Solutions to Big Oh Tutorial

IMPORTANT - Each question in this notebook provides both annotated solutions of the proofs and a visualisation for sanity checking the values of c and n_0 make sense. The important part is the proof and methodology to reach the solution. You will not have access to the plots in your exams and in-class tests. Furthermore the plots themselves are not proofs.

Note: you need to run the below two code cells before you can use any of the plots to sanity check your answer(s).

In [2]:
```
import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [3]:
```
def plot_oh(f, g, c, n_0, fn, cgn, min_x, max_x):
    xs = np.arange(min_x,max_x,1, dtype='int64')
    fn_ys = f(xs)
    gn_ys = c*g(xs)
    plt.axvline(x = n_0, color = 'k', linestyle='--')
    plt.scatter(x=xs, y=fn_ys, marker='x')
    plt.scatter(x=xs, y=gn_ys, marker='.')
    plt.xlabel('n')
    plt.ylabel('f(n)')
    plt.legend(['n_0 = '+str(n_0),'f(n): '+fn, 'cg(n):'+cgn])
```

Big-Oh Definition

Given positive functions $f(n)$ and $g(n)$, we can say that $f(n)$ is $O(g(n))$ if and only if there exists positive constants c and n_0 such that $f(n) \leq c \cdot g(n), \forall n \geq n_0$.

Q1. Prove that 5 is $O(1)$

Solution

Picking apart the definition, we have that:

- $f(n) = 5$
- $g(n) = 1$

We need to show that there exists positive constants c and n_0 that satisfies:

$$5 \leq c \cdot 1, \forall n \geq n_0$$

This is trivial since we just need to choose a value of $c \geq 5$ and any positive value for n_0. This gives us:

$$5 \leq 5 \cdot 1, \forall n \geq 1$$

Notice that we chose a value for n_0 despite our inequality not depending on n. This is required from the definition: $\exists c \geq 0, \exists n_0 \geq 0$...

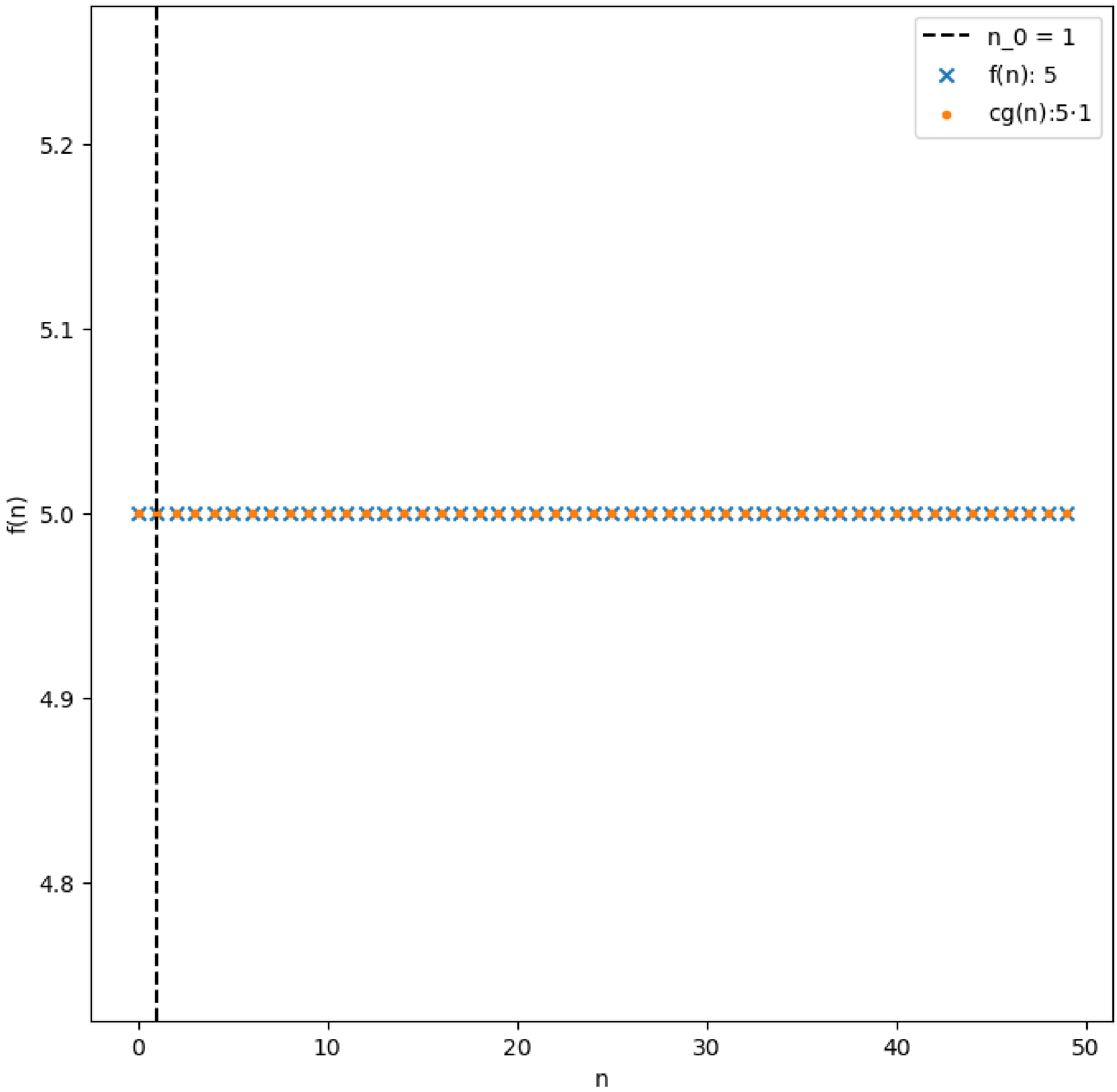
This reduces to the following inequality which is trivially true:

$$5 \leq 5, \forall n \geq 1$$

$\therefore 5$ is $O(1)$ using $c = 5$ and $n_0 = 1$

Sanity check

In [10]:
```
c = 5
n_0 = 1
f = lambda n: 0*n+5
g = lambda n: 0*n+1

plot_oh(f, g, c, n_0, '$5$', str(c)+'$\cdot 1$', 0, 50)
#plt.rcParams["figure.figsize"] = (4,4)
```



Q2. Prove that $2n + 1$ is $O(3n)$

Solution

To recall the definition of Big-Oh: Given positive functions $f(n)$ and $g(n)$, we can say that $f(n)$ is $O(g(n))$ if and only if there exists positive constants c and n_0 such that $f(n) \leq c \cdot g(n), \forall n \geq n_0$.

As before, to prove that $2n+1$ is $O(3n)$, we need to substitute the definition which gives:

$$2n+1 \text{ is } O(3n) \text{ if and only if there exists positive constants } c \text{ and } n_0 \text{ such that } 2n+1 \leq$$

If we choose $c = 1$, we are left to show that there exists an $n > 0$ such that $2n+1 \leq 3n, \forall n \geq n_0$.

But first we can simplify the equality by subtracting $2n$ from both sides to give $1 \leq n, \forall n \geq n_0$.

We can trivially pick $n_0 = 1$ to satisfy the equation

$$1 \leq n, \forall n \geq 1$$

Step-wise:

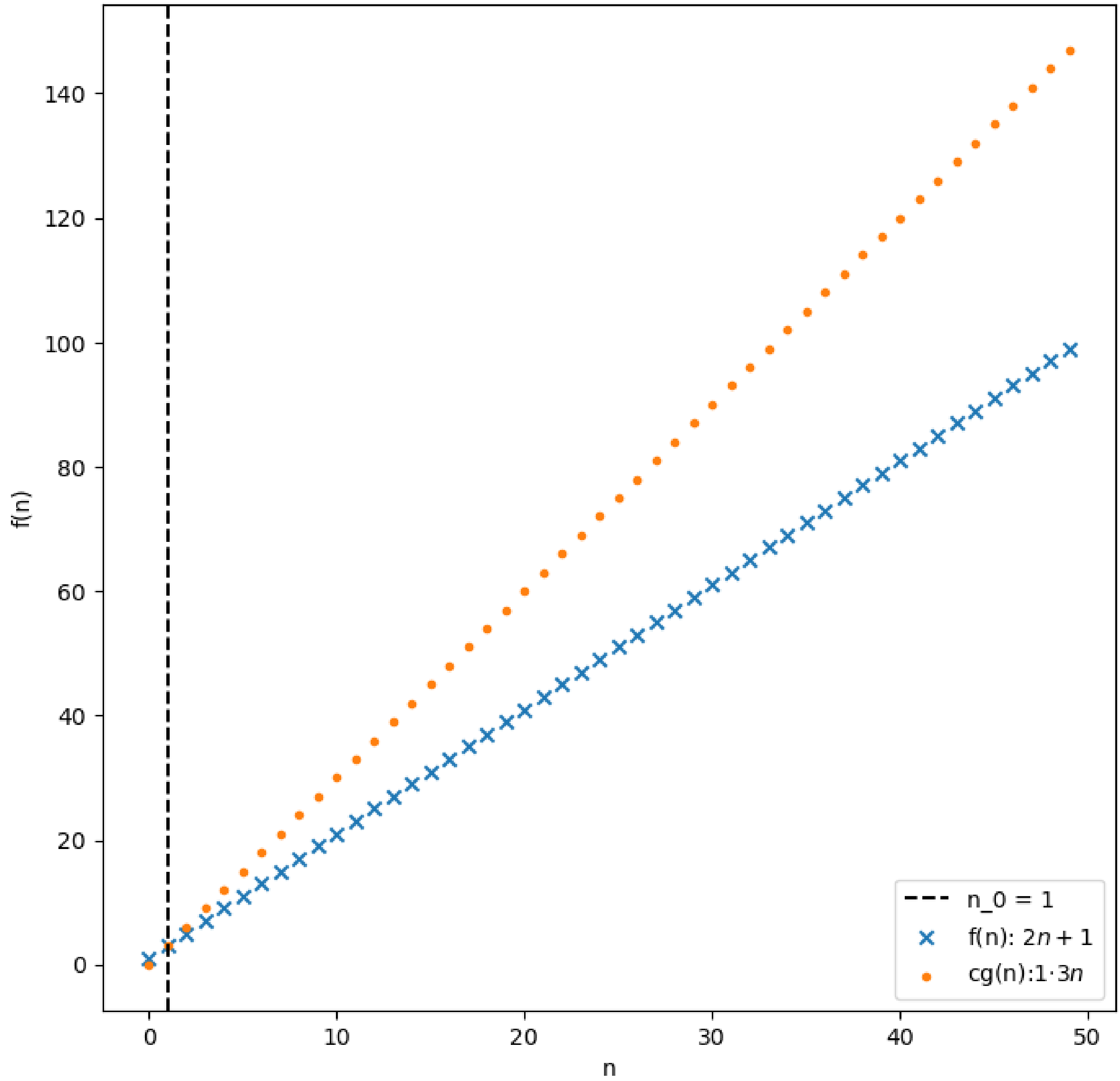
$$2n+1 \text{ is } O(3n) \text{ if and only if there exists positive constants } c \text{ and } n_0 \text{ such that } 2n+1 \leq$$

- choosing $c = 1$ gives $2n+1 \leq 3n, \forall n \geq n_0$
- simplifying by subtracting $2n$ gives $1 \leq n, \forall n \geq n_0$
- picking $n_0 = 1$ gives $1 \leq n, \forall n \geq 1$
- $1 \leq n, \forall n \geq 1$ is trivially true

$\therefore 2n+1$ is $O(3n)$ using $c = 1$ and $n_0 = 1$

Sanity check

In [9]:
```
c = 1
n_0 = 1
f = lambda n: 2*n+1
g = lambda n: 3*n

plot_oh(f, g, c, n_0, '$2n+1$', str(c)+'$\cdot 3n$', 0, 50)
#plt.rcParams["figure.figsize"] = (4,4)
```



Basic Questions

Q3. Prove that 4 is $O(2)$

Solution

To recall the definition of Big-Oh: Given positive functions $f(n)$ and $g(n)$, we can say that $f(n)$ is $O(g(n))$ if and only if there exists positive constants c and n_0 such that $f(n) \leq c \cdot g(n), \forall n \geq n_0$.

To prove that 4 is $O(2)$, we need to show that there exists positive constants c and n_0 such that

$$4 \leq c \cdot 2, \forall n \geq n_0$$

Again this one is trivial, pick any $c \geq 2$ and any value for n_0. Note however that when doing proofs, we should pick sensible values for c and n_0. Yes, $n_0 = 25897257025724980576 4$ would work but is ***bad*** style.

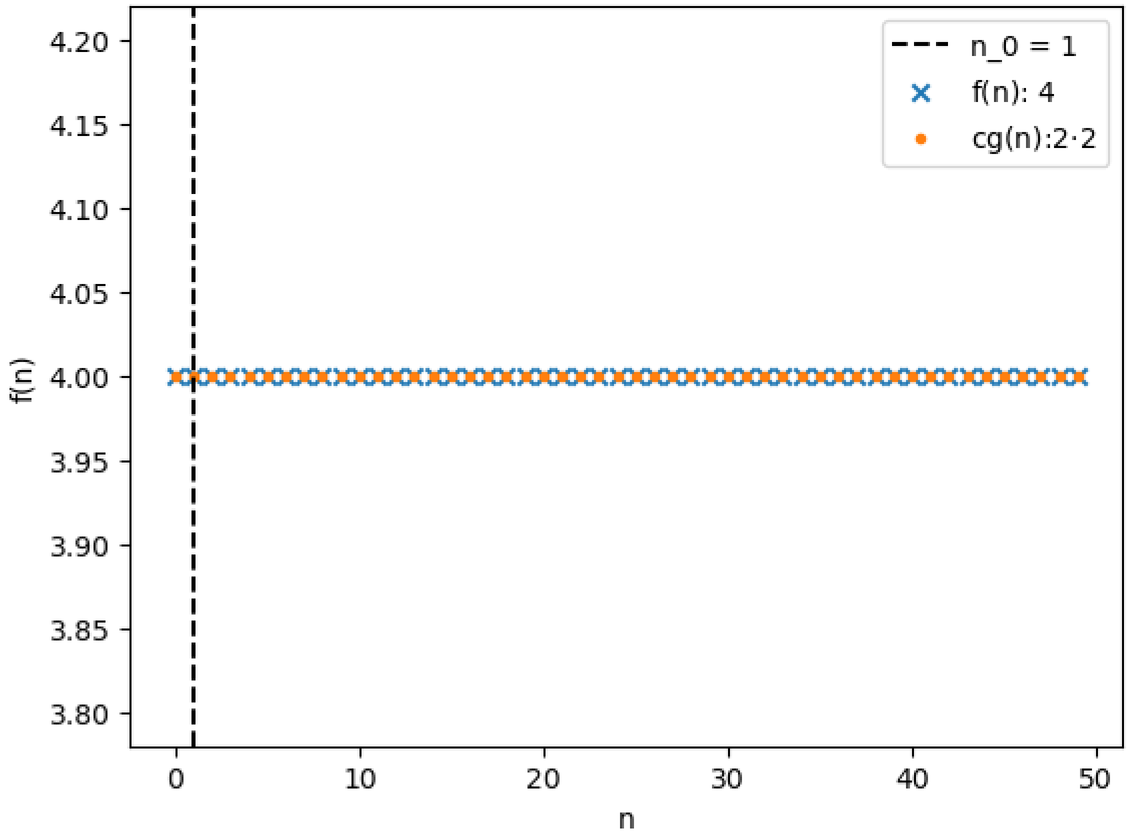
Choosing $c = 2$ and $n_0 = 1$, we need to show that:

- $4 \leq 2 \cdot 2, \forall n \geq 1$
- $4 \leq 4, \forall n \geq 1$ which is trivially true.

$\therefore 4$ is $O(2)$ using $c = 2$ and $n_0 = 1$.

Sanity check

In [4]:
```
c = 2
n_0 = 1
f = lambda n: 0*n+4
g = lambda n: 0*n+2

plot_oh(f, g, c, n_0, '$4$', str(c)+'$\cdot 2$', 0, 50)
#plt.rcParams["figure.figsize"] = (4,4)
```



Q4. Prove that $2n + 1$ is $O(n)$

Solution(s)

To recall the definition of Big-Oh: Given positive functions $f(n)$ and $g(n)$, we can say that $f(n)$ is $O(g(n))$ if and only if there exists positive constants c and n_0 such that $f(n) \leq c \cdot g(n), \forall n \geq n_0$.

We can try the same as we did before, but we find that we are using a trial-and-error process to find a value of c that works. Another approach is to attempt to "derive" the value for c.

$2n + 1 \text{ is } O(n) \text{ if and only if there exists positive constants } c \text{ and } n_0 \text{ such that } 2n + 1 \leq$

We can first try and simplify the inequality by subtracting $2n$ to give:

$$1 \leq cn - 2n, \forall n \geq n_0$$

which further simplifies to:

$$1 \leq (c - 2)n, \forall n \geq n_0$$

It is more obvious to us now what value of c should be chosen as this would never work for any value of $c \leq 2$. We *could* choose $c = 2.01$ but then we need to work out the value for n_0. We can avoid complicating things by choosing $c = 3$.

- choosing $c = 3$ gives $1 \leq (3 - 2)n, \forall n \geq n_0$
- which simplifies to $1 \leq n, \forall n \geq n_0$
- From this we can see that $n_0 \geq 1$
- which gives $1 \leq n, \forall n \geq 1$ which is trivially true

$\therefore 2n + 1$ is $O(n)$ using $c = 3$ and $n_0 = 1$.

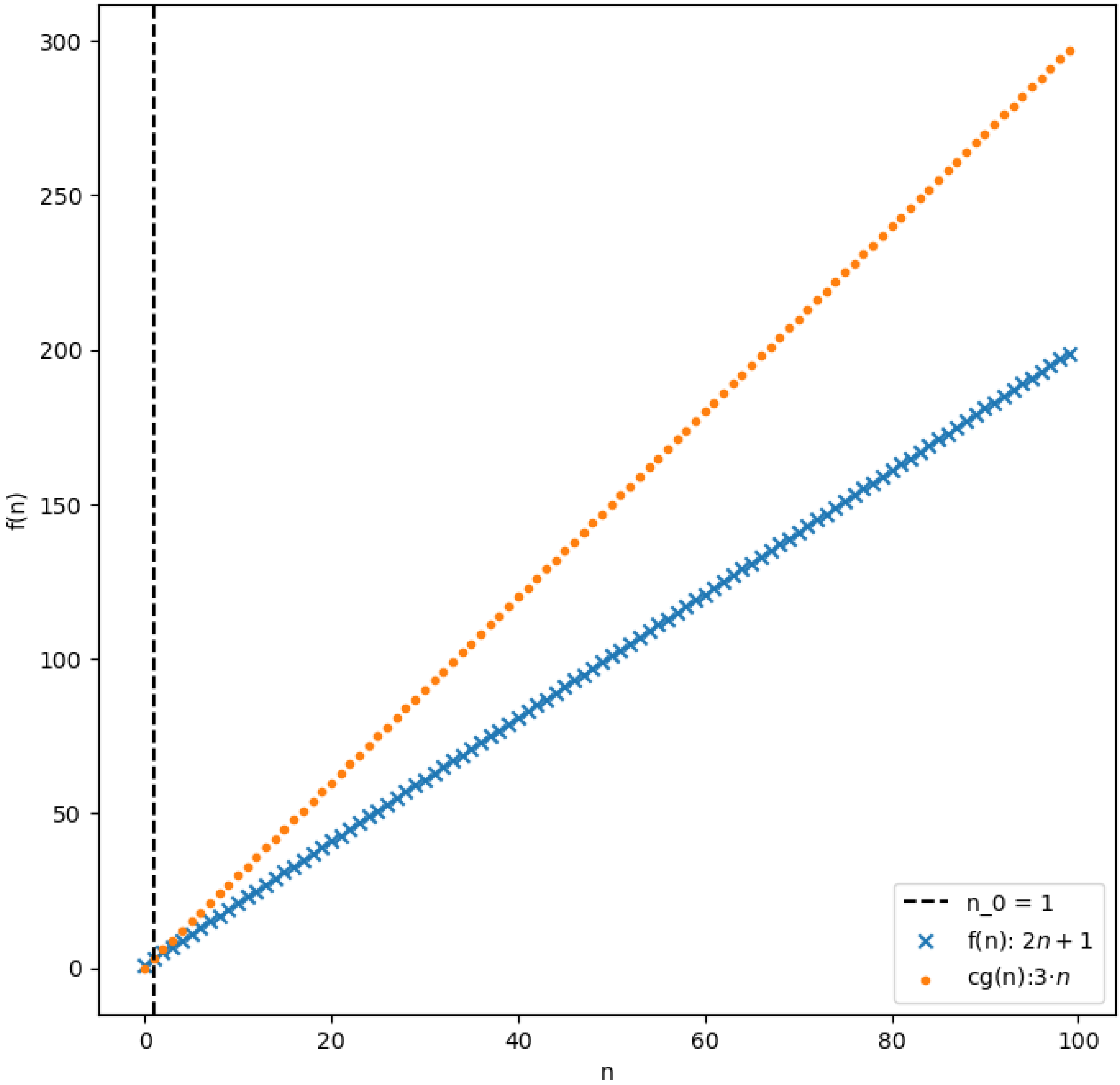
We are allowed to use fractional values of c. Below is an example of the solution using $c = 2.01$.

- choosing $c = 2.01$ gives $1 \leq (2.01 - 2)n, \forall n \geq n_0$
- which simplifies to $1 \leq 0.01n, \forall n \geq n_0$
- multiplying by 100 to remove fractional n gives $100 \leq n, \forall n \geq n_0$
- From this we can see that $n_0 \geq 100$
- which gives $100 \leq n, \forall n \geq 100$ which is trivially true

$\therefore 2n + 1$ is $O(n)$ using $c = 2.01$ and $n_0 = 100$.

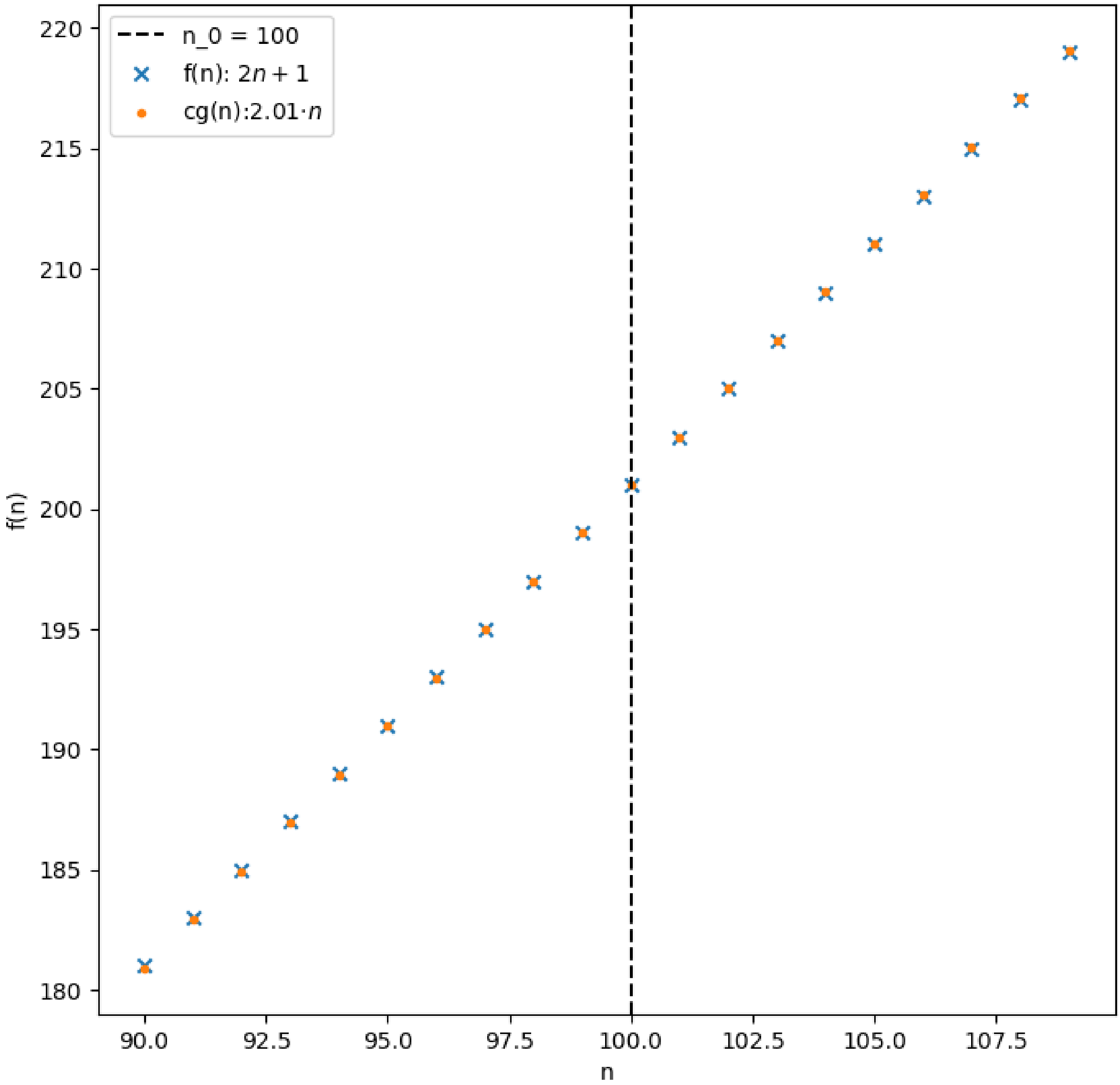
Sanity check using $c = 3$

In [7]:
```
c = 3
n_0 = 1
f = lambda n: 2*n+1
g = lambda n: n

plot_oh(f, g, c, n_0, '$2n+1$', str(c)+'$\cdot n$', 0,100)
```



Sanity check using $c = 2.01$

In [6]:
```
c = 2.01
n_0 = 100
f = lambda n: 2*n+1
g = lambda n: n

plot_oh(f, g, c, n_0, '$2n+1$', str(c)+'$\cdot n$', 90, 110)
plt.rcParams["figure.figsize"] = (8,8)
```



Medium Difficulty Questions

To recall the definition of Big-Oh: Given positive functions $f(n)$ and $g(n)$, we can say that $f(n)$ is $O(g(n))$ if and only if there exists positive constants c and n_0 such that $f(n) \leq c \cdot g(n), \forall n \geq n_0$.

Q5. Prove that n^2 is $O(2n^2)$

Solution

We can say that n^2 is $O(2n^2)$ if and only if there exists positive constants c and n_0 such that $n^2 \leq c \cdot 2n^2, \forall n \geq n_0$.

- $n^2 \leq c \cdot 2n^2, \forall n \geq n_0$
- $1 \leq c \cdot 2, \forall n \geq n_0$ (divide by n^2)
- $0.5 \leq c, \forall n \geq n_0$ (divide by 2)

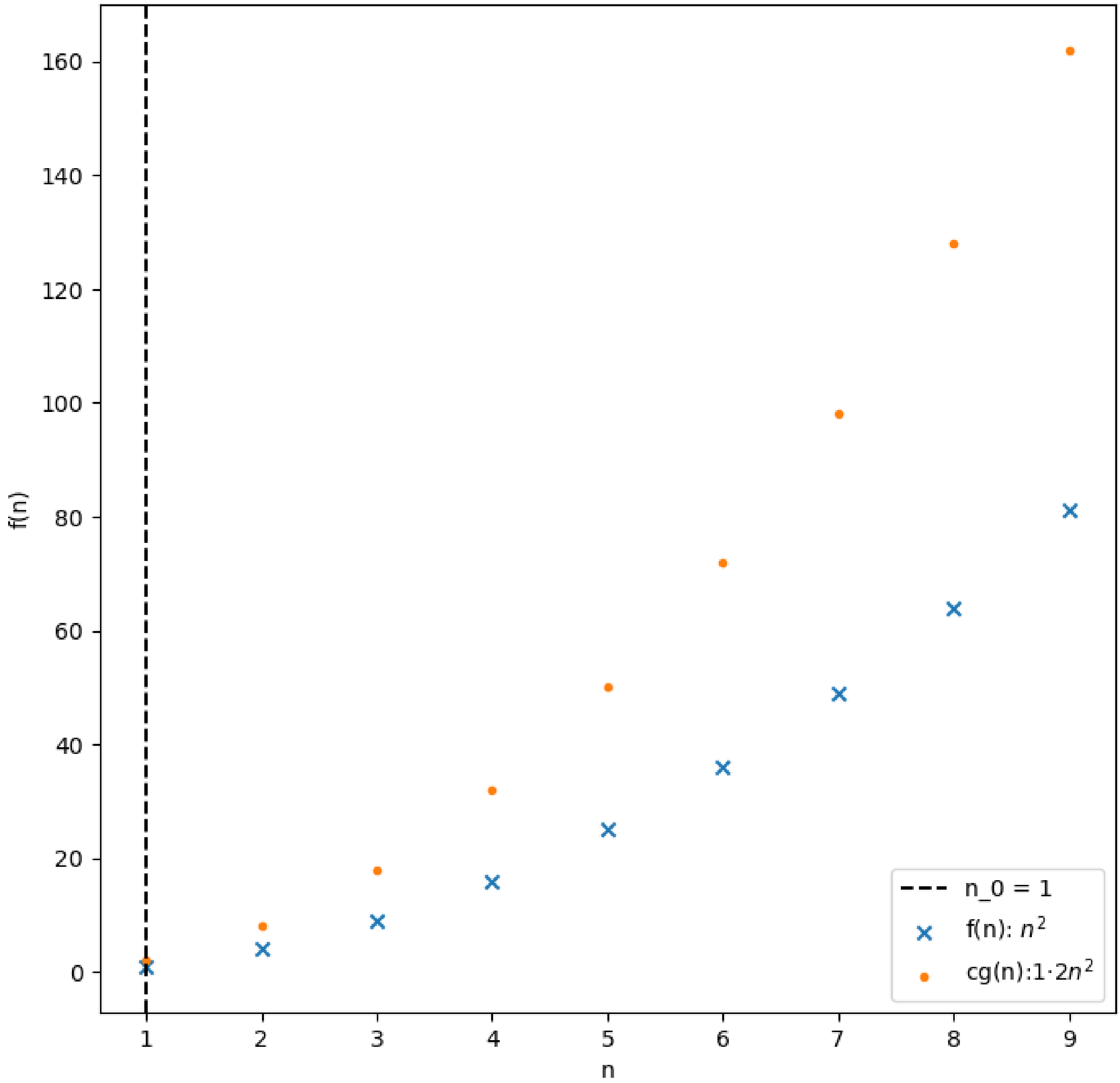
Here we can choose any c not less than 0.5. Here we will choose $c = 1$ but suggest you try the proof on your own with $c = 0.5$; what differs (if anything).

- $0.5 \leq 1, \forall n \geq n_0$ (choose $c = 1$)
- $0.5 \leq 1, \forall n \geq 1$ (choose $n_0 = 1$)

$\therefore n^2$ is $O(2n^2)$ using $c = 1$ and $n_0 = 1$.

Sanity check

In [8]:
```
c = 1
n_0 = 1
f = lambda n: (n**2)
g = lambda n: (2*(n**2))

plot_oh(f, g, c, n_0, '$n^2$', str(c)+'$\cdot 2n^2$', 1, 10)
#plt.rcParams["figure.figsize"] = (4,4)
```



Q6. Prove that $n^2 - 3$ is $O(n^2)$

Solution

We can say that $n^2 - 3$ is $O(n^2)$ if and only if there exists positive constants c and n_0 such that $n^2 - 3 \leq c \cdot n^2, \forall n \geq n_0$.

- $n^2 - 3 \leq c \cdot n^2, \forall n \geq n_0$

- $n^2 - c \cdot n^2 \leq 3, \forall n \geq n_0$
- $n^2(1-c) \leq 3, \forall n \geq n_0$

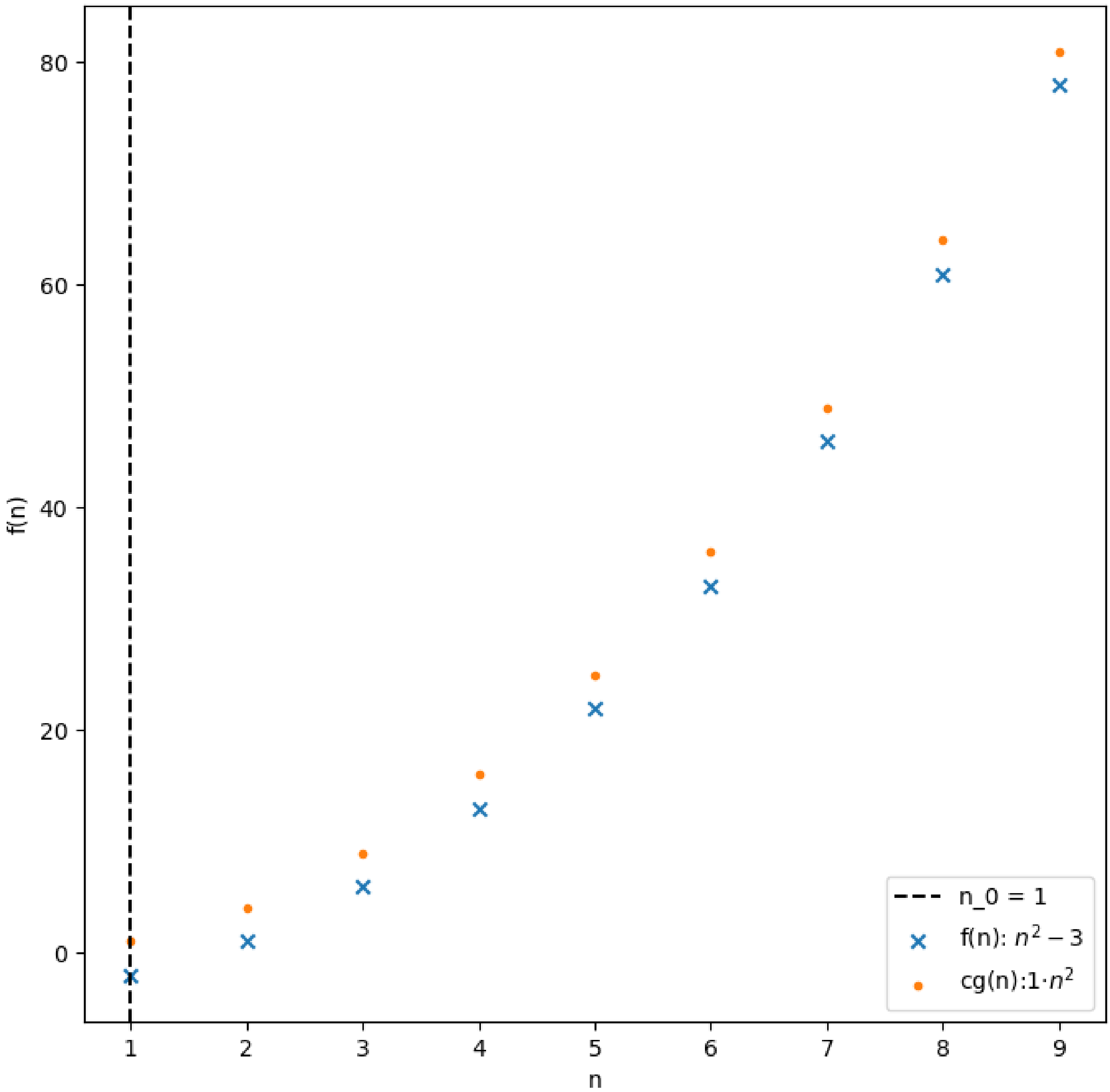
Need n^2 term to be zero (or negative) to satisfy the inequality.

- $n^2(1-1) \leq 3, \forall n \geq n_0$ (choose $c = 1$)
- $0 \leq 3, \forall n \geq n_0$
- $0 \leq 3, \forall n \geq 1$ (choose $n_0 = 1$)

$\therefore n^2 - 3$ is $O(n^2)$ using $c = 1$ and $n_0 = 1$

Sanity check

In [11]:
```
c = 1
n_0 = 1
f = lambda n: (n**2 - 3)
g = lambda n: (n**2)

plot_oh(f, g, c, n_0, '$n^2-3$', str(c)+'$\cdot n^2$', 1, 10)
# plt.rcParams["figure.figsize"] = (8,8)
```



Q7. Prove that $n^2 - 5n$ is $O(n^2)$

Solution

We can say that $n^2 - 5n$ is $O(n^2)$ if and only if there exists positive constants c and n_0 such that $n^2 - 5n \leq c \cdot n^2, \forall n \geq n_0$.

- $n^2 - 5n \leq c \cdot n^2, \forall n \geq n_0$
- $0 \leq c \cdot n^2 - n^2 + 5n, \forall n \geq n_0$ (rearranging)
- $0 \leq n(c \cdot n - n + 5), \forall n \geq n_0$ (factorising)
- $0 \leq n(n(c-1) + 5), \forall n \geq n_0$ (factorising)
- $0 \leq 5n, \forall n \geq n_0$ (choose $c = 1$)

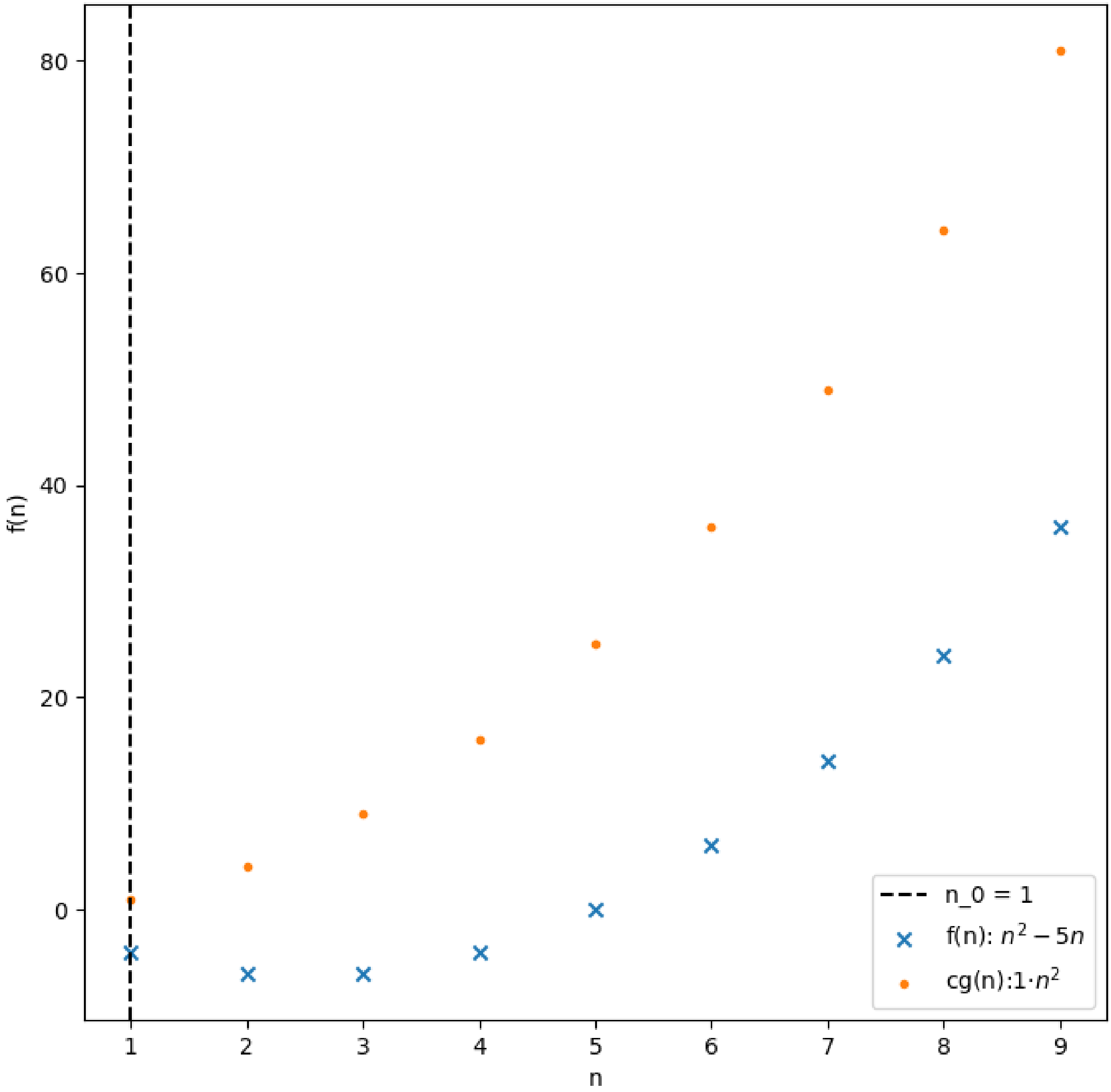
Which is trivially true for $n \geq 0.2$; hence, $n_0 = 1$.

- $0 \leq 5, \forall n \geq 1$ (choose $n_0 = 1$)

$\therefore n^2 - 5n$ is $O(n^2)$ using $c = 1$ and $n_0 = 1$

Sanity check

In [12]:
```
c = 1
n_0 = 1
f = lambda n: (n**2 - 5*n)
g = lambda n: (n**2)

plot_oh(f, g, c, n_0, '$n^2-5n$', str(c) + '$ \cdot n^2$', 1, 10)
#plt.rcParams["figure.figsize"] = (8,8)
```



Q8. Prove that $n^2 + 1$ is $O(n^2)$

Solution

We can say that $n^2 + 1$ is $O(n^2)$ if and only if there exists positive constants c and n_0 such that $n^2 + 1 \leq c \cdot n^2, \forall n \geq n_0$.

- $n^2 + 1 \leq c \cdot n^2, \forall n \geq n_0$
- $1 \leq (c - 1) \cdot n^2, \forall n \geq n_0$ (subtract n^2)

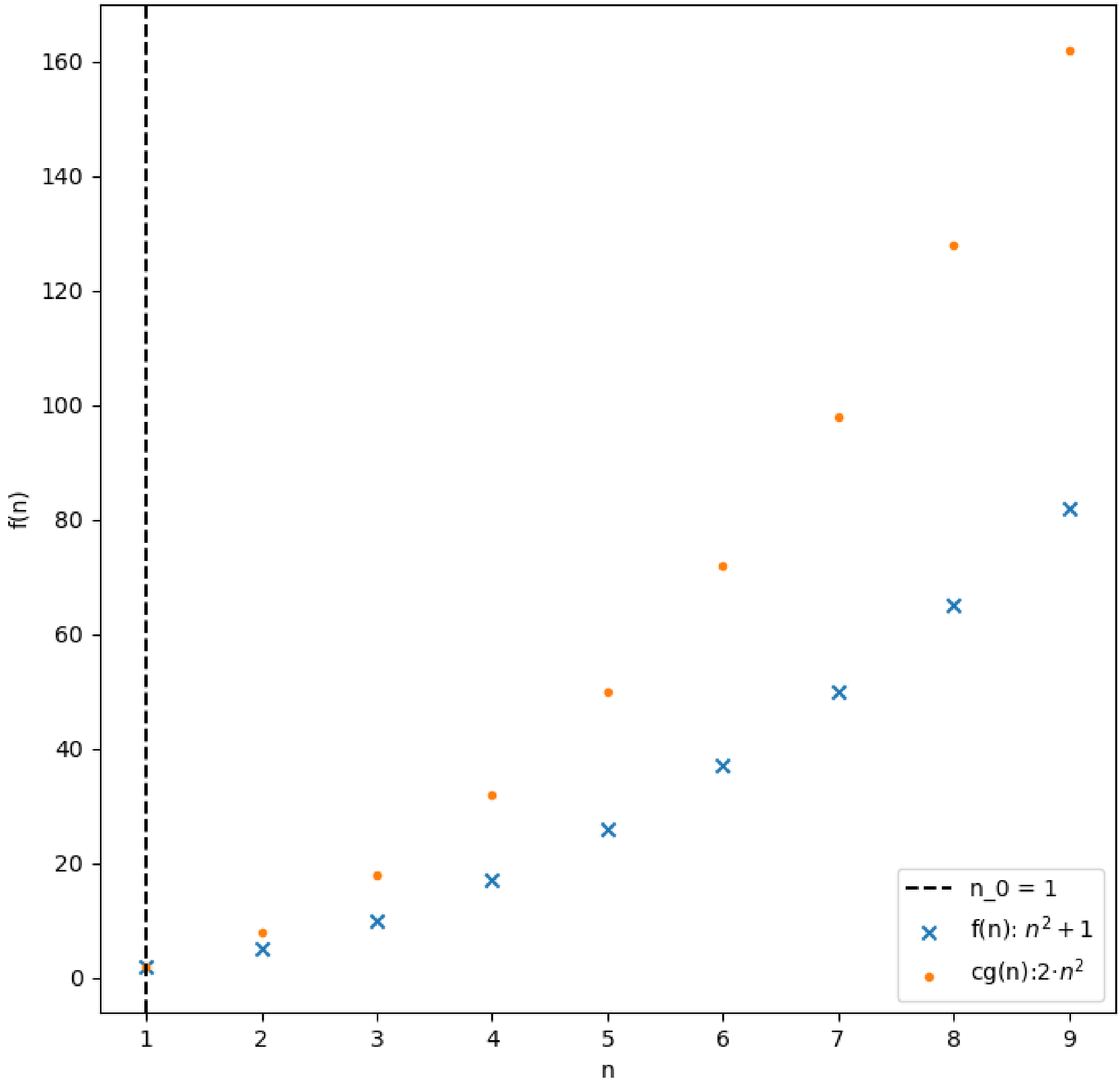
c must be greater than 1 for the inequality to be true!

- $1 \leq (2 - 1) \cdot n^2, \forall n \geq n_0$ (choose $c = 2$)
- $1 \leq n^2, \forall n \geq n_0$ (simplification)
- $1 \leq n^2, \forall n \geq 1$ (pick $n_0 = 1$)

$\therefore n^2 + 1$ is $O(n^2)$ using $c = 2$ and $n_0 = 1$.

Sanity check

In [14]:
```
c = 2
n_0 = 1
f = lambda n: (n**2 + 1)
g = lambda n: (n**2)

plot_oh(f, g, c, n_0, '$n^2+1$', str(c)+'$\cdot n^2$', 1, 10)
#plt.rcParams["figure.figsize"] = (8,8)
```



Conceptually Challenging Questions (Answers)

To recall the definition of Big-Oh: Given positive functions $f(n)$ and $g(n)$, we can say that $f(n)$ is $O(g(n))$ if and only if there exists positive constants c and n_0 such that $f(n) \leq c \cdot g(n), \forall n \geq n_0$.

Q9. From the definitions, prove or disprove that 1 is $O(n)$

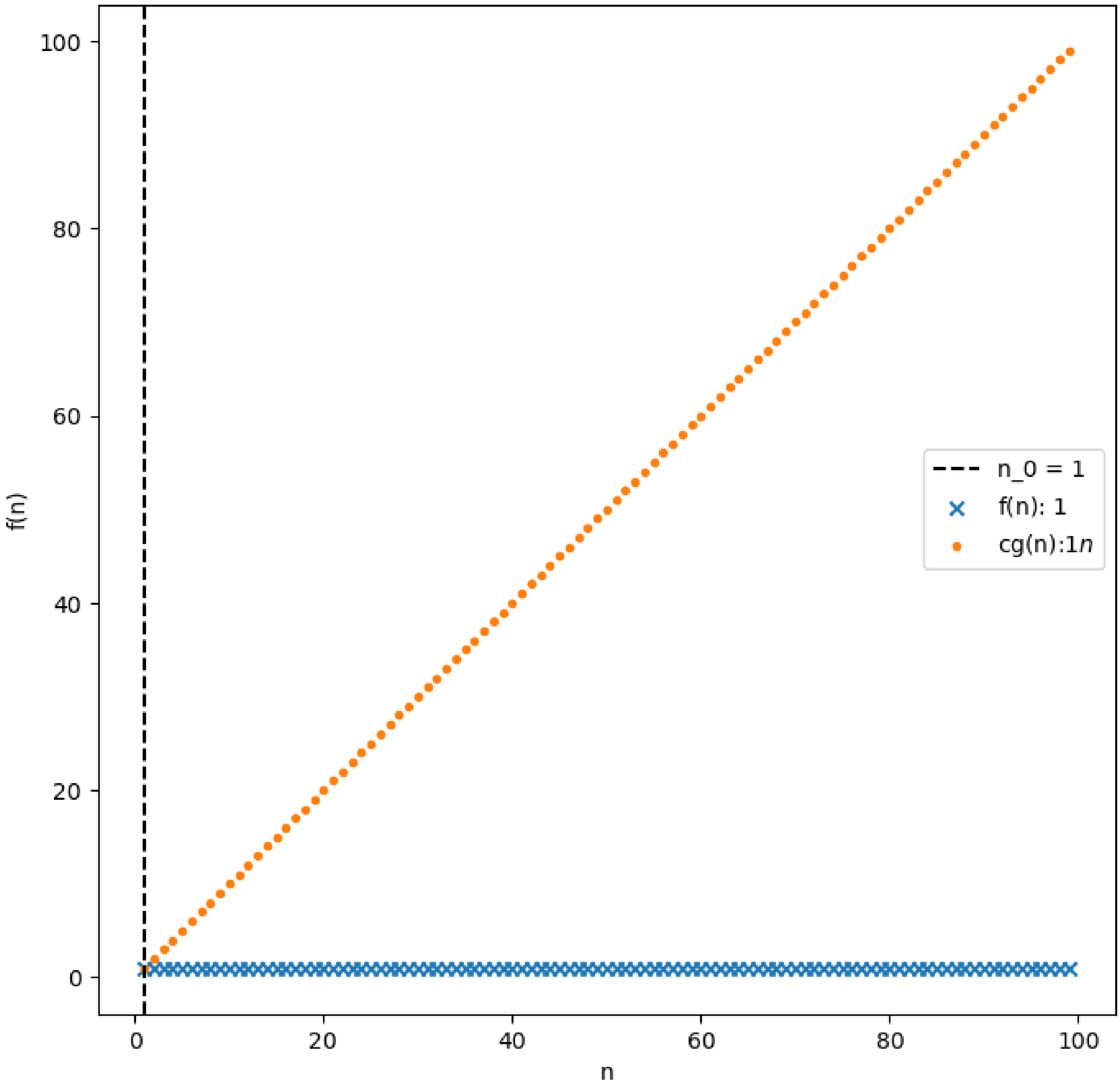
Solution

Provably true:

- $1 \leq c \cdot n, \forall n \geq n_0$
- $1 \leq n, \forall n \geq n_0$ (choose $c = 1$)
- $1 \leq n, \forall n \geq 1$ (pick $n_0 = 1$)
- $\therefore 1$ is $O(n)$

Sanity check

In [15]:
```
c = 1
n_0 = 1
f = lambda n: (n*0+1)
g = lambda n: (n)

plot_oh(f, g, c, n_0, '$1$', str(c)+'$n$', 1, 100)
```



Q10. From the definitions, prove or disprove that n is $O(1)$

Solution

From the definition, $\exists n_0 \geq 0, \exists c \geq 0$ such that $n \leq c \cdot 1, \forall n \geq n_0$ and c and n_0 are constants.

For this inequality to hold true, c would need to depend on n_0 however from the definition both are constants hence we can always choose a value of $n = c + 1$ which invalidates the inequality. That is, $\forall c \geq 0, \forall n_0 \geq 0, \exists n \geq n_0$ such that $n \nleq c \cdot 1$ whereby $n = c + 1$.

Q11. From the definitions, prove or disprove that n^2 is $O(n)$

Solution

Similarly with Q10 this can be disproved.

Would need to prove that $n^2 \leq c \cdot n, \forall n \geq n_0$

Simplifying the inequality by dividing my n gives us $n \leq c \cdot 1, \forall n \geq n_0$

Proof: `"goto Q10"`

Q12. Given that $f(n) = \text{ IF } even(n) \text{ THEN } n + 3 \text{ ELSE } n^2 + 5$ state the Big-Oh Behaviour and prove it from the definition

Any natural number "n" is either even or not even. If the number is even, then we get $f_{even}(n) = n + 3$, otherwise we get $f_{odd}(n) = n^2 + 5$.

$f_{even} = O(n)$ whereas $f_{odd} = O(n^2)$. But as a single function f, $f(n) = O(n^2)$.

We can prove that $f(n)$ is $O(n^2)$ by choosing fixed values of c and n_0 for both even and odd cases; that is, c and n_0 should be the same for both the odd case and the even case.

Solution

Even case	Odd case	
$n + 3 \leq c \cdot n^2$	$n^2 + 5 \leq c \cdot n^2$	$\forall n \geq n_0$
$3 \leq n(cn - 1)$	$5 \leq n^2(c - 1)$	$\forall n \geq n_0$

Need $c > 1$, choose $c = 2$.

$3 \leq 2n^2 - n$	$5 \leq n^2$	$\forall n \geq n_0$
$n \geq 1.5$	$n \geq \sqrt{5}$	$\forall n \geq n_0$

Choose ceiling of 1.5 and $\sqrt{5}$.

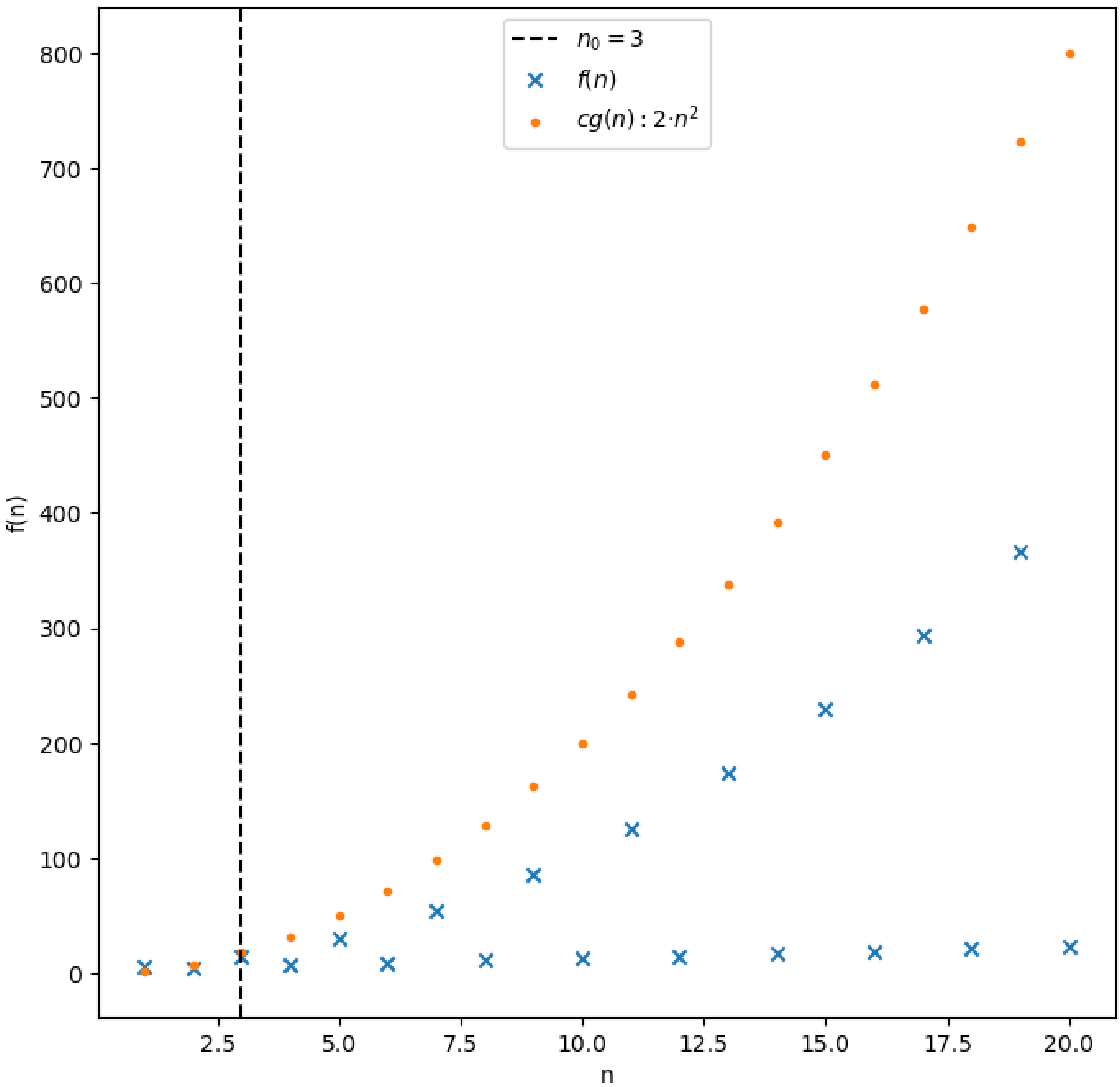
$n \geq 1.5$	$n \geq \sqrt{5}$	$\forall n \geq 3$

$\therefore f(n)$ is $O(n^2)$ using $c = 2$ and $n_0 = 3$.

Sanity check

In [43]:
```
c = 2
n_0 = 3

# max value of n in plot
max_n = 20

# f(n)
xs = np.arange(1,max_n+1,1)
fns = np.ones(max_n)
for n in range(1,max_n+1):
    fns[n-1] = ((n + 3) if (n % 2 == 0) else (n**2 + 5))

# g(n)
gns = (np.arange(1,max_n+1,1)**2)*c

# plot
plt.axvline(x = n_0, color = 'k', linestyle='--')
plt.scatter(x=xs, y=fns, marker='x')
plt.scatter(x=xs, y=gns, marker='.')
plt.xlabel('n')
plt.ylabel('f(n)')
plt.legend(['$n_0 = $'+str(n_0),'$f(n)$', '$cg(n):$' + str(c) + '$\cdot n^2$'])
```

Out[43]:
```
<matplotlib.legend.Legend at 0x1d6a8d6a070>
```



Algebraically Challenging (Answers)

Work out the Big-Oh of the following functions and prove them using the definitions.

Q13. $3n^3 + 10000n$

Solution

(Big-Oh Derivation using Rules)

- $3n^3 + 10000n$
- $n^3 \cdot \left(3 + \frac{10000}{n^2}\right)$
- $\frac{10000}{n^2} \to 0$ as $n \to \infty$
- $\therefore f(n) = O(n^3)$

(Proof)

To recall the definition of Big-Oh: Given positive functions $f(n)$ and $g(n)$, we can say that $f(n)$ is $O(g(n))$ if and only if there exists positive constants c and n_0 such that $f(n) \leq c \cdot g(n), \forall n \geq n_0$.

$3n^3 + 10000n$ is $O(n^3)$ if and only if there exists positive constants c and n_0 such that $3n^3 + 10000n \leq c \cdot n^3, \forall n \geq n_0$

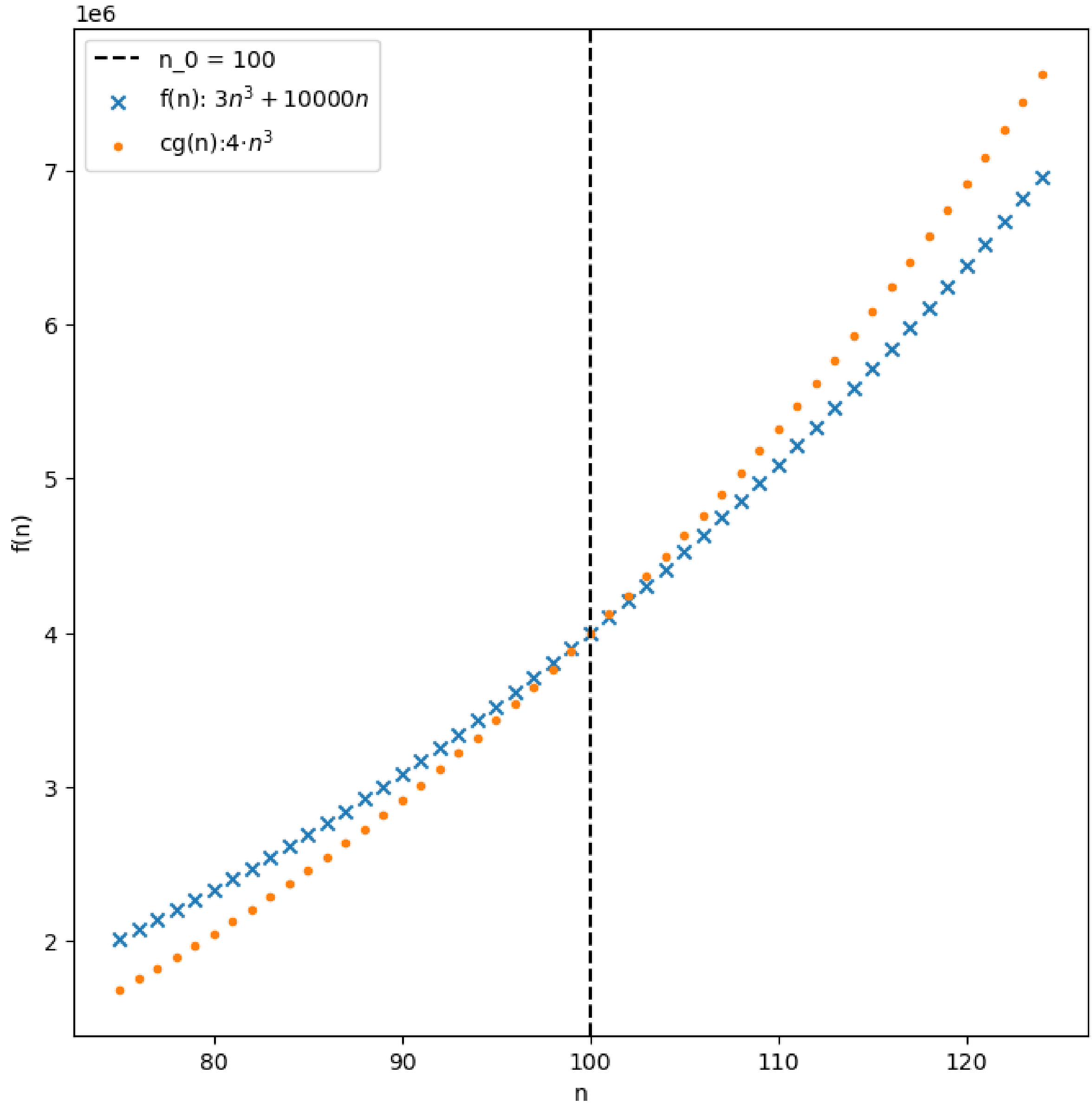
- $3n^3 + 10000n \leq c \cdot n^3, \forall n \geq n_0$
- $3n^3 + 10000n \leq 4n^3, \forall n \geq n_0$ (choose $c = 4$)
- $10000n \leq n^3, \forall n \geq n_0$ (subtract $3n^3$)
- $10000 \leq n^2, \forall n \geq n_0$ (divide by n)
- $100 \leq n, \forall n \geq n_0$ ($\sqrt{.}$)
- Hence, $n_0 \geq 100$ so pick $n_0 = 100$
- $100 \leq n, \forall n \geq 100$ -- trivial

$\therefore 3n^3 + 10000n$ is $O(n^3)$ for $c = 4$ and $n_0 = 100$.

Sanity check

In [46]:

```
c = 4
n_0 = 100
f = lambda n: 3*n**3+10000*n
g = lambda n: n**3

plot_oh(f, g, c, n_0, '$3n^3+10000n$', str(c) + '$\cdot n^3$', 75, 125)
```



Q14. $n\ log(n) + 2n$

Solution

(Big-Oh Derivation using Rules)

- $n\ log(n) + 2n$
- $n\ (log(n) + 2)$ (factor out n)
- $n\ (log\ n)$ (drop smaller terms)
- $\therefore n\ log(n) + 2n = O(n\ log(n))$

(Proof)

To recall the definition of Big-Oh: Given positive functions $f(n)$ and $g(n)$, we can say that $f(n)$ is $O(g(n))$ if and only if there exists positive constants c and n_0 such that $f(n) \leq c \cdot g(n), \forall n \geq n_0$.

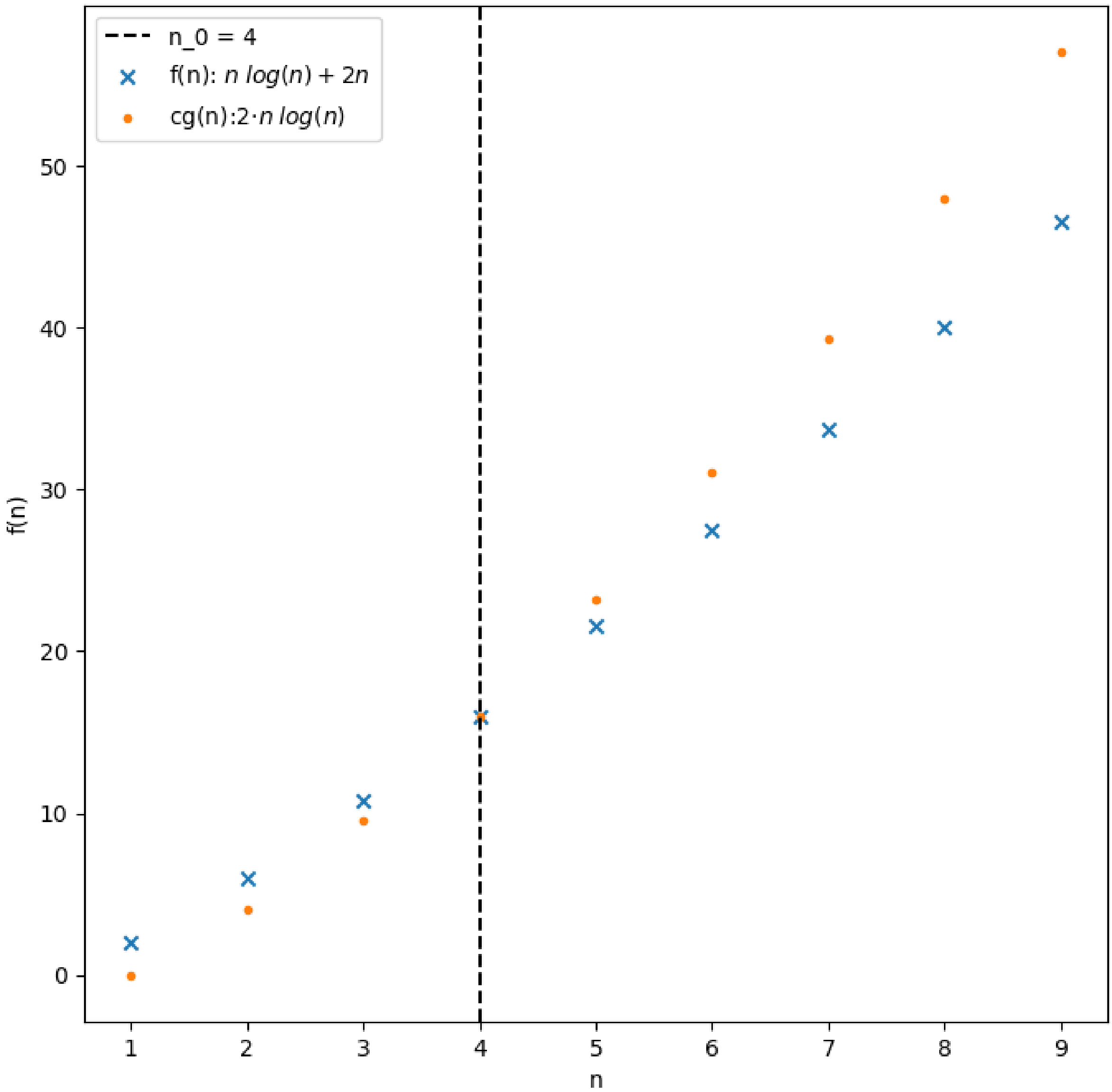
$n\ log(n) + 2n$ is $O(n\ log(n))$ if and only if there exists positive constants c and n_0 such that $n\ log(n) + 2n \leq c \cdot n\ log(n), \forall n \geq n_0$

- $n\ log(n) + 2n \leq c \cdot n\ log(n), \forall n \geq n_0$
- $2n \leq (c-1) \cdot n\ log(n), \forall n \geq n_0$
- $2n \leq n\ log(n), \forall n \geq n_0$ (choose $c = 2$)
- $2 \leq log(n)$
- $2^2 \leq 2^{log_2(n)}$
- $4 \leq n$ (using log base 2)

$\therefore n\ log(n) + 2n = O(n\ log(n))$ for $c = 2$ and $n_0 = 4$

Sanity check

In [47]:
```
c = 2
n_0 = 4
f = lambda n: (n*np.log2(n)+2*n)
g = lambda n: (n*np.log2(n))

plot_oh(f, g, c, n_0, '$n\;log(n)+2n$', str(c) + '$\cdot n\;log(n)$', 1, 10)
```



Q15. $2^n + n$

Solution

Exponents beat powers (from the rules) so is just:

$= O(2^n)$

$2^n + n \leq c \cdot 2^n, \forall n \geq n_0$

$n \leq 2^n(c-1), \forall n \geq n_0$ (choose $c = 2$)

$n \leq 2^n, \forall n \geq n_0$ which is true for all integers.

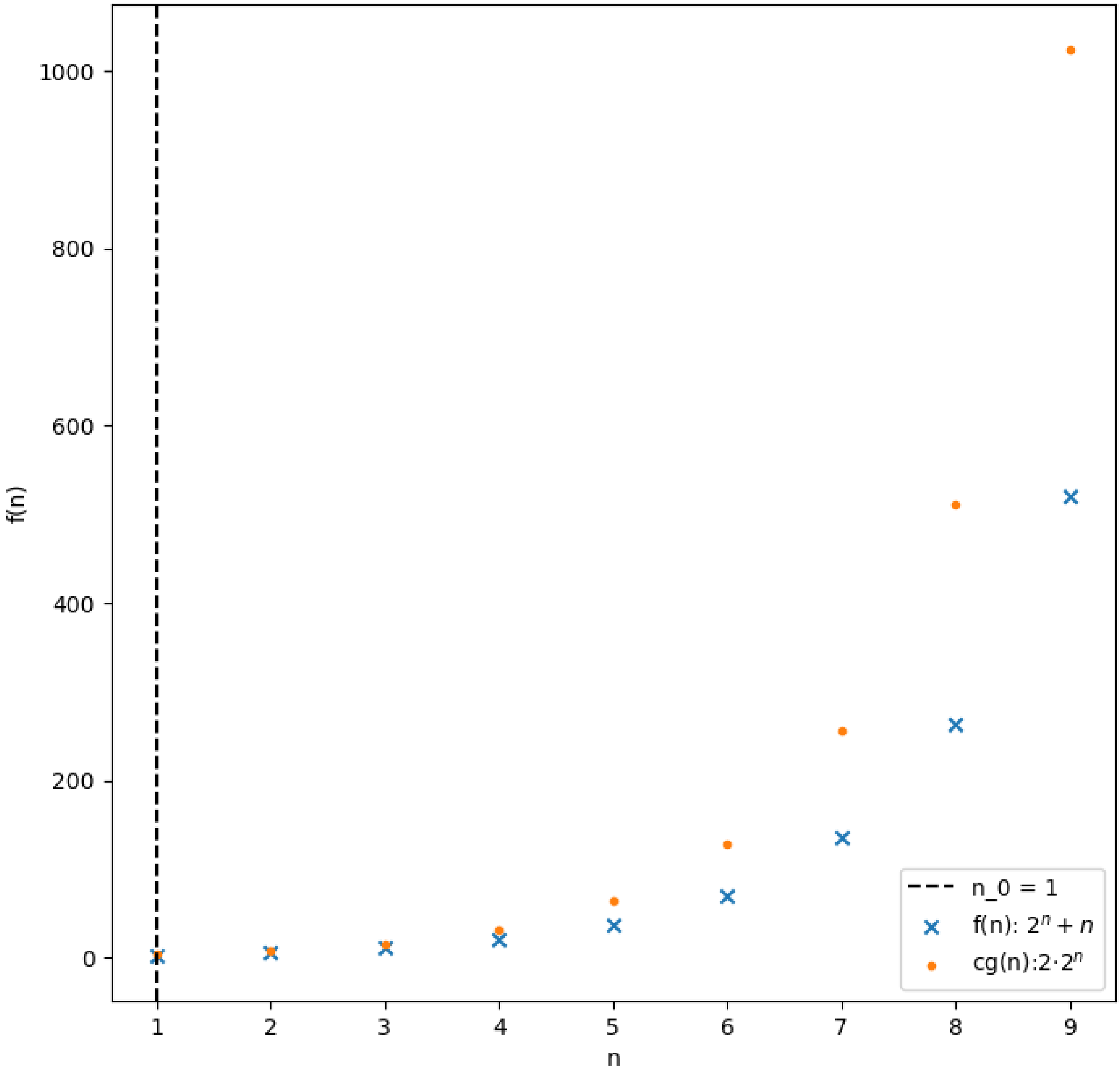
$n \leq 2^n, \forall n \geq 1$

$\therefore 2^n + n$ is $O(2^n)$ for $c = 2$ and $n_0 = 1$.

Sanity check

In [48]:

```
c = 2
n_0 = 1
f = lambda n: (2**n + n)
g = lambda n: (2**n)


plot_oh(f, g, c, n_0, '$2^n + n$', str(c) + '$\cdot 2^n$', 1, 10)
```

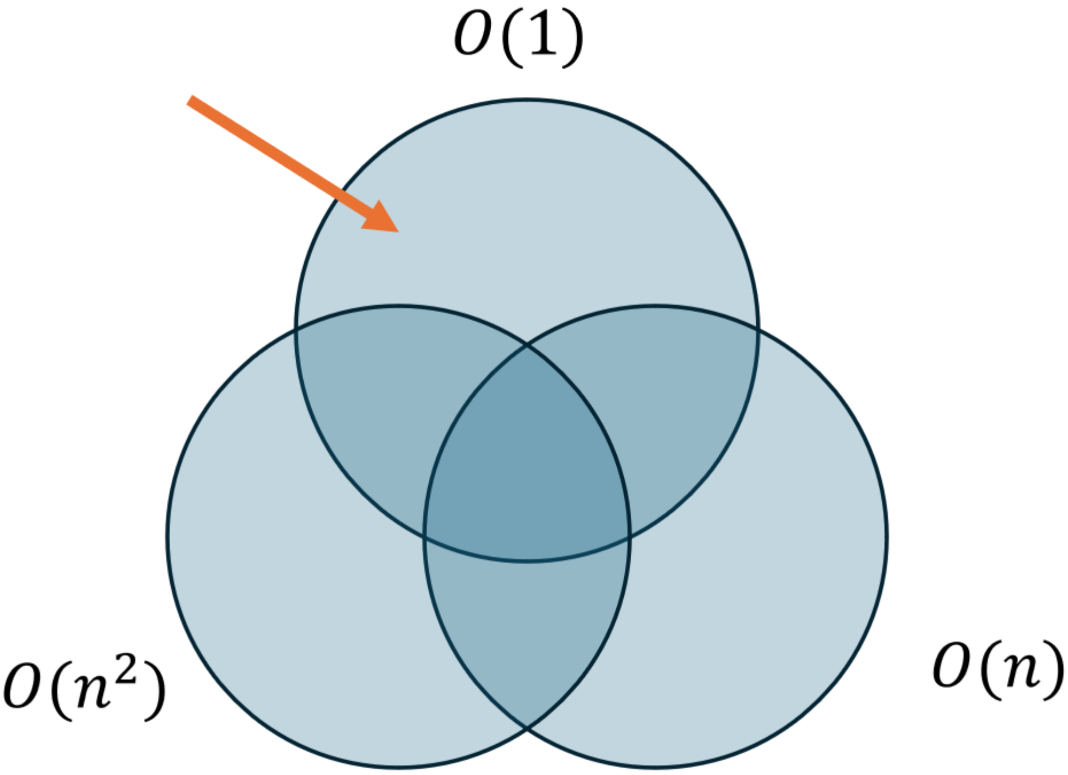


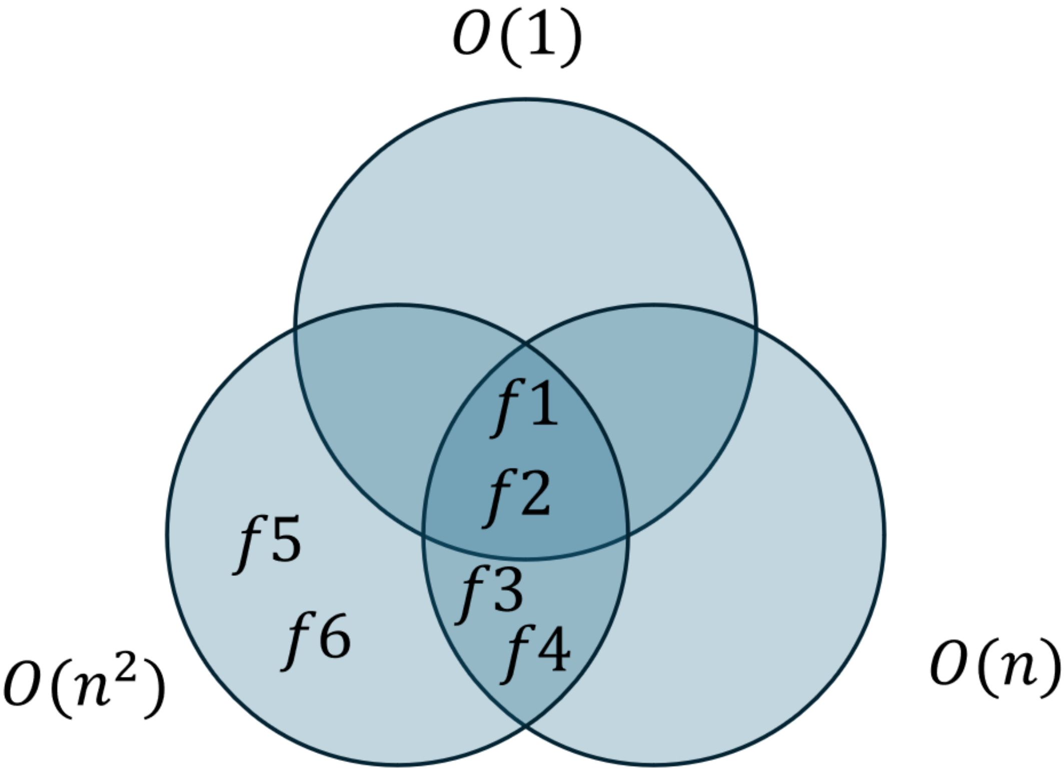
Summary: Venn Diagram

Draw a Venn Diagram of the sets $O(1)$, $O(n)$ and $O(n^2)$, and place the following functions on the diagram:

- $f1(n) = 1$
- $f2(n) = 42$
- $f3(n) = n$
- $f4(n) = 3n + 5$
- $f5(n) = n^2$
- $f6(n) = n^2 + log(n)$

Solution:


$O(1)$
$O(n^2)$
$O(n)$


$O(1)$
$f1$
$f2$
$f5$
$f3$
$f6$
$f4$
$O(n^2)$
$O(n)$