```
"""
Cap 6
"""
import numpy as np
import matplotlib.pyplot as pl
import networkx as nx
G=nx.Graph()
COLORS = ['#8dd3c7','#ffffb3','#bebada','#fb8072','#80b1d3','#fdb462',
          '#b3de69','#fccde5','#d9d9d9','#bc80bd','#ccebc5','#ffed6f']
from numpy.random import random

def reachable_nodes(G, start):  #nodi raggiungibili da start = nodo di partenza
    seen = set()
    stack = [start]
    while stack:
        node = stack.pop()
        if node not in seen:
            seen.add(node)
            stack.extend(G.neighbors(node))
    return seen

def flip(p):                    #da' vero per una probabilita' p, falso per 1-p
    return random()<p

def random_pairs(nodes, p):        #aggiunge archi al grafo con probabilità p
    for i, u in enumerate(nodes):
        for j, v in enumerate(nodes):
            if i<j and flip(p):
               yield u, v

def make_random_graph(n, p):      #genera un grafo casuale ER
    G = nx.Graph()
    nodes = range(n)
    G.add_nodes_from(nodes)
    G.add_edges_from(random_pairs(nodes, p))
    return G

n=int(input('numero di nodi = '))
p=float(input("probabilita' = "))
print()
i=int(input('numero di iterazioni= '))

def is_connected(G):                      #verifica se il grafo è connesso
    start = list(G) [0]                   #per ogni iterazione
    reachable = reachable_nodes(G, start)
    return len(reachable) == len(G)

def prob_connected(n, p, iters=i):        #calcola la probabilità di connettività
    count = 0                             #come rapporto tra numero di grafi
    for i in range(iters):                #connessi e numero totale
        random_graph = make_random_graph(n, p)
        if is_connected(random_graph):
            count += 1
    return count/iters
a=prob_connected(n,p)
print()
#prob_connected(n,p)
ps=np.logspace(-2.5, 0 , 20)
print('La probabilità di connessione è ',a)
ys=[prob_connected(n,p) for p in ps]

fig, ax = plt.subplots(facecolor='teal', alpha=0.1)
plt.style.use('ggplot')     #sottofondo grigio
```

```
pl.xlabel('probabilità degli archi',color='orange')
pl.ylabel('probabilità di connessione',color='orange')
pl.tick_params(labelcolor='tab:orange')

pl.grid(True)
pl.plot(ps,ys)
pl.show()
```