disposizione dei numeri $1, 2, \cdots, n^2$ tale che in ogni riga, in ogni colonna e nelle due diagonali la somma dei numeri sia la stessa.

Memorizzate il quadrato magico in un array bidimensionale. Iniziate mettendo il numero 1 al centro della prima riga. Mettete i rimanenti numeri nell'ordine muovendovi in alto di una riga e a destra di una colonna. Se una cella risulta essere già occupata da un altro numero, allora mettete il numero esattamente sotto al numero che lo precede. Ogni volta che raggiungete un bordo, proseguite ripartendo dal lato opposto del quadrato. Ad esempio, se avete messo un numero nella cella di riga 3 e colonna $n-1$, allora il successivo andrà messo in riga 2 e colonna 0.

Nota: verificate il funzionamento dell'algoritmo e del programma calcolando la somma dei numeri in ogni colonna, riga e diagonale!

Esempio di funzionamento:

```
Inserisci la dimensione del quadrato: 5
 17  24   1   8  15
 23   5   7  14  16
  4   6  13  20  22
 10  12  19  21   3
 11  18  25   2   9
```

1.7 Palindrome

Una stringa si dice *palindroma* se è uguale quando viene letta da destra a sinistra e da sinistra a destra. Quindi "enne" è palindroma, ma "papa" non lo è. Scrivete un programma che legga una stringa terminata da un punto e stabilisca se è palindroma. Potete assumere che la stringa sia al più di 100 caratteri.

1.8 Figure geometriche

Scrivete un programma che calcoli l'area e il perimetro di rettangoli e cerchi. Per rappresentare i punti, i rettangoli ed i cerchi, fate riferimento ad un sistema cartesiano e definite tre tipi di strutture:

- una chiamata `punto`, avente come membri le due coordinate x e y;
- una chiamata `rettangolo`, avente come membri i due vertici opposti del rettangolo;
- una chiamata `cerchio`, avente come membri il centro del cerchio ed il suo raggio.

1.9 Rubrica

Scrivete un programma per la gestione di una semplice rubrica: attraverso un menu l'utente deve poter visualizzare la rubrica e inserire nuovi numeri. Ogni voce della rubrica deve essere composta almeno da un nominativo e da un numero di telefono; documentate con opportuni commenti eventuali assunzioni sulla lunghezza massima delle stringhe e sul numero massimo di voci presenti in rubrica.

2 Esercizi da svolgere in laboratorio

2.1 Istogramma

Scrivete un programma che legga una sequenza di caratteri terminata da un punto e che visualizzi un istogramma con una barra per ogni carattere dell'alfabeto, lunga quanto il numero delle sue occorrenze. Il programma non

deve visualizzare le barre delle lettere che non compaiono e non deve fare distinzione fra maiuscole e minuscole (a tal fine potete usare le funzioni dichiarate nel file `ctype.h`).

Esempio di funzionamento:

```
C'era un RAGAZZO che come ME amava i Beatles e I rolling StoneS.
A *******
B *
C ***
E ********
G **
H *
I ***
L ***
M ***
N ***
O ****
R ***
S ***
T **
U *
V *
Z **
```

2.2 Anagrammi

Due parole costituiscono un *anagramma* se l'una si ottiene dall'altra permutando le lettere (es: attore, teatro). Scrivete un programma che legga due parole e verifichi se sono anagrammi.

Suggerimento: sfruttate il programma scritto per l'esercizio precedente!

2.3 Passeggiate aleatorie

Scrivete un programma che generi una "passeggiata aleatoria" (in inglese *random walk* in un array bidimensionale di dimensione 10×10. L'array sarà riempito di caratteri (inizialmente da punti). Il programma dovrà muoversi di elemento in elemento spostandosi ogni volta di un passo in direzione su, giù, destra o sinistra. Gli elementi visitati andranno etichettati con le lettere dalla A alla Z, nell'ordine in cui vengono visitati. E' importante controllare ad ogni passo che la passeggiata non esca dall'array e che non ritorni su posizioni già visitate. Quando si verifica una di queste condizioni, provate in altre direzioni. Se tutte e quattro le direzioni sono bloccate, il programma deve uscire.

Esempio di funzionamento completo

```
A . . . . . . . . .
B C D . . . . . . .
. . E . . . . . . .
. . F . . . . . . .
I H G . . . . . . .
J . . . . S T . . .
K N O P Q R U . . .
L M . . Z Y V . . .
. . . . . X W . . .
. . . . . . . . . .
```

Esempio di uscita prematura dal programma

```
A . . . . . . . . .
B C D . . . . . . .
K L E . . . . . . .
J M F . . . . . . .
I H G . . . . . . .
. . . . . . . . . .
. . . . . . . . . .
. . . . . . . . . .
. . . . . . . . . .
. . . . . . . . . .
```

Suggerimenti

- Per generare a caso una direzione potete usare le funzioni`time` (da `time.h`), `rand` e `srand` (da `stdlib.h`). La chiamata funzione `rand()` produce un numero apparentemente casuale, ma generato in realtà a partire da un seme. La funzione `srand(n)` inizializza il seme; se il seme non viene inizializzato, il suo valore di default è 1. La chiamata della funzione `time(NULL)` restituisce data e ora corrente, codificate come un unico intero; con la chiamata `srand(time(NULL))` è possibile differenziare i semi e quindi garantire che la passeggiata sia diversa ad ogni esecuzione del programma.

- Dopo aver generato un numero con `rand()`, considerate il suo resto modulo 4. I quattro possibili valori (0,1,2,3) possono essere usati per indicare le direzioni (rappresentatele con una variabile di tipo enum!).

2.4 Ordinamento per inserimento

Scrivete una programma che legga da standard input una sequenza di interi distinti terminati da 0 (potete assumere che la sequenza sia formata al più 100 numeri), memorizzandoli in un vettore ordinato: ogni volta che viene letto un nuovo intero, il vettore viene scorso fino a trovare l'esatta collocazione del numero, quindi si crea lo spazio per il nuovo numero spostando in avanti i numeri successivi già memorizzati.

Adattate il programma precedente e integratelo con il programma per la gestione della rubrica, in modo che l'inserimento delle nuove voci avvenga in ordine alfabetico (rispetto al nominativo). Per confrontare due stringhe rispetto all'ordine alfabetico, potete usare la funzione `strcmp` della libreria `string.h`.

3 Altri esercizi

3.1 Cancella l'ultimo carattere

Scrivete un programma che legga una sequenza di caratteri (terminata da un a-capo) e la ristampi identica ma saltando tutte le occorrenze dell'ultimo carattere. Potete assumere che la sequenza contenga al più 100 caratteri.

Esempio di funzionamento:

La vispa Teresa avea tra l'erbetta a volo sorpresa gentil farfalletta
```
L visp Teres ve tr l'erbett  volo sorpres gentil frfllett
```

3.2 La data più recente

Scrivete un programma che legga una sequenza di al massimo 100 date, nella forma `dd/mm/yyyy`, terminata dalla data `00/00/0000`, che non è considerata parte della sequenza. Alla fine il programma deve stampare la data più recente.

Suggerimento: per stampare un intero con esattamente due cifre (eventualmente preceduto da zeri), dovete usare la specifica di formato `%02d`.

3.3 Il cifrario di Cesare rivisto

Scrivete un programma che legga (usando `getchar`) un testo da cifrare, sotto forma di una sequenza di caratteri terminata da un punto, poi legga (usando `scanf`) la chiave di cifratura k e quindi stampi il testo cifrato usando il cifrario di Cesare con chiave k.

Suggerimento: osservate cosa cambia rispetto alla versione originaria dell'esercizio, svolta durante l'esercitazione precedente.

3.4 Frazioni continue

Se a_0 è un numero intero qualsiasi, e $a_1, a_2, \ldots, a_n$ sono interi positivi, la notazione $[a_0, a_1, \ldots, a_n]$ sta per l'espressione

$$[a_0, a_1, \ldots, a_n] = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\ddots}}}},$$

e viene chiamata *frazione continua*. Ad esempio,

$$[-1, 5, 2, 4] = -1 + \cfrac{1}{5 + \cfrac{1}{2 + \cfrac{1}{2 + \frac{1}{4}}}}.$$

Ovviamente, $[a_0, a_1, \ldots, a_n]$ è un numero razionale, e si può dimostrare che per ogni numero razionale esiste una sola frazione continua che lo rappresenti a meno di eventuali 1 in fondo (poiché, ad esempio, $[-1, 5, 2, 4] = [-1, 5, 2, 3, 1]$). Inoltre, per ogni numero irrazionale esiste un'unica frazione continua infinita che lo rappresenta.

Scrivete un programma che riceva in input un intero n e una sequenza di interi $a_0, a_1, \ldots, a_n$ e stampi in output $[a_0, a_1, \ldots, a_n]$. Potete assumere che n non superi 100.

In particolare, usate questo programma per calcolare $[1, 1, \ldots, 1]$. Che valore ottenete? Riuscite a immaginare che "celebre" valore rappresenta la frazione infinita $[1, 1, 1, \ldots]$?

Suggerimento: Se chiamate x il valore della frazione continua $[1, 1, 1, \ldots]$, vale che $x = 1 + \frac{1}{x}$, quindi...