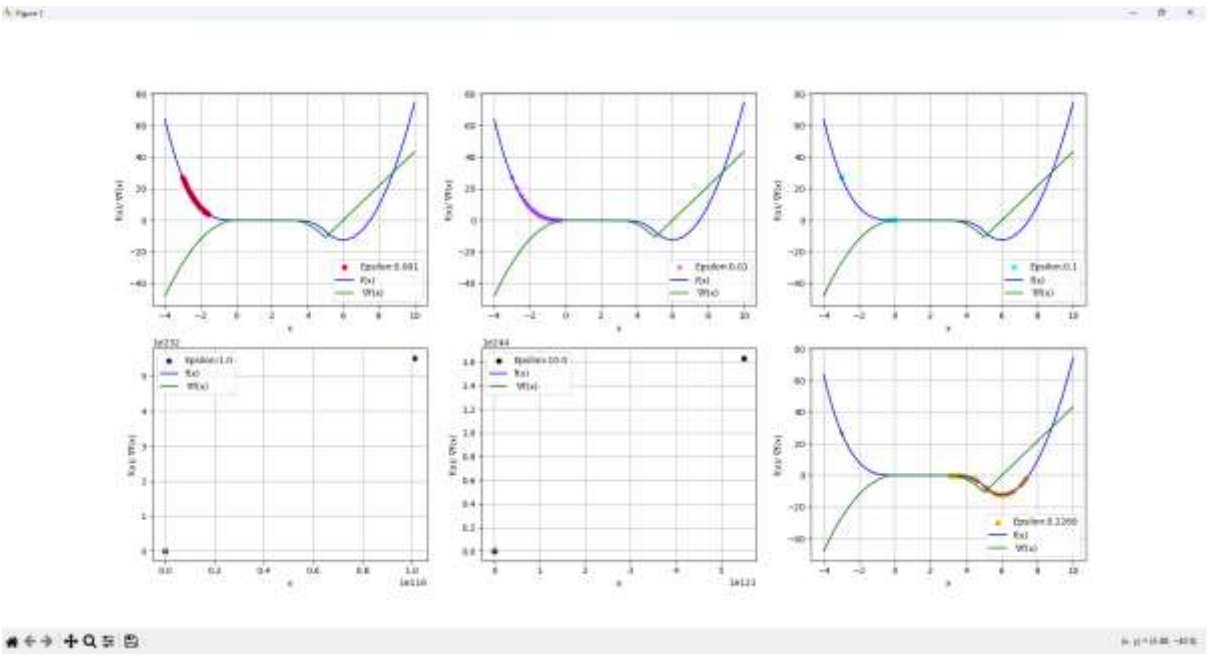
# Problem 3 (a) –

**3a i)** Please refer code in "**homework1_problem3a.py**" file

**3a ii)** Please refer code in "**homework1_problem3a.py**" file

**3a iii):**
Based on the plots for each learning rate, we observe the following: As the learning rate increases, the gradient descent attempts to converge to the global minimum. However, if the learning rate is too high, it overshoots and jumps off the curve. When the starting value of x is kept constant (e.g., x=−3), the speed of convergence varies depending on the learning rate.

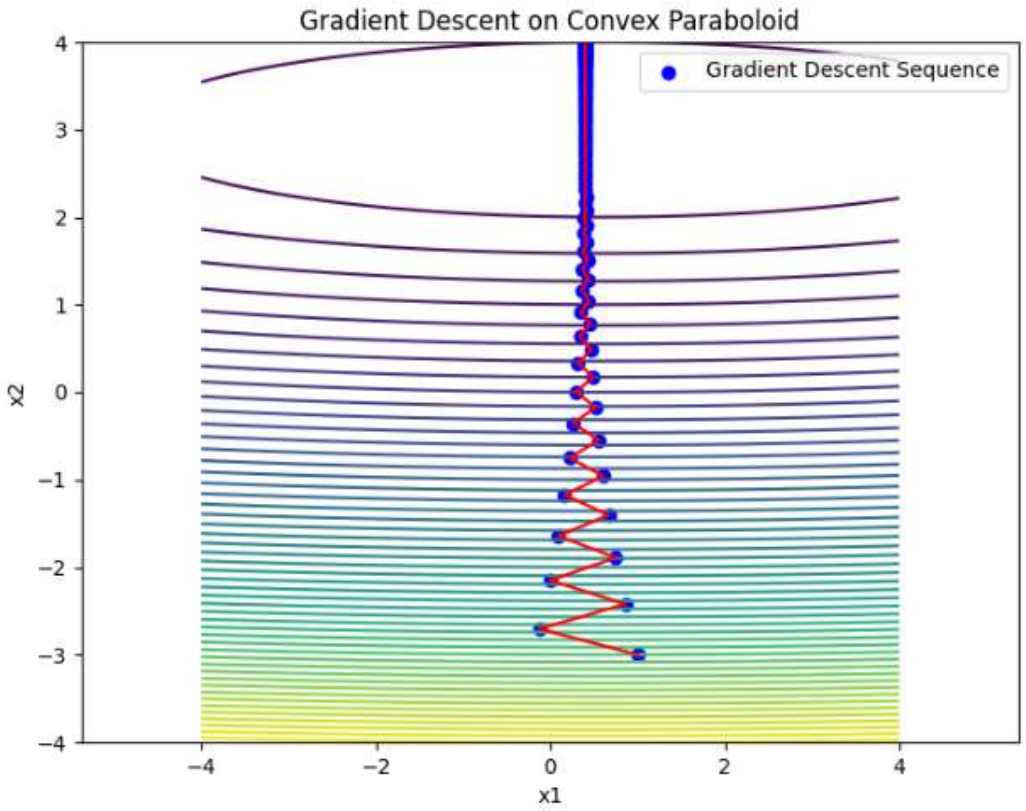
**3a iv):**
Through multiple trials and adjustments, we found that a learning rate of approximately 0.2268 causes the gradient descent to reach the minimum of f(x), but it overshoots beyond it. This demonstrates that by fine-tuning the learning rate, it is possible to converge to the minimum of f(x).



## Problem 3b-

**3b i)**: When the learning rate is drastically reduced, like making it 100 times smaller, the steps that gradient descent takes will be much tinier. This means the algorithm will move toward the minimum in a more careful and steady way, smoothing out toward the minimum, reducing the zig-zag effect. However, because the adjustments are so small, it will take much longer to reach the minimum, so the convergence will be noticeably slower.

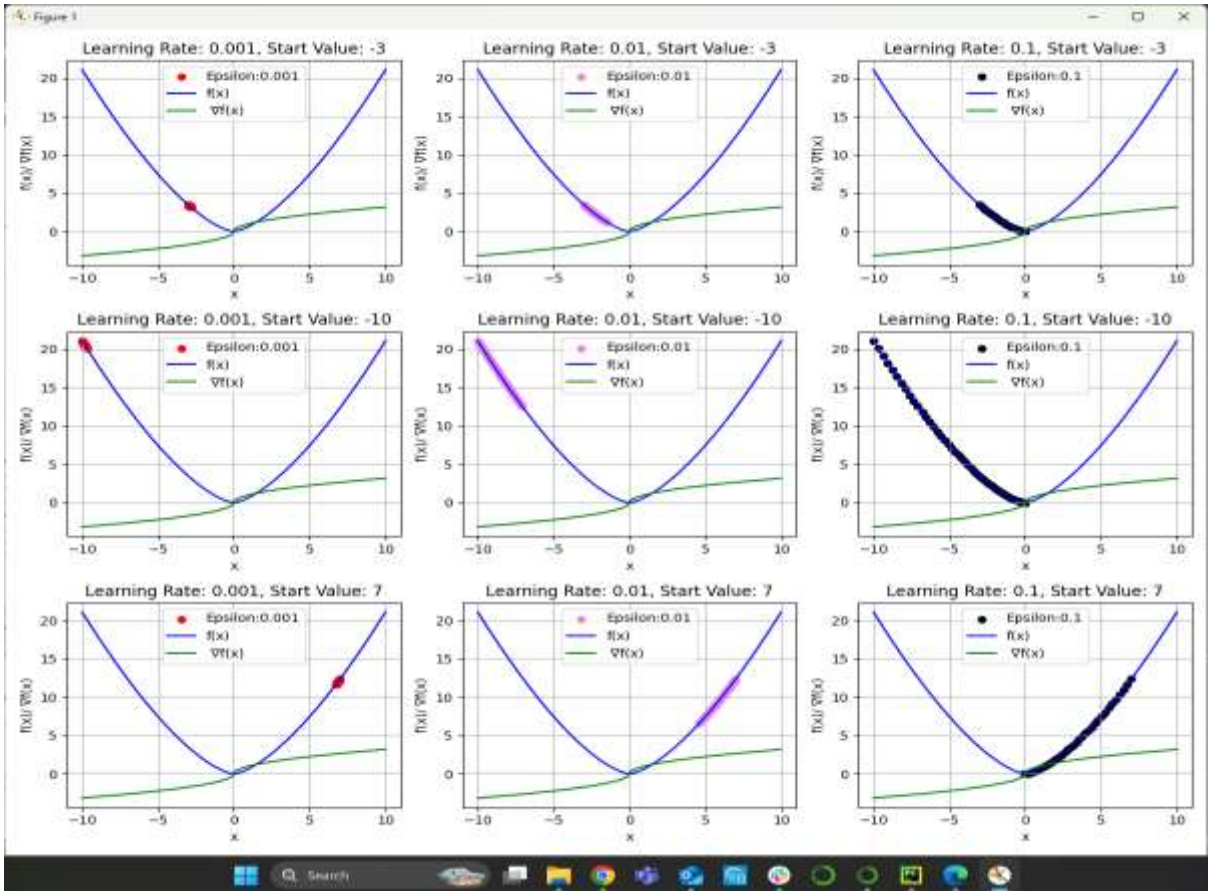
**3b ii)** Please refer code in "**homework1_problem3b.py**" file



## Problem 3c –

By examining the plots, we observe that the learning rate significantly impacts the convergence speed and stability during gradient descent. Lower learning rates lead to slower convergence, while higher rates can cause the algorithm to overshoot or even diverge from the optimal point. Additionally, while different initial values of xxx affect the descent path, the choice of the learning rate plays a more crucial role in determining whether the gradient descent will successfully reach the minimum or not. Even with extreme starting points, an appropriate learning rate can still guide the descent effectively towards the minimum.

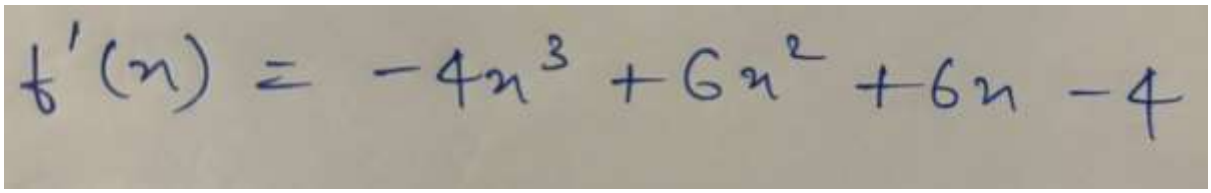
Please refer code in "**homework1_problem3c.py**" file



## Problem 3d –

Given the function $f(x) = -x\text{^}4 + 2x\text{^}3 + 3x\text{^}2 - 4x + 1$, which has local maxima at $x = -1$ and $x = 2$, we can find a scenario where gradient descent converges to one of these local maxima after just one iteration.

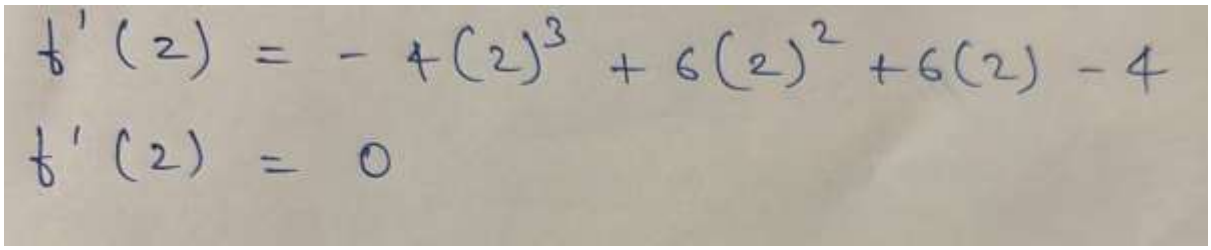
The first derivative of the function is:




We want to choose a starting point $x0$ and a learning rate $e$ such that after one gradient descent update, the new point $x1$ is exactly at a local maximum.

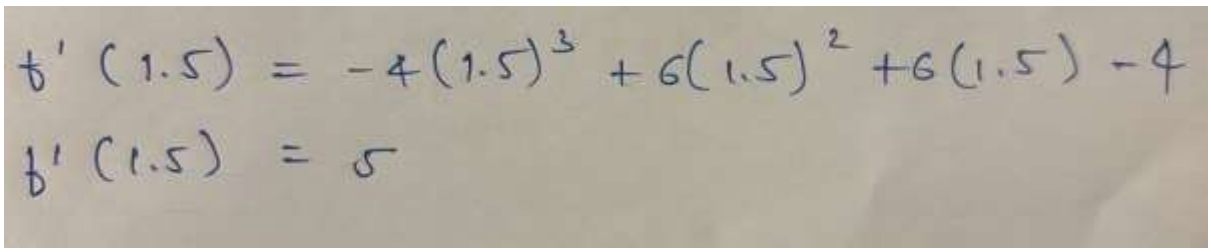
Let's aim to move towards the local maximum at $x = 2$

First, calculate the derivative at $x = 2$:

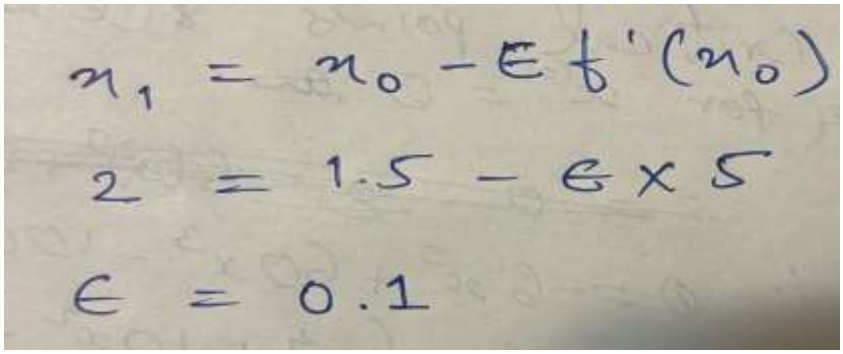



Since the gradient is zero at $x = 2$, gradient descent will stop at this point. Instead, we need to start from a different point where $f'(x) \neq 0$.

Let's choose $x0 = 1.5$ as the starting point. Calculate the derivative at $x0 = 1.5$:




Now, let's set $x1 = 2$ and solve for the e using the gradient descent update rule:



Starting at x0=1.5 with a learning rate e=0.1:

- The gradient at $x0 = 1.5$ is $f'(1.5) = 5$.
- The update step is: $x1 = 1.5 - 0.1 \times 5 = 1.5 - 0.5 = 2$.

After one iteration, the new point is $x1 = 2$, which is a local maximum, and the gradient $f'(2) = 0$, confirming that gradient descent has converged to the local maximum after one iteration.