Prova scritta del 20/6/2011

Strutturare adeguatamente i programmi ed evidenziarne la strutturazione mediante indentazione. Inserire anche adeguati commenti

1) Realizzare una funzione booleana di nome `sottostringa` che, presa una stringa `S` e due caratteri `i` e `f`, determina e restituisce la prima sottostringa `SS` di `S` delimitata a sinistra e a destra rispettivamente da `i` e `f`. Note. I caratteri delimitatori non fanno parte di `SS`. Nel caso in cui non si trovi in `S` uno dei due delimitatori o `f` non sia successivo a `i`, la funzione restituisce `false`, altrimenti restituisce `true`. Il carattere `'#'` in `i` o in `f` indica, rispettivamente, che `SS` inizia dal primo carattere di `S` e che `SS` termina con l'ultimo carattere di `S`.

Esempio – *S = "mario.verdi@studenti.unipr.it"*
se i = '@', f = '.' allora SS = "studenti"
se i = '#', f = '@' allora SS = "mario.verdi"
se i = '@', f = '#' allora SS = "studenti.unipr.it"
se i = 's', f = '@' allora false

N.B. `S` e `SS` devono essere stringhe "tipo C". Dichiarare `S`, `SS`, `i`, ed `f` come parametri della funzione.

2) Sia `"indirizzi.txt"` un file contenente indirizzi di posta elettronica della forma *nome.cognome@dom1.….domn* (si assuma che gli indirizzi siano tutti sintatticamente corretti e di lunghezza massima *64*, e siano separati l'uno dall'altro da esattamente un carattere `'\n'`). Scrivere un programma principale che cerca e stampa i nominativi (cognome e nome) relativi a tutti gli indirizzi memorizzati nel file in cui appaia la stringa `"studenti"` come primo dominio (= sottostringa a destra di `'@'`). Si richiede di utilizzare, obbligatoriamente, la funzione `sottostringa` realizzata nell'esercizio (1), sia per individuare gli indirizzi nel file che per individuare nome e cognome all'interno di ciascun indirizzo.

Esempio – *File indirizzi.txt:* *mario.verdi@studenti.unipr.it*
anna.bianchi@unipr.it
elena.rossi@studenti.unibo.it

Output:
verdi mario
rossi elena

3) Sia `dati` un tipo `struct` costituito da un campo `c1` di tipo *array* (di dimensione max. 100), e da due campi `c2` e `c3`, risp. di tipo *intero* e di tipo *carattere*. Realizzare una funzione di nome `disgiungi` che, presi come suoi parametri un array di interi `I`, il numero `n` di elementi in `I` e due strutture `I1` ed `I2` di tipo `dati`, suddivide gli elementi di `I` in `I1` ed `I2` nel modo seguente: se l'elemento di `I` e' pari viene copiato nell'array `c1` di `I1`, altrimenti, nell'array `c1` di `I2` (a partire dagli elementi di indice 0). Il campo `c2` contiene il numero di elementi presenti nel corrispondente array `c1`, mentre il campo `c3` contiene un carattere che specifica se il corrispondente array `c1` contiene numeri pari (`'p'`) o dispari (`'d'`).

Scrivere anche un programma principale che legge da std. input una sequenza di `n` interi (`n` dato di input fornito dall'utente), li memorizza in un array `prova`, richiama la funzione `disgiungi` passandole come parametri l'array `prova` e due strutture `A` e `B`, e quindi stampa `A` e `B`.

Esempio – *prova = {4,6,13,2,5} (n = 5)*
A = 4 6 2 B = 13 5
3 2
'p' 'd'

4) [SOLO CdL in FISICA] Date le seguenti istruzioni C++

```
struct elem {int info1; int info2;};
elem* p[10];
for(int i=0; i<10; i++) {
  p[i] = new elem;
  p[i]->info1 = i;
  p[i]->info2 = p[i]->info1 * i;
}
```

qual è il risultato della loro esecuzione? Illustrare mediante un disegno le strutture dati create. Giustificare dettagliatamente la risposta.