METODI UTILI IN PYTHON

Se volete provare gli algoritmi per la manipolazione delle strutture dati, vi è utile generarle.
Di seguito, alcuni metodi per farlo.

LISTE CONCATENATE:

```
class Nodo:
     def __init__(self, key=None, next=None):
          self.key = key
          self.next = next

def Crea(A):
     ''' genera una lista concatenata usando come chiavi dei nodi
     gli elementi di un array A e termina restituendo il puntatore
     alla testa della lista creata '''
     p = None
     for i in range(len(A), 0, -1):
          p = Nodo(A[i-1], p)
     return p
```

ALBERI BINARI:

```
class NodoAB:
     def __init__(self, key=None, left=None, right=None):
          self.key = key
          self.left = left
          self.right=right

import random

def generaAlbero(n, m):
     ''' genera un albero binario casuale di n nodi le cui chiavi
     sono interi tra 1 e m e restituisce il puntatore alla radice
     dell'albero '''
     if n == 0:
         return None
     p = NodoAB(random.randint(1, m))
     s = random.randint(0, n-1)
     p.left = generaAlbero(s, m)
     p.right = generaAlbero(n-1-s, m)
     return p
```

ALBERI BINARI DI RICERCA:

```
class NodoABR:
     def __init__(self, key=None, parent=None, left=None, right=None):
          self.key = key
          self.parent = parent
          self.left = left
          self.right = right
```

```
import random

def generaABR(n):
     ''' genera un albero di ricerca casuale con n nodi con chiavi
     nell'intervallo [0...5n] e restituisce il puntatore alla
     radice dell'albero '''
    if n==0:
          return None
     # genera la struttura di un albero binario di ricerca   con n nodi
     p = generaABR1(n)
     # genera la lista delle n chiavi nell'intervallo [0..5n]
     # per gli n nodi dell'albero binario di ricerca
     lista = random.sample([x for x in range(5*n)], n)
     lista.sort(reverse=True)
     # inserisce le n chiavi  negli n nodi dell'albero binario di ricerca
     inserisciChiavi(p, lista)
     return p



def generaABR1(n):
      if n == 0:
            return None
      p = NodoABR()
      s = random.randint(0, n-1)
      p.left=generaABR1(s)
      if p.left:
            p.left.parent = p
      p.right=generaABR1(n-1-s)
      if p.right:
            p.right.parent = p
      return p



def inserisciChiavi(p, lista):
      if p:
            inserisciChiavi(p.left, lista)
            p.key=lista.pop()
            inserisciChiavi(p.right, lista)
```