1. Create a function that takes a number n (integer greater than zero) as an argument, and returns 2 if n is odd and 8 if n is even.
You can only use the following arithmetic operators: addition of numbers +, subtraction of numbers -, multiplication of number *, division of number /, and exponentiation **.
You are not allowed to use any other methods in this challenge (i.e. no if statements, comparison operators, etc).
Examples
f(1) → 2
f(2) → 8
f(3) → 2

Ans:

In [19]:
```
def f(n):
    Even_Odd = [8,2]
    output = Even_Odd[int(n)%2]
    print (f"f({n}) → {output}")
f(1)
f(2)
f(3)
```

```
f(1) → 2
f(2) → 8
f(3) → 2
```

2. Create a function that returns the majority vote in a list. A majority vote is an element that occurs > N/2 times in a list (where N is the length of the list).

Examples:
majority_vote(["A", "A", "B"]) → "A"
majority_vote(["A", "A", "A", "B", "C", "A"]) → "A"
majority_vote(["A", "B", "B", "A", "C", "C"]) → None

Ans:

In [1]:
```
def majority_vote(inlist):
    N = len(inlist)
    output = None
    for i in inlist:
        if inlist.count(i)>N/2:
            output = i
    print(f'majority_vote({inlist}) →{output}')
majority_vote(["A", "A", "B"])
majority_vote(["A", "A", "A", "B", "C", "A"])
majority_vote(["A", "B", "B", "A", "C", "C"])
```

```
majority_vote(['A', 'A', 'B']) →A
majority_vote(['A', 'A', 'A', 'B', 'C', 'A']) →A
majority_vote(['A', 'B', 'B', 'A', 'C', 'C']) →None
```

3. Create a function that takes a string txt and censors any word from a given list lst. The text removed must be replaced by the given character char.

Examples:
censor_string("Today is a Wednesday!", ["Today", "a"], "-") → "----- is - Wednesday!"

censor_string("The cow jumped over the moon.", ["cow", "over"], "*"), "The *** jumped **** the moon.")
censor_string("Why did the chicken cross the road ?", ["Did", "chicken", "road"], "*") ⟶ "Why *** the ******* cross the ****?"

Ans:

In [18]:
```
def censor_string(string, inlist, char):
    index = 0
    tex = string.split()
    outstr = ''
    for i in tex:
        if i in inlist:
            tex[index] = char*len(i)
        index += 1
    outstr =' '.join(tex)
    print(f'censor_string({string}, {inlist}, {char}) ⟶ {outstr}')
censor_string("Today is a Wednesday!", ["Today", "a"], "-")
censor_string("The cow jumped over the moon.", ["cow", "over"], "*")
censor_string("Why did the chicken cross the road ?", ["Did", "chicken", "road"], "*")
```

```
censor_string(Today is a Wednesday!, ['Today', 'a'], -) ⟶ ----- is - Wednesday!
censor_string(The cow jumped over the moon., ['cow', 'over'], *) ⟶ The *** jumped **** the moon.
censor_string(Why did the chicken cross the road ?, ['Did', 'chicken', 'road'], *) ⟶ Why did the ******* cross the **** ?
```

4. In mathematics a Polydivisible Number (or magic number) is a number in a given number base with digits abcde... that has the following properties:

Its first digit a is not 0.
The number formed by its first two digits ab is a multiple of 2.
The number formed by its first three digits abc is a multiple of 3.
The number formed by its first four digits abcd is a multiple of 4. Create a function which takes an integer n and returns True if the given number is a Polydivisible Number and False otherwise.

Examples:
`is_polydivisible(1232) ⟶ True

1 / 1 = 1
12 / 2 = 6
123 / 3 = 41
1232 / 4 = 308
is_polydivisible(123220) ⟶ False

1 / 1 = 1
12 / 2 = 6
123 / 3 = 41
1232 / 4 = 308
12322 / 5 = 2464.4 # Not a Whole Number
123220 /6 = 220536.333... # Not a Whole Number

Ans:

In [24]:

```
def is_polydivisible(N):
    n = str(N)
    output = False
    for i in range(len(n)):
        if N%10!=0 and int(n[:i+1])%(i+1) == 0:
            output = True
    print(f'is_polydivisible({N}) → {output}')
is_polydivisible(1232)
is_polydivisible(123220)
```

```
is_polydivisible(1232) → True
is_polydivisible(123220) → False
```

5. Create a function that takes a list of numbers and returns the sum of all prime numbers in the list.

Examples:
sum_primes([1, 2, 3, 4, 5, 6, 7, 8, 9, 10]) → 17
sum_primes([2, 3, 4, 11, 20, 50, 71]) → 87
sum_primes([]) → None

Ans:

In [13]:

```
def sum_primes(inlist):
    prime = [2,3]
    sum1 = 0
    sum2 = 0
    for i in inlist:
            if i in prime:
                sum1 += i
            elif i in [6*n-1 for n in range(0,i)] or i in [6*n+1 for n in range(0,i)]:
                sum2 += i
    if 1 in inlist:
        sum2 = sum2 - 1
    print(f'sum_primes({inlist}) → {sum1+sum2}')
sum_primes([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
sum_primes([2, 3, 4, 11, 20, 50, 71])
sum_primes([])
```

```
sum_primes([1, 2, 3, 4, 5, 6, 7, 8, 9, 10]) → 17
sum_primes([2, 3, 4, 11, 20, 50, 71]) → 87
sum_primes([]) → 0
```