1 Esercizio 1

Scrivere nel file `esercizio2.cc` la dichiarazione e la definizione della funzione ricorsiva `get_elements` che prende come argomento un array di `char` `source`, la dimensione dell'array `source`, un puntatore `dest1` di `char`, la dimensione da calcolare dell'array `dest1`, un puntatore `dest2` di `char`, e la dimensione da calcolare dell'array `dest2`. Tale funzione:

- estrae dall'array `source` le lettere minuscole e le memorizza nell'ordine in cui compaiono nell'array `dest1` in modo che 'a' sia convertito in 'Z', 'b' in 'Y', ..., 'y' in 'B', e 'z' in 'A';
- estrae dall'array `source` i caratteri numerici e li memorizza nell'ordine in cui compaiono nell'array `dest2` in modo che '9' sia convertito in '0', '8' in '1', ... , '1' in '8', e '0' in '9';
- ignora le lettere maiuscole ed eventuali altri caratteri;
- calcola le dimensioni correnti degli array `dest1` e `dest2`, ed alloca gli array in base alle dimensioni calcolate.

Il programma per essere eseguito legge da standard input una sequenza di caratteri terminata da newline e produce in output la sequenza di caratteri letta, e il contenuto degli array risultato della chiamata a `get_elements`.

Questo è un esempio di esecuzione:

```
computer > echo ab01AajK39 | ./a.out
Source = a b 0 1 A a j K 3 9
D1     = Z Y Z Q
D2     = 9 8 6 0
```

Note:

- Scaricare il file `esercizio2.cc`, modificarlo per inserire la dichiarazione e la definizione della funzione `get_elements`, e caricare il file risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito.
- La funzione `get_elements` deve essere ricorsiva ed al suo interno non ci possono essere cicli o chiamate a funzioni contenenti cicli. Si può però fare uso di una sola funzione ricorsiva ausiliaria da chiamare all'interno di questa funzione.
- Le uniche assunzioni che si possono fare sull'input e su dimensioni di eventuali strutture/array utilizzate nel file di partenza fornito sono **solo quelle espressamente specificate in questo testo** (e NON quelle riportate nel file fornito, che sono SOLO indicative per consentire di svolgere l'esame).
- Non é consentito l'uso di altre funzioni ausiliare per il calcolo delle dimensioni e dell'allocazione degli array oltre quella specificata al punto precedente. Il calcolo ed allocazione devono quindi essere effettuati dall'unica funzione ausiliaria chiamata all'interno della funzione `get_elements`.
- All'interno di questo programma **non è ammesso** l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `cstddef`. In particolare **non sono ammesse** funzioni definite in `cctype` (e.g. `tolower`, `isdigit`), ...).
- Si ricorda che, l'esempio di esecuzione è puramente indicativo, e la soluzione proposta NON deve funzionare solo per l'input fornito, ma deve essere robusta a variazioni compatibili con la specifica riportata in questo testo.

2 Esercizio 2

Scrivere nel file `esercizio2.cc` la dichiarazione e la definizione della funzione ricorsiva `get_elements` che prende come argomento un array di `char` `source`, la dimensione dell'array `source`, un puntatore `dest1` di `char`, la dimensione da calcolare dell'array `dest1`, un puntatore `dest2` di `char`, e la dimensione da calcolare dell'array `dest2`. Tale funzione:

- estrae dall'array `source` le lettere minuscole e le memorizza nell'ordine in cui compaiono nell'array `dest2` in modo che 'a' sia convertito in 'Z', 'b' in 'Y', ..., 'y' in 'B', e 'z' in 'A';
- estrae dall'array `source` le lettere maiuscole e le memorizza nell'ordine in cui compaiono nell'array `dest1` in modo che 'A' sia convertito in 'z', 'B' in 'y', ..., 'Y' in 'b', e 'Z' in 'a';
- ignora ogni altro tipo di carattere (e.g. numeri o punteggiature);
- calcola le dimensioni correnti degli array `dest1` e `dest2`, ed alloca gli array in base alle dimensioni calcolate.

Il programma per essere eseguito legge da standard input una sequenza di caratteri terminata da newline e produce in output la sequenza di caratteri letta, e il contenuto degli array risultato della chiamata a `get_elements`.

Questo è un esempio di esecuzione:

```
computer > echo ab01AajK39 | ./a.out
Source = a b 0 1 A a j K 3 9
D1     = z p
D2     = Z Y Z Q
```

Note:

- Scaricare il file `esercizio2.cc`, modificarlo per inserire la dichiarazione e la definizione della funzione `get_elements`, e caricare il file risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito.
- La funzione `get_elements` deve essere ricorsiva ed al suo interno non ci possono essere cicli o chiamate a funzioni contenenti cicli. Si può però fare uso di una sola funzione ricorsiva ausiliaria da chiamare all'interno di questa funzione.
- Le uniche assunzioni che si possono fare sull'input e su dimensioni di eventuali strutture/array utilizzate nel file di partenza fornito sono **solo quelle espressamente specificate in questo testo** (e NON quelle riportate nel file fornito, che sono SOLO indicative per consentire di svolgere l'esame).
- Non é consentito l'uso di altre funzioni ausiliare per il calcolo delle dimensioni e dell'allocazione degli array oltre quella specificata al punto precedente. Il calcolo ed allocazione devono quindi essere effettuati dall'unica funzione ausiliaria chiamata all'interno della funzione `get_elements`.
- All'interno di questo programma **non è ammesso** l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `cstddef`. In particolare **non sono ammesse** funzioni definite in `cctype` (e.g. `tolower`, `isdigit`), ...).
- Si ricorda che, l'esempio di esecuzione è puramente indicativo, e la soluzione proposta NON deve funzionare solo per l'input fornito, ma deve essere robusta a variazioni compatibili con la specifica riportata in questo testo.

3 Esercizio 3

Scrivere nel file `esercizio2.cc` la dichiarazione e la definizione della funzione ricorsiva `get_elements` che prende come argomento un array di `int source`, la dimensione dell'array `source`, un puntatore `dest1` di `int`, la dimensione da calcolare dell'array `dest1`, un puntatore `dest2` di `int`, e la dimensione da calcolare dell'array `dest2`. Tale funzione:

- estrae dall'array `source` gli elementi che si trovano in posizione dispari e che sono multipli di 2 e li memorizza nell'ordine in cui compaiono nell'array `dest1` dopo averci sommato 10 e diviso il risultato per 3;
- estrae dall'array `source` gli elementi che si trovano in posizione pari e che sono multipli di 5 e li memorizza nell'ordine in cui compaiono nell'array `dest2` dopo averlo elevato al quadrato e sottratto 10;
- ignora il numero negli altri casi;
- calcola le dimensioni correnti degli array `dest1` e `dest2`, ed alloca gli array in base alle dimensioni calcolate.

Il programma per essere eseguito legge da standard input una sequenza di numeri terminata da -1 e produce in output la sequenza di caratteri letta, e il contenuto degli array risultato della chiamata a `get_elements`.

Questo è un esempio di esecuzione:

```
computer > echo 1 2 5 8 9 25 9 8 50 -1 | ./a.out
Source = 1 2 5 8 9 25 9 8 50
D1     = 4 6 6
D2     = 15 2490
```

Note:

- Scaricare il file `esercizio2.cc`, modificarlo per inserire la dichiarazione e la definizione della funzione `get_elements`, e caricare il file risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito.
- Si consideri lo 0 come pari.
- La funzione `get_elements` deve essere ricorsiva ed al suo interno non ci possono essere cicli o chiamate a funzioni contenenti cicli. Si può però fare uso di una sola funzione ricorsiva ausiliaria da chiamare all'interno di questa funzione.
- Le uniche assunzioni che si possono fare sull'input e su dimensioni di eventuali strutture/array utilizzate nel file di partenza fornito sono **solo quelle espressamente specificate in questo testo** (e NON quelle riportate nel file fornito, che sono SOLO indicative per consentire di svolgere l'esame).
- Non é consentito l'uso di altre funzioni ausiliare per il calcolo delle dimensioni e dell'allocazione degli array oltre quella specificata al punto precedente. Il calcolo ed allocazione devono quindi essere effettuati dall'unica funzione ausiliaria chiamata all'interno della funzione `get_elements`.
- All'interno di questo programma non è ammesso l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `ctypes`.

- Si ricorda che, l'esempio di esecuzione è puramente indicativo, e la soluzione proposta NON deve funzionare solo per l'input fornito, ma deve essere robusta a variazioni compatibili con la specifica riportata in questo testo.

4 Esercizio 4

Scrivere nel file `esercizio2.cc` la dichiarazione e la definizione della funzione ricorsiva `get_elements` che prende come argomento un array di `int source`, la dimensione dell'array `source`, un puntatore `dest1` di `int`, la dimensione da calcolare dell'array `dest1`, un puntatore `dest2` di `int`, e la dimensione da calcolare dell'array `dest2`. Tale funzione:

- estrae dall'array `source` gli elementi che si trovano in posizione multiple di 3 e che hanno un valore dispari e li memorizza nell'ordine in cui compaiono nell'array `dest1` dopo averlo elevato al quadrato;
- estrae dall'array `source` gli elementi che si trovano in posizione multiple di 2 e che hanno un valore divisibile per 7 e li memorizza nell'ordine in cui compaiono nell'array `dest2` dopo averli divisi per 2;
- calcola le dimensioni correnti degli array `dest1` e `dest2`, ed alloca gli array in base alle dimensioni calcolate.

Il programma per essere eseguito legge da standard input una sequenza di numeri terminata da -1 e produce in output la sequenza di caratteri letta, e il contenuto degli array risultato della chiamata a `get_elements`.

Questo è un esempio di esecuzione:

```
computer > echo 1 2 14 8 9 25 9 8 49 -1 | ./a.out
Source = 1 2 14 8 9 25 9 8 49
D1     = 1 81
D2     = 7 24
```

Note:

- Scaricare il file `esercizio2.cc`, modificarlo per inserire la dichiarazione e la definizione della funzione `get_elements`, e caricare il file risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito.
- Si consideri lo 0 come pari.
- La funzione `get_elements` deve essere ricorsiva ed al suo interno non ci possono essere cicli o chiamate a funzioni contenenti cicli. Si può però fare uso di una sola funzione ricorsiva ausiliaria da chiamare all'interno di questa funzione.
- Le uniche assunzioni che si possono fare sull'input e su dimensioni di eventuali strutture/array utilizzate nel file di partenza fornito sono **solo quelle espressamente specificate in questo testo** (e NON quelle riportate nel file fornito, che sono SOLO indicative per consentire di svolgere l'esame).
- Non é consentito l'uso di altre funzioni ausiliare per il calcolo delle dimensioni e dell'allocazione degli array oltre quella specificata al punto precedente. Il calcolo ed allocazione devono quindi essere effettuati dall'unica funzione ausiliaria chiamata all'interno della funzione `get_elements`.
- All'interno di questo programma non è ammesso l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `ctypes`.
- Si ricorda che, l'esempio di esecuzione è puramente indicativo, e la soluzione proposta NON deve funzionare solo per l'input fornito, ma deve essere robusta a variazioni compatibili con la specifica riportata in questo testo.