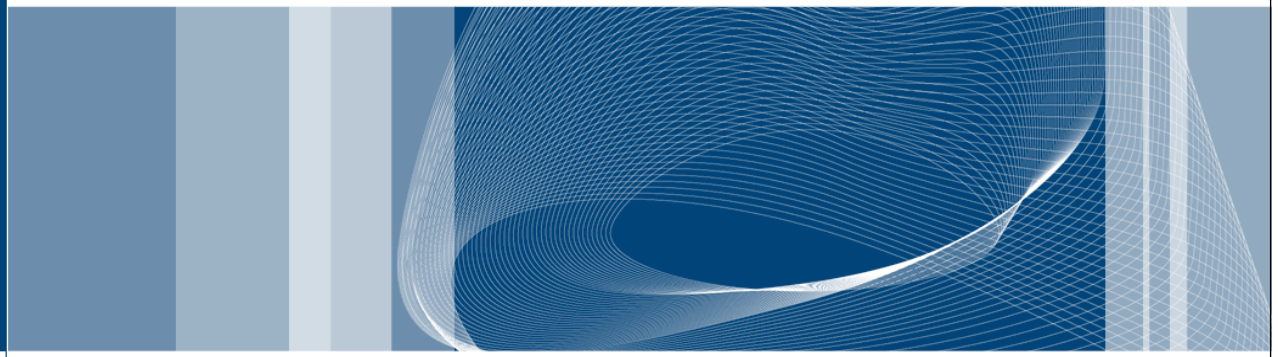
POLITECNICO DI MILANO



Fondamenti di Informatica 2013-2014

Funzioni

Paola Mussida
Area Servizi ICT

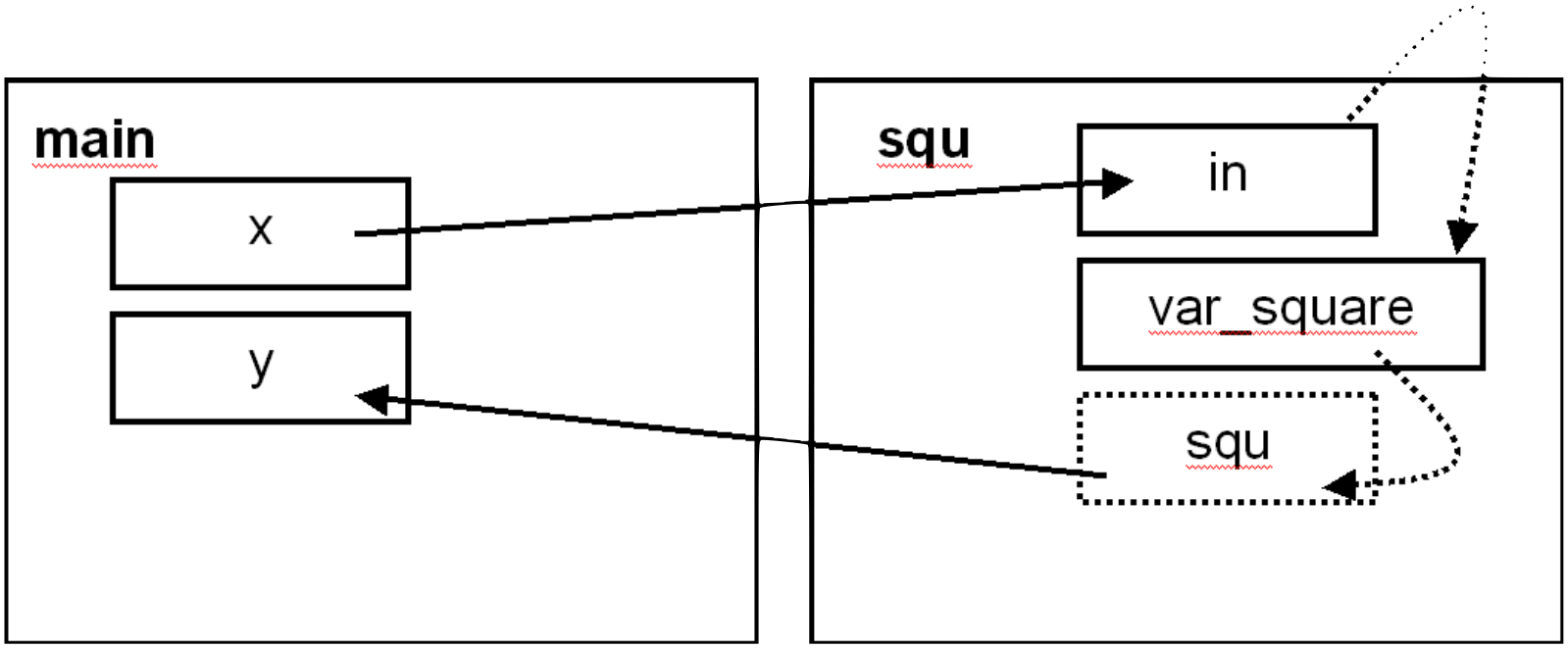
Scrivere una funzione per il calcolo del quadrato di un intero.
La funzione riceve in ingresso un intero e restituisce l'intero corrispondente al quadrato.

Esercizio F_1 - Quadrato - funzione


```
...   /*funzione che calcola il quadrato del valore di “in”*/
int squ(int in)
{
   int var_square;
   var_square = in * in;
   return var_square; /* Assegna a squ() il valore di
                                                  var_square */
}

main() /* Programma principale */
{
    int x,y;
    for(x = 0;x <= 7;x++)   {
       y = squ(x);  /* assegna ad y il valore restituito
                       dalla funzione */
       printf("Il quadrato di %d e’ %d\n", x, y);
    }
    for (x = 0;x <= 7;++x)
    printf("Il quadrato di %d e’ %d\n", x,  squ(x) );
}
```

Esercizio F_1 - Quadrato - esecuzione



Scrivere una funzione per calcolare la somma di due interi passati per valore.

Esercizio F_2 - Somma di due interi

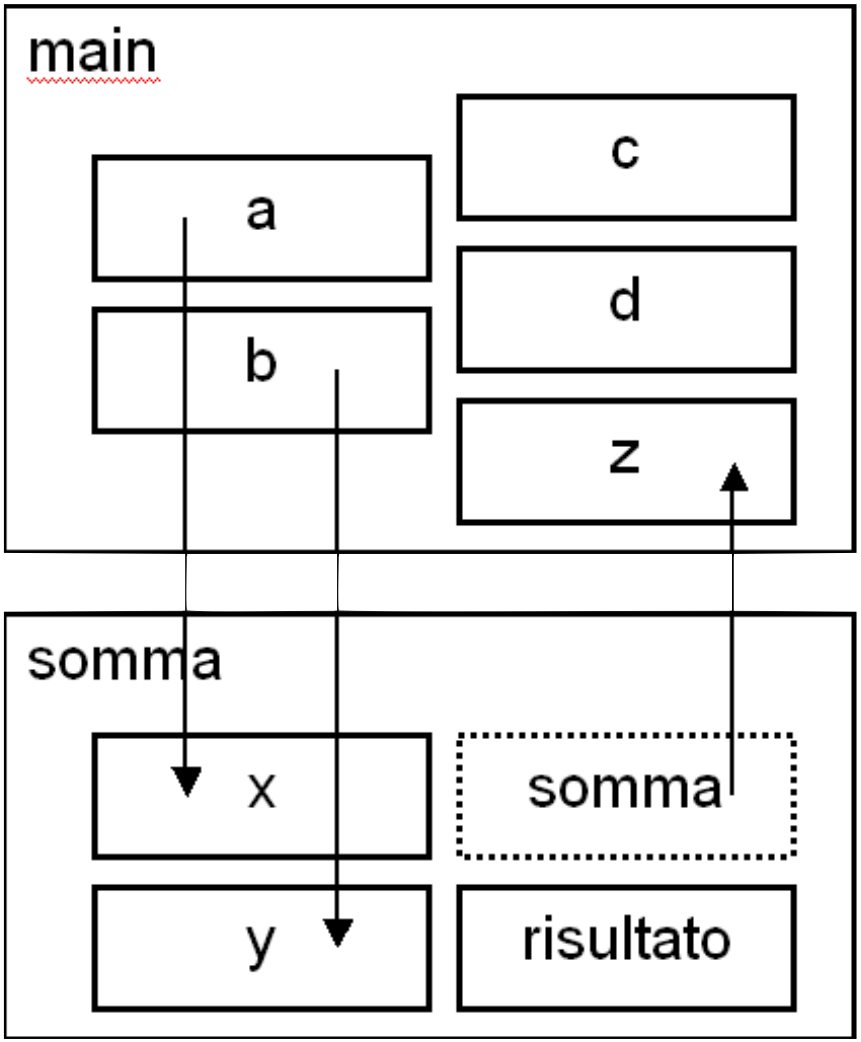

```c
int somma(int,int);          /* prototipo funzione somma  */

main() {
        int a=5, b=7, c=2, d=9, z;

        z=somma(a,b);
        printf("\nz = %d ",z);
        z=somma(a,b) * somma(c,d);
        printf("\nz = %d ",z);
        z=somma(a,somma(b,c));
        printf("\nz = %d ",z);
}

/* definizione della funzione somma */
int somma(int x, int y) {
    int risultato;
    risultato=x+y;
    return (risultato);
}
```

Esercizio F_2 - Somma di due interi



Scrivere una funzione che stampa un numero di asterischi passato per valore.

Esercizio F_3 - Stampa *

```
Output:

1   ********** 13
2   *** 3
3   ********* 9
4   ************************ 23
5   ********** 15

                . . .

20  ********* 9
```

Definire una costante per il numero di volte in cui la funzione viene chiamata. Impostiamo a 20 il valore

Esercizio F_3 - Stampa *

```c
#define MAX 20

void riga(int n); /*stampa una riga di <n> asterischi*/

void main() {
   int n, i;

   /* visualizza MAX righe */
   for (i=0 ; i < MAX ; i++)        {
      n=rand() % 50;   /* generazione numero casuale */

      riga(n);   /*visualizzazione riga di asterischi*/

      printf(" %d", n); /* stampo n che non è variato*/
   }
}
```

Esercizio F_3 - Stampa *

```
/* stampa una riga di <n> asterischi */

void riga (int n) {

        printf ("\n");
        while (n>0)
        {
                printf("*");
                n--;
        }

}
```

Scrivere due funzioni per verificare se un numero è primo, utilizzando i prototipi forniti; la prima funzione chiama la seconda.

Esercizio F_4 - Numero primo


```
typedef enum {falso,vero} SiNo;
                              /* prototipi delle funzioni */
SiNo primo (int);

SiNo divisibile (int,int);

void main() {
   int numero=1;

   while(numero!=0) { /* fino all'inserimento del numero 0 ... */
      printf("\nInserisci numero:");
      scanf("%d",&numero);
      if (primo(numero)==vero) /*verifica se il numero è primo*/
         printf("\n Il numero è primo");
      else
         printf("\n Il numero non è primo");
   }
}
```

Esercizio F_4 - Numero primo

```
/* la funzione restituisce vero se il numero
                        passato come parametro è primo */
SiNo primo(int num) {

  int i;
  /* verifica se num è divisibile per un numero
                                  compreso tra 2 e num */
       /*Sarebbe sufficiente arrivare alla radice di n*/
  for(i=2; i<num; i++) {

     if ( divisibile(num,i) == vero )
        return(falso);
  }
  /* non è stato trovato nessun numero che divide num */
  return(vero);
}
```

Esercizio F_4 - Numero primo

```
/* ritorna vero se il primo parametro è divisibile per
                                                il secondo */
SiNo divisibile (int num, int divisore) {


    int resto;

    resto = num % divisore;

    if (resto!=0)                /* equivale a if (resto) */
       return(falso);
    else
       return(vero);



}
```

Scrivere una funzione che calcoli gli interessi maturati su un capitale in N anni.

Esercizio F_5 - Interessi


```
typedef enum {FALSO, VERO} boolean;
float pot(float x, int exp) {          /* calcolo di x elevato a exp */
 float potenza=1.0;
 int positivo=VERO;

 if (exp<0) {                    /* verifica se l'esponente e' negativo */
    positivo=FALSO;
    exp=abs(exp);                          /* cambio segno all'esponente */
 }
 while (exp>0) {          /*calcolo della potenza con esp sempre positivo*/
  /*nel caso di esp dispari, moltiplico il risultato parziale per x*/
    if ( (exp % 2) == 1 )
        potenza = potenza * x;
    x = x * x;
    exp = exp / 2;  /*elevo il numero x al quadrato e dimezzo l'esp*/
 }
 if ( positivo==VERO ) /* valore restituito */
    return (potenza);
 else
    return (1.0/potenza);
}
```

Esercizio F_5 - Interessi


```
void main() {
        float cap, interes, montante, maturato;
         int anni;

         /* inserimento dati */
         printf("\nCapitale: ");
         scanf("%f", &cap);
         printf("\nInteresse: ");
         scanf("%f", &interes);
         printf("\nAnni: ");
         scanf("%d", &anni);

         /* calcolo interessi */
         montante = cap * pot( (1+interes) , anni);
         maturato = montante - cap;

         /* visualizzazione interessi maturati */
         printf("\nMontante: %f", montante);
         printf("\nMaturato: %f", maturato);
 }
```

Scrivere una funzione che riceve tre parametri di tipo puntatore a intero e restituisce il puntatore del parametro con valore maggiore.

Esercizio F_6 - Maggiore di tre


```
int a, b, c;
int *magg;
/* restituisce l'indirizzo del parametro maggiore */
int *maggiore(int *x, int *y, int *z) {

            if ( *x >= *y )
                if ( *x >= *z )
                        return x;
                else
                        return z;
            else
                if ( *y >= *z )
                        return y;
                else
                        return z;
}

/* Esempio di chiamata alla funzione maggiore():
             magg = maggiore(&a, &b, &c);
             printf("\nIl numero maggiore e': %d", *magg);*/
```

Scrivere una funzione che scambia i contenuti dei tre parametri ricevuti in modo che il primo contenga il valore maggiore.
La funzione restituisce inoltre la somma dei tre valori.

Passaggio per valore e modifica del parametro formale

Esercizio F_7 - Ordina 3 parametri

```
int a, b, c, *p;
char risp;
/* ordina i tre parametri e restituisce la loro somma*/
int ordina(int *x, int *y, int *z) {
        int tmp;
         if ( *x < *y ) {
                  tmp = *x;
                  *x = *y;
                  *y = tmp;
         }
         if ( *x < *z ) {
                  tmp = *x;
                  *x = *z;
                  *z = tmp;
         }
                           /* ... */
```

Esercizio F_7 - Ordina 3 parametri

```
                /* ... */

        if ( *y < *z ) {
                tmp = *y;
                *y = *z;
                *z = tmp;
         }
         return (*x + *y + *z);
 }


/* Esempio di chiamata alla procedura ordina():
       printf("\nLa somma e': %d", ordina( &a, &b, &c));
*/
```

Esercizio F_7 - Ordina 3 parametri - Altra versione


```c
int a, b, c, *p;
char risp;
/* cambia di posto due valori se il secondo è maggiore del primo*/
int scambiaSeMag (int *var1, int *var2) {
  int tmp;
  if ( *var1 < *var2 ) {
        tmp = *var1;
        *var1 = *var2;
        *var2 = tmp;
  }
}

/* ordina i tre parametri e restituisce la loro somma */
int ordina(int *x, int *y, int *z) {
  scambiaSeMag (x, y);
  scambiaSeMag (x, z);
  scambiaSeMag (y, z);

  return (*x + *y + *z);
}

/* Esempio di chiamata:
   printf("\nLa somma e': %d",
          ordina( &a, &b, &c));  */
```

Simulazione di chiamate a funzioni.

Esercizio F_8 - Esempi


```
int varA = 100;

int proc1(int par) {
    varA=varA+1;
    par = par - 10;
    return (par);
}

int proc2(int *par) {
    *par = *par - 10;
    return (*par);
}

int proc3(int par) {
    int varA;
    varA = par + 1;
    return (varA);
}
```

Esercizio F_8 - Esempi


```
void main() {
   int varB=0, varC=50;

   /* situazione iniziale */
   printf ("0) varA = %d, varB = %d, varC = %d ", varA, varB, varC);

   varB = proc1(varC); /* passo 1 */
   printf ("1) varA = %d, varB = %d, varC = %d ", varA, varB, varC);

   varB = proc2(&varC); /* passo 2 */
   printf ("2) varA = %d, varB = %d, varC = %d ", varA, varB, varC);

   varB = proc3(varC); /* passo 3 */
   printf ("3) varA = %d, varB = %d, varC = %d ", varA, varB, varC);

   varC = proc1(varA); /* passo 4 */
   printf ("4) varA = %d, varB = %d, varC = %d ", varA, varB, varC);
}
```

Esercizio F_8 - Esempi - Risultato

```
0) varA = 100,  varB =  0,  varC = 50

1) varA = 101,  varB = 40,  varC = 50

2) varA = 101,  varB = 40,  varC = 40

3) varA = 101,  varB = 41,  varC = 40

4) varA = 102,  varB = 41,  varC = 91
```

Simulazione di chiamate a funzioni.

Esercizio F_9 - Esempi


```c
#include <stdio.h>
int varA = 100, varB, varC = 56;

int proc1(int par) {
        par = varA + par;
        return (par);
}

int proc2(int *par) {
        varB = *par;
        *par = *par - varC;
        return (*par);
}

int proc3(int *par) {
        int varA = 5;
        *par = varB;
        return (varA + varC);
}
```

Esercizio F_9 - Esempi

```
void main() {

   int varB=0, varC=50;

   /* situazione iniziale */
   printf ("0) varA = %d, varB = %d, varC = %d \n", varA, varB, varC);

   varB = proc1(varC); /* passo 1 */
   printf ("1) varA = %d, varB = %d, varC = %d \n", varA, varB, varC);

   varB = proc2(&varC); /* passo 2 */
   printf ("2) varA = %d, varB = %d, varC = %d \n", varA, varB, varC);

   varA = proc3(&varC); /* passo 3 */
   printf ("3) varA = %d, varB = %d, varC = %d \n", varA, varB, varC);

   varC = proc1(varA); /* passo 4 */
   printf ("4) varA = %d, varB = %d, varC = %d \n", varA, varB, varC);
}
```

Esercizio F_9 - Esempi _ Risultato

```
0) varA = 100,  varB =   0,  varC =  50
1) varA = 100,  varB = 150,  varC =  50
2) varA = 100,  varB =  -6,  varC =  -6
3) varA =  61,  varB =  -6,  varC =  50
4) varA =  61,  varB =  -6,  varC = 122
```