Esercizio 1. All'interno di una matrice di caratteri di dimensione N*N è nascosta una frase. Per ricostruirla è necessario scandire la matrice per colonne, dall'alto verso il basso, partendo da una certa posizione, che potrà variare da colonna a colonna. L'esempio seguente spiega come effettuare la ricostruzione.

(S)	S	I	I	N
A	O	B	M	I
B	N	E	(O)	(P)
A	O	R	D	E
T	(O)	(L)	A	G

Si scandisce prima la prima colonna partendo dalla prima cella in alto contenente la S e ottenendo SABAT. Si prende l'ultima lettera considerata, la T in questo caso, e la si usa per determinare la posizione dalla quale deve partire la scansione della seconda colonna. In particolare, partendo da 0, la T è la lettera numero 19 nell'alfabeto, e poiché le righe della matrice sono 5, la lettera T indica la posizione 4 (resto della divisione tra 19 e 5). Quindi la scansione della seconda colonna parte dalla O in posizione 4 e si ottiene SABATOSONO. L'ultima lettera inserita è una O, la numero 14 nell'alfabeto, che di nuovo indica la posizione 4, perciò si riprende con la terza colonna dalla lettera L, e si ottiene SABATOSONOLIBER. Si continua così per tutte le colonne, ottenendo SABATOSONOLIBERODAIMPEGNI.

Si scriva in C++ una funzione che ricevuta una matrice di dimensione N*N, stampi la stringa risultante dopo la ricostruzione. Per semplicità, si può assumere che la matrice contenga solo lettere dell'alfabeto inglese maiuscole.

Esercizio 2 Un numero naturale si dice **abbondante** se è minore della somma dei suoi divisori interi, escludendo sé stesso. Ad esempio, 12 è un numero abbondante. Infatti, i divisori di 12 (escluso il 12 stesso) sono 1,2,3,4 e 6 e si ha che 12 < (1+2+3+4+6). Scrivere una funzione booleana ricorsiva che, ricevuto come parametro (almeno) un numero naturale N, restituisca true se N è un numero abbondante e false altrimenti. Specificare come deve essere invocata la prima volta la funzione.

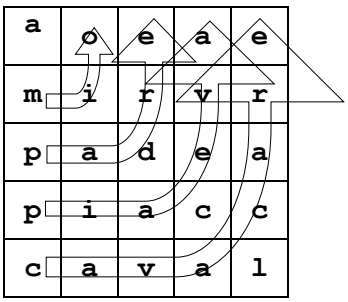
Esercizio 3.
Sia la funzione f(n) definita come segue:

$$\begin{cases} f(n) = 1 & \text{, se } n = 1 \\ f(n) = f(n-1) + 2 \cdot n - 1 & \text{, se } n > 1 \end{cases}$$

Sia dato un vettore V di caratteri contenente una sequenza di parole. Ogni parola è composta da una sequenza di lettere terminata da un trattino (il carattere '-'), mentre il vettore è terminato da un punto (il carattere '.'). Implementare in C++ una funzione che determini se il vettore rispetta la proprietà così descritta: i caratteri che compongono le prime N parole di V (quindi senza considerare i trattini) sono esattamente f(n), per ogni n da 1 al numero di parole in V. Se la proprietà non è rispettata, la funzione restituisce "false". Altrimenti, restituisce "true", dopo aver stampato una matrice in cui la parola di indice n+1 occupa le posizioni (n,0) ... (n, n-1) (n,n) (n-1,n) ... (0, n)
Esempio: Se V contiene "a-mio-padre-piaceva-cavalcare." la funzione restituisce true perché:

- la prima parola è composta da esattamente F(1) = 1 carattere;
- le prime 2 parole sono composte da esattamente F(2) = 4 caratteri;
- le prime 3 parole sono composte da esattamente F(3) = 9 caratteri;
- le prime 4 parole sono composte da esattamente F(4) = 16 caratteri;
- le prime 5 parole sono composte da esattamente F(5) = 25 caratteri.
- La matrice che la funzione stampa prima di restituire true è
-

CAMPUS DI ARCAVACATA http://www.unical.it/portale/strutture/dipartimenti_240/matinf/
87036 Arcavacata di Rende (CS) - Ponte Pietro Bucci Cubo 30B
tel. (+39) 0984 496402 - fax (+39) 0984 496410



CAMPUS DI ARCAVACATA http://www.unical.it/portale/strutture/dipartimenti_240/matinf/
87036 Arcavacata di Rende (CS) - Ponte Pietro Bucci Cubo 30B
tel. (+39) 0984 496402 - fax (+39) 0984 496410