ESERCIZIO 1

Introduzione:

Tutti conoscono il prodotto notevole:

$(a + b)^2 = a^2 + 2ab + b^2$

E magari anche:

$(a + b)^3 = a^3 + 3a^2b + 3ab^2 + b^3$

Le formule si somigliano; non è un caso: il risultato può essere generalizzato:

$$(a+b)^n = \sum_{i=0}^{n} \binom{n}{i} a^i b^{n-i}$$

Dove:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

ESERCIZIO 1

Esempio:

$$(a+b)^3=\binom{3}{0}a^0b^{3-0}+\binom{3}{1}a^1b^{3-1}+\binom{3}{2}a^2b^{3-2}+\binom{3}{3}a^3b^{3-3}$$

Dove (0! = 1, per definizione):

$$\binom{3}{0}=\frac{3!}{0!(3-0)!}=\frac{3*2*1}{1*3*2*1}=1$$

$$\binom{3}{1}=\frac{3!}{1!(3-1)!}=3$$

$$\binom{3}{2}=\frac{3!}{2!(3-2)!}=3$$

$$\binom{3}{3}=\frac{3!}{3!(3-3)!}=1$$

...In pratica abbiamo lo stesso risultato di prima, ma adesso sappiamo perché

ESERCIZIO 1

Esercizio:

Definire una funzione:

```
int potenza_bin(int a, int b, int n);
```

Che dati due interi *a*, *b* e un esponente *n* restituisca $(a+b)^n$, calcolato usando la formula vista.

Chiaramente andiamo per gradi...

...Per ognuna delle funzioni seguenti costruire un piccolo main di prova che chieda i dati che servono all'utente e stampi il risultato.

ESERCIZIO 1

Parte 1:
Si scriva una funzione

```
int fatt(int n);
```

che dato un intero n restituisca n! = n*(n-1)*(n-1)...
Si noti che 0! = 1

Esempio:
Se si chiama `fatt(4)`:
`fatt(4) = 4 * 3 * 2 * 1 = 24`

ESERCIZIO 1

Parte 2:
Si scriva una funzione

```
int coeff_bin(int n, int k);
```

che dati due interi *n* e *k* restituisca :

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

chiamando la funzione `fatt`

Esempio:
Se si chiama `coeff_bin(3,2)`:

```
coeff_bin(3,2) = 3*2*1/2*1*(1) = 3
```

ESERCIZIO 1

Parte 3:
Si scriva una funzione

```
int potenza(int a, int n);
```

che dato un intero *a* e un esponente *n* restituisca: a^n

Esempio:
Se si chiama `potenza(3,2)`:
`potenza(3,2) = 3*3 = 9`

ESERCIZIO 1

Parte 4:
Si scriva una funzione

```
int potenza_bin(int a, int b, int n);
```

che dati due interi *a, b* e un esponente *n* restituisca:

$$(a+b)^n = \sum_{i=0}^{n} \binom{n}{i} a^i b^{n-i}$$

Esempio:
Se si chiama `potenza_bin(3,2,2)`:

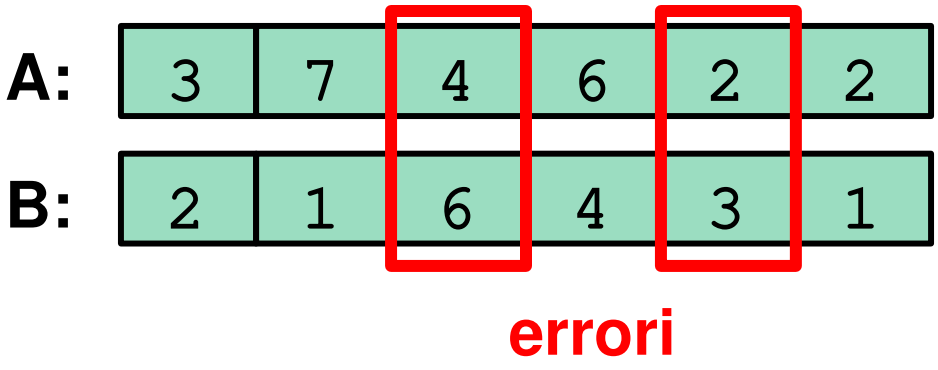
`potenza(3,2,2)= 1*3`^{2} `+ 2*3*2 + 1*2`^{2}`=9+12+4 = 25`

ESERCIZIO 2

Introduzione:

Vogliamo correggere degli errori di misura. Siano date due serie di misure *A* e *B*; entrambe contengono lo stesso numero di elementi; la serie *A* dovrebbe contenere dati maggiori o uguali di quelli della serie *B.* In pratica questo potrebbe non succedere.

Esempio:



ESERCIZIO 2

Parte 1:

Si costruisca un programma che chiede all'utente un numero di elementi da inserire *n (*al massimo n =10), e quindi chieda all'utente di immettere due array *A* e *B* di *n* elementi.

Si definisca una procedura

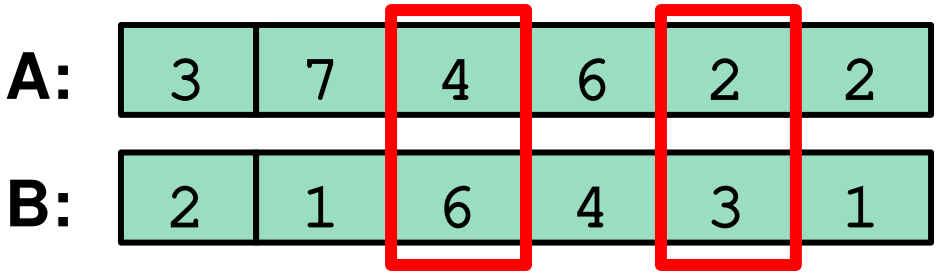
```
void controlla(int[] A, int[] B, int n,
                    int* nErr, int* maxErr);
```

Che al termine della sua esecuzione in *nErr* inserisca il numero di celle per cui `A[i]<B[i]` e in *maxErr* inserisca la massima differenza `B[i]-A[i]` per tali celle

ESERCIZIO 2

Esempio:

Date




```
nErr = 2
maxErr = max(6-4,3-2) = 2
```

ESERCIZIO 2

Parte 2:

Vogliamo correggere le due serie in modo da eliminare le anomalie. Si definisca una procedura:

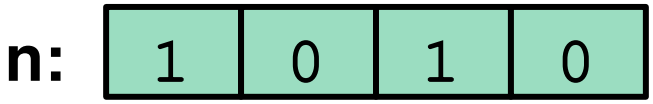
```
void correggi(int* a, int* b);
```

Che, se l'intero *a* è minore di quello *b* assegni ad *a* il valore di *b*. Si utilizzi la procedura per correggere gli array immessi dall'utente.

ESERCIZIO 3

Parte 1:

Dato un array di interi compresi tra 0 e 1 di dimensione dim:



Si definisca una funzione

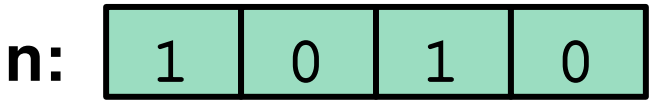
```
int zero(int[] n, int dim);
```

Che restituisce 1 se tutte le celle dell'array contengono il valore 0; altrimenti resituisce 0.

ESERCIZIO 3

Parte 2:

Dato un array di interi di dimensione dim compresi tra 0 e 1:



Si definisca una procedura

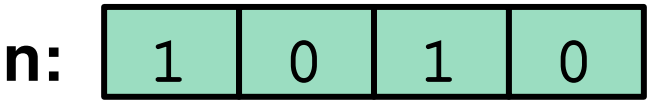
```
void reset(int[] n, int dim);
```

Che assegni 0 a tutte le celle dell'array

ESERCIZIO 3

Parte 3:

Dato un array di interi compresi tra 0 e 1 di dimensione dim:



Si definisca una procedura

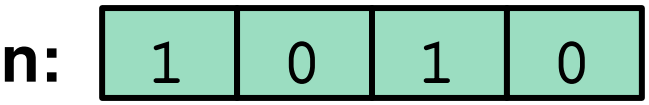
```
int sposta_dx(int[] n, int dim);
```

Che dato l'array, faccia "scorrere" l'array verso destra e restituisca il numero che "esce" dall'array

ESERCIZIO 3

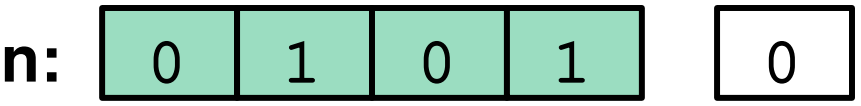
Esempio:

Dato:

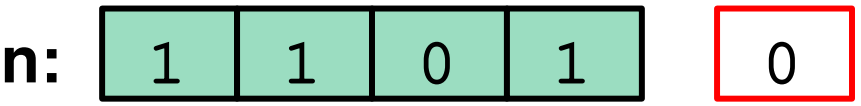



```
sposta_dx(n,4);
```

Fa scorrere l'array:



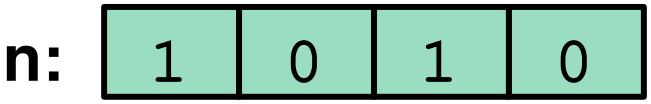
Restituisce 0:



ESERCIZIO 3

Parte 3:

Dato un array di interi compresi tra 0 e 1 di dimensione dim:



Si definisca una procedura

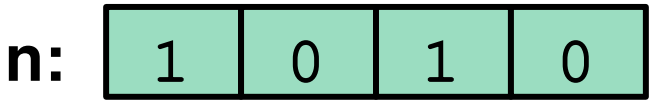
```
int sposta_sx(int[] n, int dim);
```

Che dato l'array, faccia "scorrere" l'array verso sinistra e restituisca il numero che "esce" dall'array

ESERCIZIO 3

Parte 4:

Dato un array di interi compresi tra 0 e 1 di dimensione dim:



Si definisca una procedura

```
int sum(int[] n, int[] n2, int dim);
```

Che dati due array *n* e *n2* sommi ad ogni cella di *n* il contenuto della cella corrispondente in *n2*. Se il risultato è maggiore di 1 la cella rimane ad 1.