Esercizio 1

Le serie di Maclaurin sono metodi conveniente per approssimare funzioni difficili. La serie di Maclaurin della funzione esponenziale è la seguente:

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} \qquad \text{per ogni } x$$

Si costruisca un programma che calcoli l'esponenziale attraverso la serie di Maclaurin (espansa all' *n* inserito dall'utente) e la si confronti con il risultato della vera funzione esponenziale (suggerimento: usare la funzione exp(esponente) contenuta in Math.h

Esercizio 2

Dati in ingresso massimo 20 numeri (ma possono essere anche meno, terminare la lettura inserendo un qualsiasi numero negativo), scrivere un programma che, attraverso funzioni, permetta di:

1. Calcolare la somma di tutti i numeri inseriti, il minimo ed il massimo;
2. Calcolare la media (usando la funzione precedente), escludendo il numero massimo e il numero minimo
3. Restituire solo i numeri dispari (non usare i puntatori)
4. Determinare se il numero è primo o meno (restituisce 1 se primo, 0 se non primo)
5. Restituire solo i numeri primi (nuovo array).

Esercizio 3

Scrivere una funzione che permetta di convertire un numero da una base a un'altra. Questo, per qualsiasi base di partenza e destinazione (scelte dall'utente)

Esercizio 4

Scrivete un programma che generi a caso due array di N=10 elementi interi, il cui valore sia compreso tra 1 a 200.

Si richiede di:

- Creare una funzione per generare gli array;
- Creare una funzione che stampi gli array generati;
- Creare una procedura che costruisca un terzo array di dimensione 2N i cui elementi di posizione pari siano gli elementi del primo vettore e gli elementi di posizione dispari siano gli elementi del secondo vettore

Per generare numeri casuali usare la funzione rand()

```
#include <time.h>
#include <stdlib.h>
…
int main(){
srand(time(NULL));    //solo all'inizio del main
int r = rand();  //genera un int casuale da 0 a RANDMAX
int g = rand() % 20;  //genera un int casuale da 0 a 19
```

Esercizio 5

Realizzare le seguenti funzioni, che presa in input una stringa:

- Restituiscano tutti i caratteri in maiuscolo
  toUpperCase("Hello World") = "HELLO WORLD"

- Inverta il case dei caratteri
  invertCase("Hello World") = "hELLO wORLD"

- Converta il Camel Case in Snake Case
  invertCase("HelloWorld") = "Hello_World"

Esercizio 6

Dati N prodotti in M categorie, realizzare:

- Un array di N elementi, contenente i prezzi di N oggetti
- Un array di N elementi, dove, per ogni elemento dell'array precedente, venga specificata una sua categoria (numero da 0 a M)
- Un array di M elementi, contenente gli sconti da applicare a ciascuna categoria. I valori di questo array devono essere casuali.

- Una funzione che permetta di ottenere l'array degli N prodotti scontati e visualizzarli.

Esercizio 7

Realizzare una funzione che, dato in ingresso una stringa, la filtri attraverso una maschera (un array di elementi 0/1: 1 elemento consentito, 0 elemento eliminato).

Stringa	N	e	l		m	e	z	z	o		d	e	l		c	a	m	m	i	n		d	i		n	o	s	t	r	a
Maschera	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
Risultato		e	l			e														n										a

Il risultato deve essere fornito sotto forma di stringa. (es. precedente "elena")

Si calcoli la lunghezza della stringa così costruita e la riduzione percentuale del testo rispetto l'originale (es. precedente, lunghezza iniziale 30 caratteri, finale 5, riduzione dell' 83.3%)

Esercizio 8

Si scriva una funzione in linguaggio C che prenda in ingresso il seguente array di 22 interi

```
int array_in[DIM] = {67, 79, 77, 80, 76, 73, 77, 69, 78,
                     84, 73, 33, 72, 97, 105, 70, 105,
                     110, 105, 116, 111, 33};
```

ed un puntatore ad un secondo array di char. La funzione riempie il secondo array con i valori del primo array convertiti nel rispettivo carattere (secondo il codice ASCII). Non utilizzare gli indici per gli array, ma solo la logica dei puntatori.