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(p):
#genera una lista concatenata usando come chiavi gli elementi di
un array e termina restituendo la testa alla lista creata
     p = None
     for i in range(len(A),0,-1):
          P = Node(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):
     if n==0: return None
     p=NodoAB(random.randint(1,m))
     n-=1
     if n>0:
          s=random.randint(0,n)
          p.left=generaAlbero(s,m)
          p.right=generaAlbero(n-s,m)
     return p
```

ALBERI BINARI DI RICERCA:

```
import random

def generaABR(n):
#genera la struttura di un albero binario con n nodi con chiavi
nell'intervallo [0...5n]
     if n==0: return None
     p= generaABR1(n)
     lista=random.sample([x for x in range(5*n)],n)
     #genera la lista delle chiavi
```

```
    lista.sort(reverse=True)
    #inserisce le n chiavi negli n nodi dell'albero
    inserisciChiavi(p,lista)
    return p

def generaABR1(n):
      if n==0: return None
      p=NodoABR()
      s=random.randint(0,n)
      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)
```