Nota 6.15

```
# 0615.py

class ldn (list):
  def __add__ (A,t): return ldn(lista=[x+t for x in A])
  def __and__ (A,B): return ldn(lista=list(A)+list(B)) # Concatenazione
  def __div__ (A,t): t=float(t); return ldn(lista=[x/t for x in A])
  def __iadd__ (A,t): v=A+t; del A[:]; A.extend(v); return A
  def __idiv__ (A,t): v=A/t; del A[:]; A.extend(v); return A
  def __imul__ (A,t): v=A*t; del A[:]; A.extend(v); return A
  def __init__ (A,a=0,b=1,n=None,lista=None,ap=0):
    if lista==None:
      dx=float(b-a)/n
      if ap==0: r=xrange(0,n+1)
      elif ap==1: r=xrange(1,n) # intervallo aperto
      elif ap==2: r=xrange(1,n+1) # aperto a sinistra
      else: r=xrange(0,n) # aperto a destra
      a=float(a); A.extend([a+i*dx for i in r])
    else: A.extend(map(float,lista))
  def __isub__ (A,t): v=A-t; del A[:]; A.extend(v); return A
  def __mul__ (A,t): return ldn(lista=[x*t for x in A])
  def __radd__ (A,t): return A+t
  def __rand__ (A,u): return ldn(lista=list(u)+list(A))
  def __rmul__ (A,t): return A*t
  def __sub__ (A,t): return ldn(lista=[x-t for x in A])

a=ldn(a=4,b=6,n=8)
print a
# [4.0, 4.25, 4.5, 4.75, 5.0, 5.25, 5.5, 5.75, 6.0]

b=ldn(lista=[6,5,3,1,0])
print b
# [6.0, 5.0, 3.0, 1.0, 0.0]

print b+10
# [16.0, 15.0, 13.0, 11.0, 10.0]

print b/2
# [3.0, 2.5, 1.5, 0.5, 0.0]
```

Osservazione 6.21

```
if type(a)==types.IntType: ...
if type(a)==types.FloatType: ...
if type(a)==types.ComplexType: ...
if type(a)==types.FunctionType: ...
if type(a)==types.LambdaType: ...
if type(a)==types.StringType: ...
if type(a)==types.ListType: ...
if type(a)==types.DictionaryType: ...
if type(a)==types.FileType: ...
if type(a)==types.ClassType: ...
```