Laboratorio di Programmazione: Linguaggio C
Lezione 14 del 30 gennaio 2014

Damiano Macedonio

Esercizio 1

Si scriva una *funzione ricorsiva*
`int somma (int a [], int size)`
che riceve un array di interi, calcola la somma dei suoi elementi e restituisce il risultato.

Caso Base: Se l'array è vuoto (`size == 0`) allora la somma dei suoi elementi è zero

Passo Ricorsivo: Se l'array `a[0]`...`a[size - 1]` non è vuoto allora la somma dei suoi elementi è data da `a[size - 1]` più la somma degli elementi in `a[0]`...`a[size - 2]`.

Si scriva quindi una funzione `main` che testa la funzione chiedendo all'utente i valori per riempire un array, passa tale array alla funzione `somma` e stampa il risultato dopo la sua esecuzione.

Esercizio 2

Si scriva un programma C che calcola il quoziente della divisione tra interi `x` e `y` usando una funzione ricorsiva.

Caso base: se `x < y` allora il quoziente è zero,

Passo ricorsivo: altrimenti quoziente è `1` più il quoziente della divisione di `(x - y)` per `y`.

Il programma dovrà contenere anche una funzione `int leggi(void)` che richiede all'utente un intero non negativo (controllare l'input). Tale funzione sarà chiamata due volte all'interno della funzione `main` per inizializzare `x` e `y`.

Esercizio 3 [Tratto dalla prova parziale del 24 gennaio 2013]

Scrivere un programma C che definisce le funzioni:

`int leggi(void)` che legge da tastiera un numero intero non negativo e lo restituisce. Se il valore inserito fosse negativo, deve continuare a chiederlo all'utente;

`void stampa(int numero)` che stampa le cifre del numero intero indicato, in italiano. Per esempio, se `numero` è 4301 allora deve stampare `quattro tre zero uno`; se `numero` è 0 allora deve stampare `zero`. È possibile definire ulteriori funzioni ausiliarie, se servono. La funzione `stampa()` deve essere ricorsiva o chiamare una vostra funzione ricorsiva.

- Caso base: `numero == 0`
- Chiamata ricorsiva: `numero / 10`

`int main(void)` che chiama `leggi` per leggere un numero non negativo e poi chiama `stampa` per stamparne le cifre in italiano.

Suggerimento: Si veda la versione non ricorsiva della Lezione 12. La versione ricorsiva non richiede di utilizzare un array.

Esempi

```
$ ./a.out
inserisci un numero: 10985
uno zero nove otto cinque

$ ./a.out
inserisci un numero positivo: -13
inserisci un numero positivo: 8901
otto nove zero uno

$ ./a.out
inserisci un numero: 300896
tre zero zero otto nove sei

$ ./a.out
inserisci un numero: 0
zero

$ ./a.out
inserisci un numero: 0006
sei
```

Esercizio 4 [Tratto dalla prova parziale del 4 febbraio 2013]

Si scriva un programma C che definisce la *funzione ricorsiva*

```
int cifra_massima(int num)
```

la quale deve restituire la cifra massima nella rappresentazione decimale di `num`. Tale programma dovrà inoltre definire un `main` che (1) chiede all'utente di inserire un numero non negativo, (2) chiama la funzione `cifra_massima` per calcolarne la cifra massima e (3) stampa tale cifra massima trovata.

Esempi

```
$ ./a.out
Inserisci un numero non negativo: 1232
La cifra massima di 1232 e' 3

$ ./a.out
Inserisci un numero non negativo: 0
La cifra massima di 0 e' 0

$ ./a.out
Inserisci un numero non negativo: -5
Inserisci un numero non negativo: 30756
La cifra massima di 30756 e' 7
```

Esercizio 5 [Tratto dalla prova parziale del 24 gennaio 2013]

Scrivere un programma C che definisce le funzioni

`int next_prime(void)` che restituisce un diverso numero primo ad ogni chiamata, dal 2 in poi. I numeri primi prodotti devono essere in sequenza crescente. La funzione non ha parametri e deve essere ricorsiva. (*Suggerimento:* potete usare una variabile statica di appoggio per ricordare l'ultimo valore considerato.)

`int continua(void)` che chiede all'utente se vuole continuare. Ritorna `0` (falso) se l'utente si vuole fermare e ritorna `1` se l'utente vuole continuare.

`int main(void)` che richiama le funzioni `continua` e `next_prime` e stampa il risultato.

Esempio

```
$ ./a.out
Vuoi continuare (s/n): s
Numero primo: 2
Vuoi continuare (s/n): s
Numero primo: 3
Vuoi continuare (s/n): s
Numero primo: 4
Vuoi continuare (s/n): s
Numero primo: 5
Vuoi continuare (s/n): s
Numero primo: 7
Vuoi continuare (s/n): s
Numero primo: 9
Vuoi continuare (s/n): s
Numero primo: 11
Vuoi continuare (s/n): n
$
```

Esercizio 6

Si scriva una funzione `int inverti (int n)` che riceve un intero non negativo `n` e restituisce l'intero ottenuto invertendo l'ordine delle cifre di `n`. La funzione `inverti` deve essere ricorsiva o richiamare una funzione ricorsiva.

Si scriva inoltre una funzione `int leggi(void)` che richiede all'utente un intero non negativo (controllare l'input) e lo ritorna.

Tali funzioni sono chiamate all'interno della seguente funzione `main` per chiedere all'utente due interi e stampa a video il loro valore inertito.

```
int main(void) {
  int m = leggi();
  int n = leggi();
  printf("Il valore di %d invertito e' %d\n", m, inverti(m));
  printf("Il valore di %d invertito e' %d\n", n, inverti(n));
  return 0;
}
```

Esempi di esecuzione

```
$ ./a.out
inserisci un numero positivo: 1232
inserisci un numero positivo: 0
Il valore di 1232 invertito e' 2321
Il valore di 0 invertito e' 0

$ ./a.out
inserisci un numero positivo: -5
inserisci un numero positivo: 6723
inserisci un numero positivo: 30700
Il valore di 6723 invertito e' 3276
Il valore di 30700 invertito e' 703
```