PROVA SCRITTA DI FONDAMENTI DI INFORMATICA A
30 MARZO 2001

Esercizio 1 (punti 9)
Si scriva una funzione ricorsiva **`double f(double a, double n);`** che calcoli il seguente valore

$$\sum_{i=0}^{n} i * a^i$$

L'elevamento a potenza x^y viene reso disponibile dalla libreria **`math.h`** e ha il seguente formato

```
double pow(double x, double y)
```

Esercizio 2 (punti 8)
Dato il seguente programma C:

```
#include<stdio.h>

void mul(int V1[], int V2[]);
void stampa(int vet[],int k);

main()
{int F[4]={3,6,9,12};
int D[4]={-3,-6,-9,-12};

int cont = 0;
 stampa(F,cont);
 mul(F,D);
 stampa(F,cont);
 printf("valore di cont: %d",cont);
 }

void mul(int V1[], int V2[])
{int i;
 for(i=0;i<4;i=i+2)
      V1[i]= V1[i]*V2[i];
 }

void stampa(int vet[], int k)
{int i;
 printf("Vettore:\n");
 for(i=0;i< 4 ;i++)
  if (vet[i] < 0)
      {printf("%d ",vet[i]);
      k++;}
 printf("\n%d\n",k);}
```

Cosa viene stampato dal programma? La risposta deve essere opportunamente motivata.

Esercizio 3 (punti 4)
Si scriva una procedura che legga da input una serie di numeri positivi (massimo 10) e termini con un numero negativo o nullo. Tale procedura restituisce in uscita (come parametri passato per riferimento) un vettore **V** (di dimensione fisica 10 dichiarato nel programma chiamante) contenente i numeri letti e il numero effettivo di elementi letti **N** (ossia la dimensione logica del vettore).

```
void leggi(int V[], int *N);
```

Si dia anche un esempio di chiamata.

Esercizio 4 (punti 7)
Data la seguente funzione ricorsiva:

```
double f(double a, double n)
{ if (n==1) return a;
  else return pow(a,n) + f(a,n-1);
}
```

Si dica se la funzione è tail ricorsiva motivando la risposta.
Si dica qual è il valore restituito dalla funzione e si disegnino i record di attivazione nel caso in cui la funzione sia chiamata con i seguenti parametri attuali **f(2,3)**.

Esercizio 5 (punti 2)

Supponiamo di dover assegnare due strutture identiche contenenti un intero e un float:

```
struct prova{ int A;
              float B;
              };
struct prova S1;
struct prova S2;
```

.Quali delle seguenti istruzioni non è consentita per effettuare l'assegnamento ?

A. `S1 = S2;`
B. `S1 == S2;`
C. `S1.A = S2.A; S1.B = S2.B;`

SOLUZIONE

Esercizio 1

```
int f(int a, int n)
{ if (n==0) return 0;
  else return (n + 1) * pow(a,n) + f(a, n-1);
}
```

Esercizio 2
Vettore:

0
Vettore:
-9 -81
2
valore di cont: 0

Esercizio 3

```
void leggi(int V[], int *N)
{
    int i = 0;
    int Num;
    printf("inserisci un intero positivo, 0 o neg per terminare");
    scanf("%d", &Num);
    while (Num > 0)
    { V[i] = Num;
      i++;
      printf("inserisci un intero positivo, 0 o neg per terminare");
      scanf("%d", &Num);
    }
 *M = i;
}
```

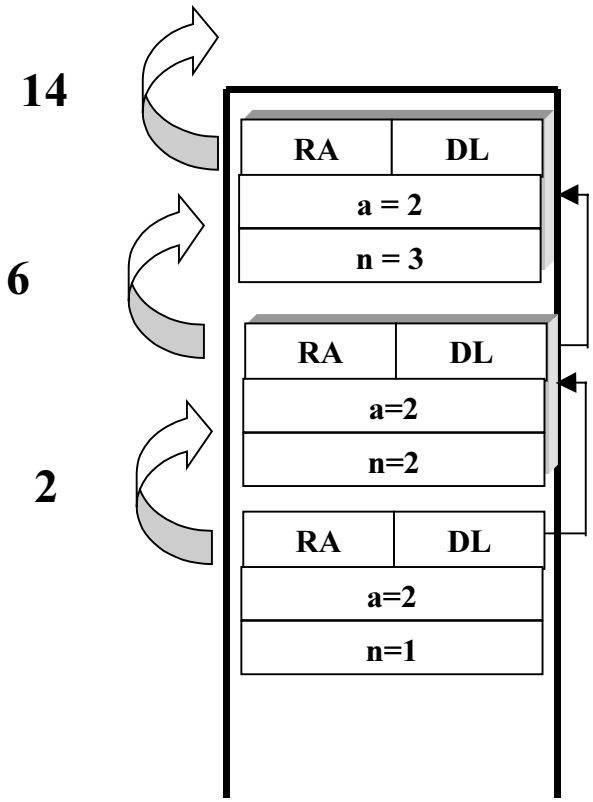
```
chiamata:

main(){
int VETTORE[10];
int Dimlogica;

leggi(VETTORE, &DimLogica);
}
```

Esercizio 4

La funzione non è tail ricorsiva, perché dopo la chiamata ricorsiva deve ancora essere calcolata la somma.
Sequenza chiamate
`f(2,3)  -> f(2,2) -> f(2,1)`



Esercizio 5 (punti 2)

B. Non e' un operatore di assegnamento ma di confronto.