Moltiplicazione veloce.

Si vogliono rappresentare interi positivi di lunghezza variabile N non limitata a priori.
In tal modo, è possibile memorizzare le N cifre decimali di un qualunque intero in N celle del modello RAM, formando così una sequenza di N caratteri scelti tra {0, 1, 2, ..., 9}.
La controparte, però, risiede nel fatto che leggere o scrivere un tale intero richiede $O(N)$ tempo, in quanto vanno lette o scritte N celle.

Siano dati due numeri interi positivi A e B di N cifre ciascuno. Se vogliamo *addizionare* tali numeri, possiamo usare l'algoritmo della scuola elementare in $O(N)$ passi. Tale algoritmo è chiaramente ottimo al caso pessimo in quanto dobbiamo esaminare le N cifre sia di A che di B.
Se invece vogliamo *moltiplicare* i due numeri con l'algoritmo della scuola elementare, scopriamo che quest'ultimo impiega $O(N^2)$ passi al caso pessimo.

Vediamo in questa lezione come tale soluzione possa essere migliorata con il metodo *divide-et-impera*, ottenendo un algoritmo che richiede $O(N^{\log_2 3}) = O(N^{1.59})$ passi (si veda il paragrafo 5.2.2 in [Lu] per una discussione dettagliata).

Assumiamo che N sia una potenza del 2 e dividiamo A e B in due parti uguali di $N/2$ cifre, per cui possiamo scrivere:

$$A = A_1 * 10^{N/2} + A_2 ,$$
$$B = B_1 * 10^{N/2} + B_2 .$$

Ora esprimiamo il loro prodotto in base alle loro metà:

$$A * B = (A_1 * 10^{N/2} + A_2) * (B_1 * 10^{N/2} + B_2)$$
$$= A_1 B_1 * 10^N + (A_1 B_2 + A_2 B_1) * 10^{N/2} + A_2 B_2.$$

In altre parole, assumendo di avere delle variabili `DIECI_ENNE` e `DIECI_ENNE_MEZZI` che contengono rispettivamente i valori 10^N e $10^{N/2}$, possiamo calcolare ricorsivamente la moltiplicazione come segue.

```
int MOLTIPLICAZIONE(A, B, N) :
if (N == 1) {
      return (A * B);
else {
      // dividi A e B in due meta' uguali A1, A2 e B1, B2 rispettivamente
      X = MOLTIPLICAZIONE(A1, B1, N/2);
      Y = MOLTIPLICAZIONE(A2, B2, N/2);
      Z = MOLTIPLICAZIONE(A1, B2, N/2) + MOLTIPLICAZIONE(A2, B1, N/2);
      return X * DIECI_ENNE + Z * DIECI_ENNE_MEZZI + Y;
      }.
```

L'equazione di ricorrenza corrispondente è data da:

$T(1)$ = costante,
$T(N) = 4\ T(N/2) + O(N).$

Tale equazione può essere risolta con il ``master theorem'' in [CLR], dove $a=4, b = 2, f(N) = O(N)$, per cui abbiamo $T(N) = O(N^2)$. Non c'è quindi alcun guadagno, a meno di eseguire un numero di chiamate ricorsive inferiore ad $a=4$.

Il miglioramento si ottiene calcolando la variabile Z in una maniera alternativa che richieda una **sola** chiamata ricorsiva invece che due. Si veda il codice seguente.

```
int MOLTIPLICAZIONE(A, B, N) :
if (N == 1) {
```

```
        return (A * B);
else {
        // dividi A e B in due meta' uguali A1, A2 e B1, B2 rispettivamente
        X = MOLTIPLICAZIONE(A1, B1, N/2);
        Y = MOLTIPLICAZIONE(A2, B2, N/2);
        Z := MOLTIPLICAZIONE_VELOCE(A1+A2, B1+B2, N/2) - X - Y;
        return X * DIECI_ENNE + Z * DIECI_ENNE_MEZZI + Y;
        }.
```

Notiamo che il valore calcolato in Z nella procedura `MOLTIPLICAZIONE_VELOCE` è identico a quello calcolato nella procedura `MOLTIPLICAZIONE`, anche se meno intuitivo in quanto sommiamo le due metà di ogni numero e le moltiplichiamo ricorsivamente.

Esercizio: Dimostrare che il valore di Z è lo stesso in entrambe le procedure.

L'equazione di ricorrenza corrispondente è data da:
$T(1)$ = costante,
$T(N) = 3\ T(N/2) + O(N)$,
per cui la soluzione diventa $T(N) = O(N^{\log_2 3}) = O(N^{1.59})$ passi. Notiamo per completezza che esiste una maniera abbastanza complicata di eseguire la moltiplicazione in $O(N \log N)$ passi (mediante la Trasformata di Fourier, si veda il paragrafo 32.1 in [CLR]).

Passiamo a vedere un esempio con $A = 3587$ e $B = 2831$. Il loro prodotto è $A * B = 10154797$.
In base all'algoritmo `MOLTIPLICAZIONE_VELOCE`, dividiamo A e B in due metà, ottenendo
$A_1 = 35$, $A_2 = 87$
$B_1 = 28$, $B_2 = 31$.
Poi calcoliamo
$X = 35 * 28 = 980$
$Y = 87 * 31 = 2697$
$Z = (35+87) * (28+31) - X - Y = 3521$.
Infine, restituiamo il prodotto (con $N=4$)
$A * B = X * 10^4 + Z * 10^2 + Y = 10154797$.

Roberto Grossi, Febbraio 2001.