# Coding Assignment Week 5 CSC 6013

"""

After k iterations of outer loop the k LARGEST elements should be sorted

rather than the k SMALLEST

On each iteration of outer loop, count the number of times two array

elements are compared and the number of times two array elements are

swapped. Reinitialize counters

After each iteration of the outer loop, print three things: the

partially sorted array, number of comparisons on this iteration, and the

number of swaps on this iteration. After the kth iteration, the k

largest elements have been placed into the last k slots of array

"""

def Swap(A, i, j):

    """

    Swap function swaps two elements on array

    Parameters:

        A (list): list of numbers

        i (int): index of first element being swapped

        j (int): index of second element being swapped

    Returns:

        None

    """

    temp = A[i]

    A[i] = A[j]

    A[j] = temp

def SelectionSort(A):

    """

    Selection Sort function sorts list of numbers in ascending order and

    sorts largest numbers first

    Parameters:

        A (list): list of numbers to be sorted

    Returns:

        None

    """

    for i in range(len(A)-1):

        largest = 0

        comparisons = 0

        swaps = 0

        for j in range(0, len(A)-i):

            comparisons = comparisons + 1

            if A[j] > A[largest]:

                largest = j

        Swap(A, len(A)-1-i, largest)

        swaps = swaps + 1

        print("Partially sorted array:", A)

        print("Number of comparisons this iteration: ", comparisons)

        print("Number of swaps this iteration: ", swaps)

**Answers to Question 1:**

Partially sorted array: [63, 44, 17, 77, 20, 6, 39, 84, 52, 99]

Number of comparisons this iteration: 10

Number of swaps this iteration: 1

Partially sorted array: [63, 44, 17, 77, 20, 6, 39, 52, 84, 99]

Number of comparisons this iteration: 9

Number of swaps this iteration: 1

Partially sorted array: [63, 44, 17, 52, 20, 6, 39, 77, 84, 99]

Number of comparisons this iteration: 8

Number of swaps this iteration: 1

Partially sorted array: [39, 44, 17, 52, 20, 6, 63, 77, 84, 99]

Number of comparisons this iteration: 7

Number of swaps this iteration: 1

Partially sorted array: [39, 44, 17, 6, 20, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 6

Number of swaps this iteration: 1

Partially sorted array: [39, 20, 17, 6, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 5

Number of swaps this iteration: 1

Partially sorted array: [6, 20, 17, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 4

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 3

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 2

Number of swaps this iteration: 1

-----------------------

Partially sorted array: [84, 52, 39, 6, 20, 17, 77, 44, 63, 99]

Number of comparisons this iteration: 10

Number of swaps this iteration: 1

Partially sorted array: [63, 52, 39, 6, 20, 17, 77, 44, 84, 99]

Number of comparisons this iteration: 9

Number of swaps this iteration: 1

Partially sorted array: [63, 52, 39, 6, 20, 17, 44, 77, 84, 99]

Number of comparisons this iteration: 8

Number of swaps this iteration: 1

Partially sorted array: [44, 52, 39, 6, 20, 17, 63, 77, 84, 99]

Number of comparisons this iteration: 7

Number of swaps this iteration: 1

Partially sorted array: [44, 17, 39, 6, 20, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 6

Number of swaps this iteration: 1

Partially sorted array: [20, 17, 39, 6, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 5

Number of swaps this iteration: 1

Partially sorted array: [20, 17, 6, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 4

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 3

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 2

Number of swaps this iteration: 1

-----------------------

Partially sorted array: [6, 84, 77, 63, 52, 44, 39, 20, 17, 99]

Number of comparisons this iteration: 10

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 77, 63, 52, 44, 39, 20, 84, 99]

Number of comparisons this iteration: 9

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 63, 52, 44, 39, 77, 84, 99]

Number of comparisons this iteration: 8

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 52, 44, 63, 77, 84, 99]

Number of comparisons this iteration: 7

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 6

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 5

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 4

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 3

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 2

Number of swaps this iteration: 1

-----------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

"""

On each iteration through outer loop, count the number of times two

array elements are compared and the number of times two array elements

are swapped. Reinitialize to zero.

After each iteration through outer loop, print three things: the partially

sorted array, number of comparisons on this iteration, and number of swaps

on this iteration

If no swaps on some iteration, terminate algorithm

When algorithm concludes, display total number of comparisons and swaps

"""

def BubbleSort(A):

    """

    Bubble Sort function sorts list of numbers by iterating through list

    and continuously swapping numbers until in ascending order; exits loop

    early if list is sorted before finishing loop

    Parameters:

        A (list): list of numbers to be sorted

    Returns:

        None

    """

    total\_comparisons = 0

    total\_swaps = 0

    terminate = False

    for i in range(len(A)-1):

        if terminate:

            break

        comparisons = 0

        swaps = 0

        for j in range(len(A)-i-1):

            total\_comparisons = total\_comparisons + 1

            comparisons = comparisons + 1

            if A[j+1] < A[j]:

                Swap(A,j+1,j)

                total\_swaps = total\_swaps + 1

                swaps = swaps + 1

        if swaps == 0:

            terminate = True

        print("Partially sorted array: ", A)

        print("Number of comparisons this iteration: ", comparisons)

        print("Number of swaps this iteration: ", swaps)

    print("Total number of comparisons to sort array: ", total\_comparisons)

    print("Total number of swaps to sort the array: ", total\_swaps)

**Answers to Question 2:**

Partially sorted array: [44, 63, 17, 20, 77, 84, 6, 39, 52, 99]

Number of comparisons this iteration: 9

Number of swaps this iteration: 6

Partially sorted array: [44, 17, 20, 63, 77, 6, 39, 52, 84, 99]

Number of comparisons this iteration: 8

Number of swaps this iteration: 5

Partially sorted array: [17, 20, 44, 63, 6, 39, 52, 77, 84, 99]

Number of comparisons this iteration: 7

Number of swaps this iteration: 5

Partially sorted array: [17, 20, 44, 6, 39, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 6

Number of swaps this iteration: 3

Partially sorted array: [17, 20, 6, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 5

Number of swaps this iteration: 2

Partially sorted array: [17, 6, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 4

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 3

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 2

Number of swaps this iteration: 0

Total number of comparisons to sort array: 44

Total number of swaps to sort the array: 23

-----------------------

Partially sorted array: [52, 6, 39, 20, 77, 17, 84, 44, 63, 99]

Number of comparisons this iteration: 9

Number of swaps this iteration: 7

Partially sorted array: [6, 39, 20, 52, 17, 77, 44, 63, 84, 99]

Number of comparisons this iteration: 8

Number of swaps this iteration: 6

Partially sorted array: [6, 20, 39, 17, 52, 44, 63, 77, 84, 99]

Number of comparisons this iteration: 7

Number of swaps this iteration: 4

Partially sorted array: [6, 20, 17, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 6

Number of swaps this iteration: 2

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 5

Number of swaps this iteration: 1

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 4

Number of swaps this iteration: 0

Total number of comparisons to sort array: 39

Total number of swaps to sort the array: 20

-----------------------

Partially sorted array: [6, 17, 20, 39, 44, 52, 63, 77, 84, 99]

Number of comparisons this iteration: 9

Number of swaps this iteration: 0

Total number of comparisons to sort array: 9

Total number of swaps to sort the array: 0

-----------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

"""

Create an algorithm to evaluate a polynomial using a function that

calculates the power of some real number to some non negative integer

with a for loop (not using Python operator \*\*)

"""

def power(x,p):

    """

    Calculate the power of a real number to a non negative integer

    Parameters:

        x (float): real number base

        p (float): non negative ineger exponent

    Returns:

        exponential (float)

    """

    if p == 0:

        return 1

    else:

        exponential = 1

        for \_ in range(p):

            exponential = exponential\*x

        return exponential

def evaluate\_polynomial(A,x):

    """

    Evaluate polynomial f(x) given list of coefficients of each term from 0 to

    degree of the polynomial and a value of x

    Parameters:

        A (list): list of float values for each coefficient

        x (float): value at which to evaluate polynomial

    Returns:

        y (float): value of f(x) at x

    """

    y = 0

    for i in range(len(A)):

        y = y + power(x,i)\*A[i]

    return y

def main():

    # Question 1

    A1 = [63,44,17,77,20,6,99,84,52,39]

    A2 = [84,52,39,6,20,17,77,99,63,44]

    A3 = [99,84,77,63,52,44,39,20,17,6]

    print("Answers to Question 1: ")

    SelectionSort(A1)

    print("-----------------------")

    SelectionSort(A2)

    print("-----------------------")

    SelectionSort(A3)

    print("-----------------------")

    print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

    # Question 2

    A4 = [44,63,77,17,20,99,84,6,39,52]

    A5 = [52,84,6,39,20,77,17,99,44,63]

    A6 = [6,17,20,39,44,52,63,77,84,99]

    print("Answers to Question 2: ")

    BubbleSort(A4)

    print("-----------------------")

    BubbleSort(A5)

    print("-----------------------")

    BubbleSort(A6)

    print("-----------------------")

    print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

    # Question 3

    func = [12.3,40.7,-9.1,7.7,6.4,0,8.9]

    x\_value = 5.4

    print("Answers to Question 3:")

    answer = evaluate\_polynomial(func, x\_value)

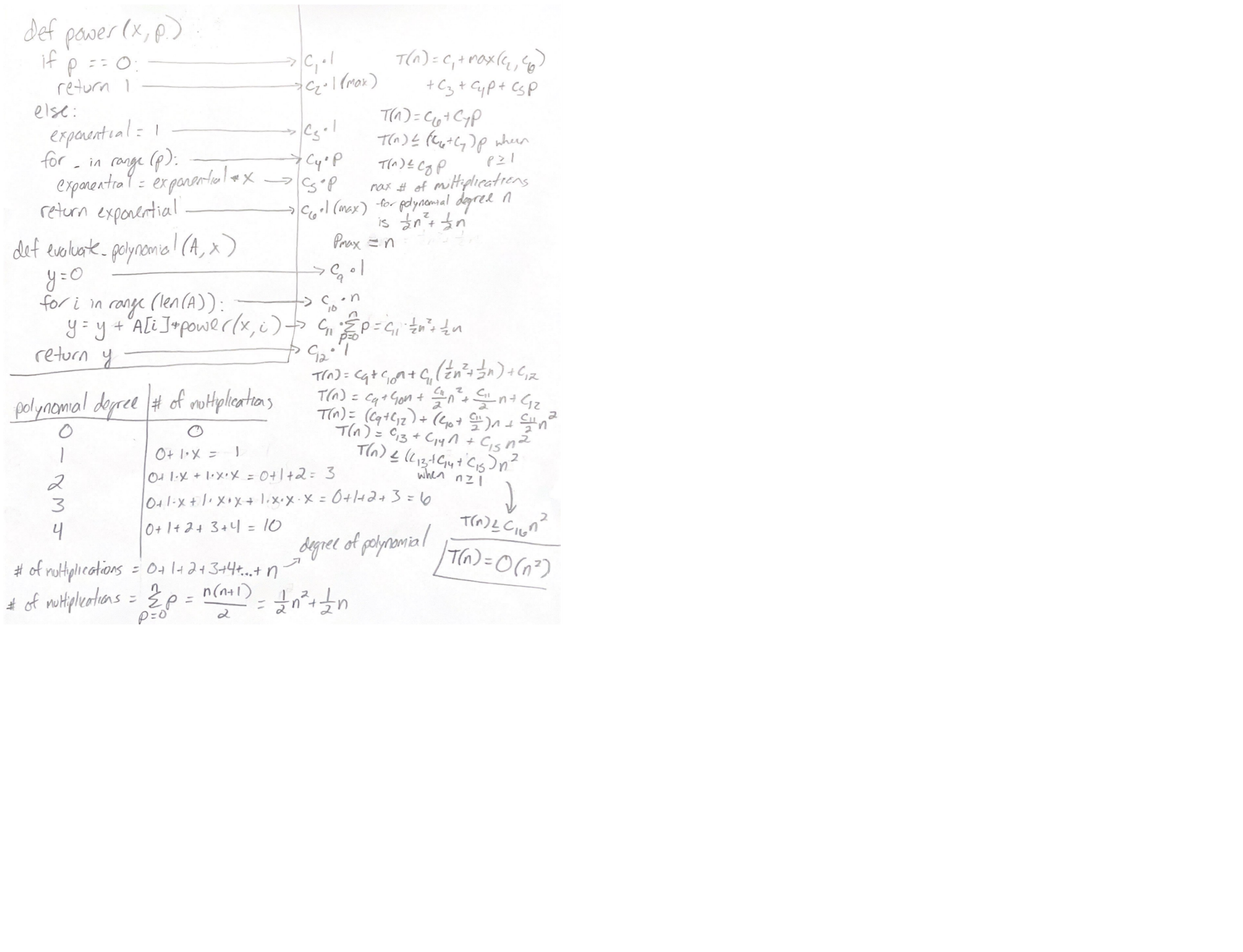
    print(answer)

    print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

main()

**Answers to Question 3:**

227295.86317440012

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*