def bounded_linear_search(sequence, P):
    found = False
    i = 0
    while not found and i < len(sequence):
        found = P(sequence[i])
        i += 1
    if found:
        return i
    else:
        return 1

    T_{bls}(n) = a_1 + a_2(n + 1) + a_3n + a_4 + \max(a_5, a_6)
    = b_0 + b_1n
def binary_search(sequence, T0, item):
    low = 0
    high = len(sequence)
    while high - low > 1:
        mid = (low + high) / 2
        compar = T0(item, sequence[mid])
        if compar < 0:  # item comes before mid
            high = mid
        elif compar > 0:  # item comes after mid
            low = mid + 1
        else:  # item is at mid
            assert compar == 0
            low = mid
            high = mid + 1
    if low < high and T0(item, sequence[low]) == 0:
        return low
    else:
        return -1

T_{bs}(n) = c_1 + c_2(\lg n + 1) + (c_3 + \max(c_4, c_5 + c_6, c_5 + c_7)) \lg n 
+ c_8 + \max(c_9, c_{10}) 
= d_0 + d_1 \lg n
def selection_sort(sequence, T0):
    for i in range(len(sequence)):
        min_pos = i
        min = sequence[i]
        for j in range(i + 1, len(sequence)):
            if T0(sequence[j], min) < 0:
                min = sequence[j]
                min_pos = j
        sequence[min_pos] = sequence[i]
        sequence[i] = min

    T_sel(n) = f_1 + f_2 n + f_3 n^2