```
from cmath import pi
from matplotlib import pyplot as plt


def sequence_value(*, number):
    return pi / 2 * (6 * number - 5) / (2 * number + 4) if number % 2 == 0 else 0


def get_graphic_of_sequence(*, sequence):
    """
    graphic of the sequence
    :param sequence: sequence
    :return: picture
    """
    plt.figure(figsize=(20, 8), dpi=80)
    plt.title("график последовательности")
    plt.xlabel("n")
    plt.ylabel("значение члена последовательности")
    for point in range(1, len(sequence)):
        if point % 2 == 0:
            plt.scatter(point, sequence[point], color='red', s=100)
        else:
            plt.scatter(point, sequence[point], color='blue', s=100)
    plt.hlines(3 * pi / 2, 0, len(sequence), colors='black')
    plt.annotate('supremum and superior limit = 3/2 * pi', xy=(0, 3 * pi / 2), xytext=(0, 3 * pi / 2 + 0.05))
    plt.hlines(0, 0, len(sequence), colors='black')
    plt.annotate('infinum and inferior limit = 0', xy=(0, 0), xytext=(0, 0.2))
    # plt.show()
    plt.savefig('original_sequence.png', bbox_inches='tight')


def get_graphic_of_subsequence(*, sequence):
```

```
    """
    k0 = [(17*pi)/(8*epsilon)]+1
    graphic of the subsequence
    :param sequence:
    :return:
    """
    epsilon = 0.0001
    plt.figure(figsize=(20, 8), dpi=80)
    plt.title("график подпоследовательности x_2k")
    plt.xlabel("n")
    plt.ylabel("значение члена подпоследовательности")
    k0 = (round((17 * pi) / (8 * epsilon)) + 1) * 2
    plt.xlim(k0 - 10, k0 + 200)
    plt.ylim(3 * pi / 2 - 0.001, 3 * pi / 2 + 0.001)
    for point in range(k0, k0 + 200, 2):
        plt.scatter(point, sequence_value(number=point), color='blue', s=100)
    plt.hlines(3 * pi / 2, 0, k0 + 200, colors='black')
    plt.annotate('superior limit = 3/2 * pi', xy=(k0, 3 * pi / 2), xytext=(k0, 3 * pi / 2 + 0.0001))
    plt.annotate(f'k0 = [(17*pi)/(8*ε)]+1 = {k0}', xy=(k0, 3 * pi / 2), xytext=(k0, 3 * pi / 2 - 0.0002))
    plt.annotate(f'epsilon={epsilon}', xy=(k0, 3 * pi / 2), xytext=(k0, 3 * pi / 2 - 0.0004))
    plt.savefig('subsequence.png', bbox_inches='tight')


def get_graphic_of_supremum(*, sequence, epsilon):
    """
    graphic of supremum
    :param sequence:
    :param epsilon:
    :return:
    """
    plt.figure(figsize=(20, 8), dpi=80)
    plt.title("график супремума")
```

```
    plt.xlabel("n")
    plt.ylabel("значение члена последовательности")
    m = 0
    while True:
        m += 1
        if sequence_value(number=m) > 3 * pi / 2 - epsilon:
            break
    plt.xlim(m - 10, m + 200)
    plt.ylim(3 * pi / 2 - 0.001, 3 * pi / 2 + 0.001)
    for point in range(m, m + 200, 2):
        plt.scatter(point, sequence_value(number=point), color='blue', s=100)
    plt.hlines(3 * pi / 2, 0, m + 200, colors='black')
    plt.annotate('supremum = 3/2 * pi', xy=(m, 3 * pi / 2), xytext=(m, 3 * pi / 2 + 0.0001))
    plt.annotate(f'm = {m}', xy=(m, 3 * pi / 2), xytext=(m, 3 * pi / 2 - 0.0002))
    plt.annotate(f'epsilon={epsilon}', xy=(m, 3 * pi / 2), xytext=(m, 3 * pi / 2 - 0.0004))
    plt.savefig('supremum.png', bbox_inches='tight')


def main():
    original_sequence = [sequence_value(number=n) for n in range(101)]
    get_graphic_of_sequence(sequence=original_sequence)
    get_graphic_of_subsequence(sequence=original_sequence)
    get_graphic_of_supremum(sequence=original_sequence, epsilon=0.0001)


if __name__ == '__main__':
    main()
```