In [1]:
```
#Given a string and a non-negative int n, return a larger string that is n c


#string_times('Hi', 2) → 'HiHi'
#string_times('Hi', 3) → 'HiHiHi'
#string_times('Hi', 1) → 'Hi'

def string_times(str, n):
  r=""
  for i in range(n):
    r=r+str
  return r
string_times('Hi',2)

```

Out[1]: 'HiHi'

In [2]:
```
#Given a non-empty string like "Code" return a string like "CCoCodCode".


#string_splosion('Code') → 'CCoCodCode'
#string_splosion('abc') → 'aababc'
#string_splosion('ab') → 'aab'


def string_splosion(str):
  r=""
  for i in range(len(str)):
    r=r+str[:i+1]
  return r
string_splosion('code')


```

Out[2]: 'ccocodcode'

In [3]:
```
#Given an array of ints, return True if one of the first 4 elements in the a


#array_front9([1, 2, 9, 3, 4]) → True
#array_front9([1, 2, 3, 4, 9]) → False
#array_front9([1, 2, 3, 4, 5]) → False


def array_front9(nums):
  e=len(nums)
  if e > 4:
    e = 4
  for i in range(e):
    if nums[i]== 9:
      return True

  return False

array_front9([1,2,9,3,4])
```

Out[3]: True

In [5]:
```
#Given a string and a non-negative int n, we'll say that the front of the st


#front_times('Chocolate', 2) → 'ChoCho'
#front_times('Chocolate', 3) → 'ChoChoCho'
#front_times('Abc', 3) → 'AbcAbcAbc'

def front_times(str, n):
  fl=3
  if fl > len(str):
    fl=len(str)
  f=str[:fl]
  r=""
  for i in range(n):
    r=r+f
  return r
front_times('Abc',3)
```

Out[5]: 'AbcAbcAbc'

In [6]:
```
#Given an array of ints, return the number of 9's in the array.


#array_count9([1, 2, 9]) → 1
#array_count9([1, 9, 9]) → 2
#array_count9([1, 9, 9, 3, 9]) → 3

def array_count9(nums):
  c=0
  for i in nums:
    if(i==9):
      c=c+1
  return c
array_count9([1,2,9])

```

Out[6]: 1

In [ ]:
```

```