```
/*
 * Corso di Fondamenti di Informatica
 * Esercizio:
 * elaborazione di polinomi rappresentati tramite array
 * (Soluzione ESERCIZIO A)
 */

/*************************************************************************
 * Versione del programma rielaborata in modo da corrispondere alle
 * indicazioni dell'ESERCIZIO A in coda alla versione originale del programma.
*************************************************************************/

#include <stdio.h>

#define GRADO_MAX_AB 10
/* massimo grado ammesso per i polinomi A e B */
/* Nota: per semplificare al massimo il codice per il calcolo della somma
 *       (si veda l'ESERCIZIO A in coda al programma originale) e' opportuno
 *       che gli array contenenti i due polinomi abbiano lo stesso numero di
 *       elementi */

#define GRADO_MAX_RIS (GRADO_MAX_AB+GRADO_MAX_AB)
/* massimo grado che il polinomio risultato può assumere, dati i gradi dei due
 * polinomi addendi o fattori. Notare che il grado così definito è quello
 * del polinomio prodotto, ma è anche sicuramente maggiore o uguale al grado
 * del polinomio somma. In definitiva, un array in grado di rappresentare un
 * polinomio di grado GRADO_MAX_RIS è adatto per contenere sia il prodotto che
 * la somma dei polinomi A e B */

int main()
{
  float PoliA[GRADO_MAX_AB+1];
  /* polinomio A: l'elemento dell'array di indice i rappresenta il
   * coefficiente del termine di grado i. Tutti gli elementi dell'array
   * vengono utilizzati, indipendentemente dal grado effettivo del polinomio:
   * quelli corrispondenti ai coefficienti dei termini assenti nel polinomio
   * vengono semplicemente uguagliati a 0.0. */
  float PoliB[GRADO_MAX_AB+1];
  /* polinomio B: l'elemento dell'array di indice i rappresenta il
   * coefficiente del termine di grado i. Tutti gli elementi dell'array
   * vengono utilizzati, indipendentemente dal grado effettivo del polinomio:
```

```
 * quelli corrispondenti ai coefficienti dei termini assenti nel polinomio
 * vengono semplicemente uguagliati a 0.0. */
float PoliRis[GRADO_MAX_RIS+1];
/* polinomio risultato della somma o del prodotto: l'elemento dell'array di
 * indice i rappresenta il coefficiente del termine di grado i. Tutti gli
 * elementi dell'array vengono utilizzati, indipendentemente dal grado
 * effettivo del polinomio: quelli corrispondenti ai coefficienti dei
 * termini assenti nel polinomio vengono semplicemente uguagliati a 0.0. */
int GradoA, GradoB, GradoRis;
/* gradi effettivi dei polinomi A e B (inseriti dall'utente) e grado
 * effettivo del polinomio risultante dal prodotto tra A e B */
int GradoCorrA, GradoCorrB, GradoCorrRis;
/* contatori usati per scorrere gli array (il terzo non è necessario, ma
 * introdurlo chiarifica il programma) */

/* ------------inizializzazione dei polinomi A e B:------------ */

printf("\nInserisci il grado del polinomio A:\n");
scanf("%d", &GradoA);
printf("\nInserisci il grado del polinomio B:\n");
scanf("%d", &GradoB);
GradoRis = GradoA + GradoB;
/* per le regole del prodotto tra polinomi */

printf("\nInserisci i coefficienti del polinomio A:\n");
GradoCorrA = GRADO_MAX_AB;
while (GradoCorrA >= 0)
{
  if (GradoCorrA > GradoA)
  {
    PoliA[GradoCorrA] = 0.0;
  }
  else
  {
    printf("\nA%d (coeff. di x^%d): ", GradoCorrA, GradoCorrA);
    scanf("%f", &PoliA[GradoCorrA]);
  }
  --GradoCorrA;
}

printf("\nInserisci i coefficienti del polinomio B:\n");
```

```
GradoCorrB = GRADO_MAX_AB;
while (GradoCorrB >= 0)
{
  if (GradoCorrB > GradoB)
  {
    PoliB[GradoCorrB] = 0.0;
  }
  else
  {
    printf("\nB%d (coeff. di x^%d): ", GradoCorrB, GradoCorrB);
    scanf("%f", &PoliB[GradoCorrB]);
  }
  --GradoCorrB;
}

/* ------------stampa dei coefficienti dei polinomi A e B:------------ */
printf("\n\nA:\t");
GradoCorrA = GradoRis;
while (GradoCorrA >=0)
{
  if (GradoA >= GradoCorrA )
      /* se il polinomio possiede un termine di grado GradoCorrA */
  {
        printf("%+.3f x^%d\t", PoliA[GradoCorrA], GradoCorrA);
    /* Nota: lo specificatore ".3" prima di "%f" serve a limitare a 3 il
     * numero di cifre dopo la virgola che vengono stampate; il segno + e'
     * un indicatore che richiede a printf di indicare sempre il segno del
     * numero stampato, anche quando e' positivo. */
      }
  else
      {
    printf("\t\t");
      }
      --GradoCorrA;
}

printf("\n\nB:\t");
GradoCorrB = GradoRis;
while (GradoCorrB >=0)
{
  if (GradoB >= GradoCorrB )
```

```
    /* se il polinomio possiede un termine di grado GradoCorrB */
  {
    printf("%+.3f x^%d\t", PoliB[GradoCorrB], GradoCorrB);
      }
  else
      {
        printf("\t\t");
      }
      --GradoCorrB;
}

/* ------------calcolo e stampa della somma:------------ */
GradoCorrRis=0;
while (GradoCorrRis <= GradoRis)
{
  if ( GradoCorrRis > GRADO_MAX_AB )
      /* se i due array PoliA e PoliB non sono in grado di rappresentare monomi
       * di grado GradoCorrRis (a causa della loro dimensione limitata) */
      {
        PoliRis[GradoCorrRis] = 0.0;
      }
  else
  /* i polinomi A e B contengono (eventualmente con coefficiente nullo) un
   * monomio di grado GradoCorrRis */
      {
    PoliRis[GradoCorrRis] = PoliA[GradoCorrRis] + PoliB[GradoCorrRis];
  }

        ++GradoCorrRis;
}

printf("\n\nA+B:\n");
GradoCorrRis = GradoRis;
while (GradoCorrRis >=0)
{
  if ( (GradoB >= GradoCorrRis) || (GradoA >= GradoCorrRis) )
      /* se il polinomio possiede un termine di grado GradoCorrB */
      {
        printf("%+.3f x^%d\t", PoliRis[GradoCorrRis], GradoCorrRis);
      }
  else
```

```
        {
      printf("\t\t");
        }
        --GradoCorrRis;
    }

    /* ------------calcolo e stampa del prodotto:------------ */
    GradoCorrRis = 0;
    while (GradoCorrRis <= GradoRis)
    {
      /* azzeramento di tutti i coefficienti del risultato */
      PoliRis[GradoCorrRis] = 0.0;
      ++GradoCorrRis;
    }

    GradoCorrA = 0;
    while (GradoCorrA <= GradoA)
    {
      GradoCorrB = 0;
      while (GradoCorrB <= GradoB)
        {
          PoliRis[GradoCorrA+GradoCorrB] = PoliRis[GradoCorrA+GradoCorrB] +
                                           PoliA[GradoCorrA] * PoliB[GradoCorrB];

       ++GradoCorrB;
        }
        ++GradoCorrA;
    }

    printf("\n\nA*B:\n");
    GradoCorrRis = GradoRis;
    while (GradoCorrRis >=0)
    {
      printf("%+.3f x^%d\t", PoliRis[GradoCorrRis], GradoCorrRis);
      --GradoCorrRis;
    }

    printf("\n\n");

    return(0);
}
```