Fondamenti di Programmazione - CdL in MATEMATICA

I Prova di verifica del 3/5/2011

num. eserc.	1	2	3	4	5
punt. tot	8	8	8	3	3

N.B.: Negli esercizi di programmazione, viene valutata anche la leggibilità del codice proposto. Inoltre, non è consentito l'uso di istruzioni che alterino il normale flusso dell'esecuzione all'interno di cicli che ne provochino l'uscita forzata.

ESERCIZIO 1 (8 punti)

- Definire sull'alfabeto $\Sigma = \{a, b, ..., z\}$ un automa a stati finiti non deterministici (NFA) che riconosca tutte le stringhe che contengono la **sottostringa** *war*. Ad esempio *aware* e *warrior* appartengono al linguaggio, mentre *wear* no.
- Dimostrare che l'NFA definito effettivamente riconosca il linguaggio
- Utilizzando la tecnica di costruzione per sottoinsiemi, produrre l'equivalente automa a stati finiti deterministici (DFA).

ESERCIZIO 2 (8 punti)
Definire una funzione `boolean check (char *v1, char *v2, int dim1, int dim2)` che dati due array di caratteri, verifica se per ogni elemento x del primo vettore (di dimensione `dim1`) esiste il carattere successivo a x nel secondo vettore (di dimensione `dim2`).

ESERCIZIO 3 (8 punti)
Si definisca una funzione che, ricevendo in ingresso una sequenza di interi di lunghezza arbitraria, terminata da uno zero, produca in uscita la somma del primo elemento pari e dell'ultimo elemento dispari, escluso lo zero. Ad esempio se la sequenza fosse " 10 5 13 3 9 7 8" la funzione dovrebbe restituire 10+7 = 17.

ESERCIZIO 4 (3 punti)
Definire il prototipo della funzione `foo`, in modo che la sua chiamata all'interno del seguente frammento di codice risulti corretta a tempo di compilazione.

```
main()
{
int *p, x;
char c = 'b' - 1;
p = &x;
c = foo(*p,&c, &p);
...
}
```

ESERCIZIO 5 (3 punti)
Data la seguente definizione

```
void p(int x, int y)
{
int a, b, *h, *k;
a = (x + y);
b = (x * y);
h = &a;
k = &b;
if ((x % y) != 0)
{   h = k;
    k = &a; }
else
{  *k = (*k) + (*h);
   *h = (*k) + (*h);   }
printf("%d %d %d %d\n",a,b,*h,*k);
}
```

dire quali valori vengono stampati in corrispondenza delle due chiamate:

i) `p(5, 5)`

ii) `p(8, 3)`