ЗАДАНИЯ 3

1. Сформировать вектор-строку или вектор-столбец длины *n* из одних только нулей или одних только единиц

А) используя встроенную функцию zeros или, соответственно, ones

Б) используя только одно единственное присваивание скалярного значения последнему элементу вектора (так можно сформировать только нулевой вектор)

В) добавляя в динамически расширяемый вектор последовательно по одному элементу, начиная с первого элемента

Г) добавляя в динамически расширяемый вектор последовательно по одному элементу, используя операцию конкатенации, начиная с пустого вектора

2. Реализовать встроенную функцию length для вектора (определяющую длину вектора) в виде пользовательской функции (лучше - с другим именем)

А) используя только счетчик, операцию удаления одного элемента (первого или последнего) вектора и функцию isempty

(для того, чтобы удалить элемент одномерного массива, этому элементу надо присвоить пустое значение – []; последний элемент массива может индексироваться словом end, и вообще, элементы массива могут индексироваться выражениями, содержащими это слово, например, fix(end/2) – индекс середины массива)

Б) не удаляя элементов из вектора, но используя обработку исключительной ситуации с помощью конструкции try-catch-end, предназначенной для программирования обработки исключительной ситуации (надо последовательно перебирать индексы элементов вектора, до тех пор, пока не произойдет выход за пределы массива, что и приведет к исключительной ситуации)

3. Реализовать встроенную операцию *a*:*h*:*b* в виде пользовательской функции (не использующей эту операцию)

3\*. Дан числовой вектор x (входной параметр), написать функцию, возвращающую

А) среднее значение элементов вектора x

Б) значение максимального (или, как вариант, минимального) элемента вектора x

В) значение первого (или, как вариант, последнего) индекса вектора x, при котором его элемент имеет наибольшее (наименьшее) значение

Г) одномерный массив всех индексов вектора x, при которых его элементы имеют максимальное (минимальное) значение

Д) одномерный массив всех индексов вектора x, при которых его элементы не равны нулю

Е) сумму всех элементов вектора x

Ж) произведение всех элементов вектора x

З) массив всех частичных сумм последовательности, содержащейся в векторе х

И) массив всех частичных произведений последовательности, содержащейся в векторе х

(все перечисленные функции реализованы в языке МАТЛАБ в виде встроенных библиотечных функций: mean, max, min, find, sum, prod, cumsum, cumprod, использовать которые при выполнении этих заданий, разумеется, нельзя)

4. Дан числовой вектор x и вектор p, содержащий некоторую подстановку индексов вектора x (т.е. вектора x и p имеют одинаковую длину, каждый элемент вектора p содержит какое-либо значение индекса вектора х; если при всех i p(i) = i, то имеем тождественную подстановку, в общем случае, p(i)=j, и i не обязательно равно j, но разным i не может соответствовать одно и то же j). Надо переставить элементы в векторе х в соответствии с подстановкой p.

Если воспользоваться встроенной векторной индексацией, то требуемый результат может быть получен так:

x=x(p);

Получить тот же результат, не используя встроенную векторную индексацию

А) используя вспомогательный массив

Б) не используя вспомогательный массив

(Во втором случае воспользоваться тем, что каждая подстановка представляет собой некоторую композицию циклических перестановок (см. <http://www.algebraical.info/doku.php?id=glossary:group:permutation> ). Например, пусть p=[5,1,6,4,2,3], в алгебраической записи эта подстановка и ее разложение в композицию циклов выглядят так . Таким образом, в данном случае задача сводится к 2-м циклическим перемещениям соответствующих элементов вектора x: и . В программной реализации такие циклические перестановки будут осуществляться последовательно, при этом, в процессе осуществления очередной циклической перестановки, соответствующие ей элементы вектора p надо будет как-то помечать. Это удобно делать, просто меняя знаки у этих элементов на отрицательные, и тогда вся процедура должна будет завершена, когда все элементы p станут меньше нуля.)

5. Написать функцию, осуществляющую преобразование произвольного заданного числового вектора в 2 вектора, таких что 1-й из них содержит все ненулевые значения в прежнем порядке, а 2-й - числа нулей перед каждым ненулевым значением, помещаемым в 1-й вектор. Написать функцию, выполняющую обратное преобразование.

6. Написать функцию, меняющую порядок следования элементов вектора на обратный (эту операцию реализуют встроенные функции fliplr и flipud применительно к вектору-строке и вектору-столбцу, соответственно; их использовать не надо, конечно)

7. Написать функцию переставляющую первые *k* элементов массива в его конец, не изменяя порядка следования как этих *k* элементов, так и всех остальных (рассматривая эти две части массива по отдельности).

(попробуйте обойтись без использования вспомогательного массива; интересные варианты решения, можно найти в <http://www.e-academy7.narod.ru/COURSES/PROGRAM/LITERATURA/01shen.PDF> , задача 1.2.11)

8. Написать функцию, осуществляющую циклический сдвиг элементов вектора на *k* позиций (переместить 1-й элемент вектора на *k*-ю позицию, 2-й – на *k*+1-ю и т.д., считая при этом, что за последним элементом вектора как бы следует снова первый и т.д., и, таким образом, ни какой элемент вектора не должен быть «затерт», а должен быть перемещен на новое место)

9. Написать функцию, осуществляющую удаление отрицательных элементов из вектора

A) сокращая при этом длину вектора

(использовать инвариант цикла вида: «*n* = текущая длина вектора, И среди первых *k* элементов вектора отрицательных элементов нет»)

Б) не сокращая длину вектора, а перемещая следующие за отрицательными элементами неотрицательные элементы на место отрицательных (не изменяя первоначального порядка остающихся элементов), при этом появляющиеся в конце вектора лишние элементы заменять на значение NaN. Получить 2 варианта решения: с использованием вспомогательного массива и без.

(в случае «без вспомогательного массива» использовать инвариант цикла вида: «*n* = наибольший индекс вектора, при котором элемент не равен NaN, И среди первых *k* элементов вектора отрицательных элементов нет»)

10. Переместить все отрицательные элементы в начало вектора, не изменив при этом первоначального порядка следования отрицательных и положительных элементов. Получить 2 варианта решения: с использованием вспомогательного массива и без.

(в случае «без вспомогательного массива» использовать инвариант цикла вида: «среди первых m элементов вектора нет ни одного положительного, И, начиная с *n*-го элемента вектора, - нет ни одного отрицательного, И порядок следования отрицательных и положительных элементов вектора по отдельности совпадает с первоначальным»)

11. Дан числовой вектор, элементы которого расположены по не убыванию их значений (последующее больше или равно предыдущему). Требуется удалить все повторяющиеся элементы. Написать функцию с соответствующими входным и выходным параметрами.

(Сначала сформулировать нужный инвариант цикла, а потом в соответствии с ним реализовать алгоритм. Как можно было бы решить задачу, если бы исходный вектор заранее не был отсортирован? Оцените вычислительную сложность предложенного решения.)

12. Написать функцию, осуществляющую объединение двух отсортированных векторов в один, так, чтобы результирующий вектор получался тоже отсортированным и содержал все элементы исходных векторов. Требуемое число операций должно оцениваться как O(n1+n2), n1, n2 – длины исходных массивов.

13. Написать функцию осуществляющую проверку (функция должна возвращать логическое значение), содержится ли заданное числовое значение в данном отсортированном массиве за O(log(длина\_массива)) операций сравнения (указанную процедуру называют быстрым поиском).

14. Написать функцию, осуществляющую сортировку одномерного числового массива и вместе с отсортированным массивом возвращающую соответствующий вектор подстановок индексов этого массива так, как это реализовано у встроенной библиотечной функции sort (ее заголовок имеет вид [x,p]=sort(x)). Применить какой-либо алгоритм сортировки квадратичной сложности (число операций сравнения оценивается как O(log(длина\_массива^2))), например,

А) обменную сортировку, которая сводится к повторениям следующих 2-х шагов применительно к укорачивающейся с каждым разом на 1 элемент начальной части массива: 1) найти индекс наибольшего элемента рассматриваемой части массива 2) поменять местами последний элемент части массива с найденным наибольшим, в результате последний элемент этой части массива будет иметь требуемое значение (первоначально рассматривается весь массив целиком)

Б) сортировку методом «пузырька», которая сводится к повторениям следующей процедуры применительно к укорачивающейся с каждым разом на 1 элемент начальной части массива: сравнить первый элемент со вторым, если 1-ый больше, то поменять их местами, затем сравнить 2-ой элемент с 3-им, и, при необходимости поменять их местами, и т.д. до конца рассматриваемой части массива, в результате последний элемент этой части массива будет иметь требуемое значение (первоначально рассматривается весь массив целиком)

15. Написать функцию, осуществляющую быструю сортировку одномерного числового массива за O(n\*log(n)) число сравнений методом слияний (n=длина\_ массива).

Можно дать рекурсивное решение. Не рекурсивное решение приведено, например, в <http://www.e-academy7.narod.ru/COURSES/PROGRAM/LITERATURA/01shen.PDF> , задача 4.2.1)

16. Дан некоторый числовой массив (вектор) x и некоторое число b. Требуется переставить элементы массива таким образом, чтобы сначала располагались элементы меньшие или равные этому числу, а затем – большие или равные ему; требуемое число операций - O(log(длина\_массива)).

(воспользоваться инвариантом цикла вида: x(1)<=b,…,x(L)<=b & b<=x(R+1),…,b<=x(end), где L,R – некоторые вспомогательные фазовые переменные;

все решение можно найти, например, в <http://www.e-academy7.narod.ru/COURSES/PROGRAM/LITERATURA/01shen.PDF> , задача 1.2.31)

17. Дан некоторый числовой массив (вектор) x и некоторое число b. Требуется переставить элементы массива таким образом, чтобы сначала располагались элементы меньшие этого числа, затем – равные ему, и затем - большие его; требуемое число операций - O(log(длина\_массива)).

(воспользоваться инвариантом цикла вида: x(1)<b,…,x(L)<b & x(L+1) == b,…, x(L+M) == b & b<x(R+1),…,b<x(end), где L,M,R – некоторые вспомогательные фазовые переменные; все решение можно найти, например, в <http://www.e-academy7.narod.ru/COURSES/PROGRAM/LITERATURA/01shen.PDF> , задача 1.2.32)

18. Реализовать быструю сортировка Хоара, требующую в среднем O(n\*log(n)) числа операций сравнения, где n=длина\_массива (см. <http://www.e-academy7.narod.ru/COURSES/PROGRAM/LITERATURA/01shen.PDF>, раздел 4.2.1, стр. 86-87).

Указанный алгоритм содержит следующие шаги: 1) выберем случайным образом некоторый элемент массива, пусть b есть его значение; 2) переставим элементы массива так, чтобы сначала располагались элементы меньшие числа b, затем – равные ему, и затем - большие его (см. предыдущую задачу). В результате средняя часть массива (все элементы которой равны b) окажется на требуемом месте, и остается только применить туже процедуру к каждой из двух оставшихся частей массива.

Возможны как рекурсивный (см. <http://www.e-academy7.narod.ru/COURSES/PROGRAM/LITERATURA/01shen.PDF> , задача 7.4.7), так и не рекурсивный варианты решения (см. там же, задача 8.2.7).

19. Даны некоторый числовой массив и некоторый второй числовой массив, упорядоченный по возрастанию, и содержащий все возможные значения, которые могут иметь элементы первого массива (считается, что первый массив может иметь сколь годно большую длину, а длина второго - как бы фиксирована и относительно небольшая). Написать функцию, упорядочивающую первый массив за O(n) число сравнений, где n=длина\_первого\_массива (сортировка за линейное время).

(Идея алгоритма: посчитать сколько раз каждое значение 2-го массива встречается в первом, а затем сформировать требуемый массив, путем дублирования в нем в нужном порядке элементов 2-го массива подсчитанные числа раз)

20. Написать функцию, получающую на вход некоторое целое неотрицательное число и основание системы счисления *q* и возвращающую вектор *q*-ичных цифр этого числа

21. Реализовать «длинную арифметику» для *q*-ичного представления целых неотрицательных чисел (достаточно реализовать только операцию умножения) и используя ее написать функцию, возвращающую для заданного *n* вектор десятичных цифр *n*!.

22. Реализовать функцию size для матрицы (определяющую размеры матрицы)

А) используя только операцию удаления одной строки (первой или последней) матрицы и функцию isempty

Б) не удаляя строк из матрицы, но используя обработку исключительной ситуации с помощью конструкции try-catch-end (надо последовательно перебирать индексы строк матрицы, до тех пор, пока не произойдет выход за пределы массива, что и приведет к особой ситуации)

23. Дана матрица, требуется A) cформировать диагональную матрицу с той же диагональю; Б) сформировать 3-х диагональную матрицу с теми же диагоналями; В) верхнюю треугольную матрицу и нижнюю треугольную матрицы, сумма которых равна исходной матрице

24. Написать функцию, формирующую «треугольник Паскаля» заданного размера в виде квадратной матрицы (половина которой ниже побочной диагонали не используется и состоит из нулевых элементов)

25. Написать функцию, возвращающую вектор биномиальных коэффициентов порядка n.

26. Написать функцию, сортирующую строки матрицы в, так называемом, лексико-графическом порядке (пусть s1, s2 – две произвольные строки матрицы, тогда, согласно лексико-графическому порядку, s1 предшествует s2, тогда и только тогда для самого первого i, такого что s1(i) ~= s2(i) выполняется условие s1(i) < s2(i))

27. Сгладить данные, содержащиеся в числовом массиве, заменив каждый элемент вектора (или матрицы) средним арифметическим соседних элементов

А) используя вспомогательный массив

Б) не используя вспомогательный массив

28. Реализовать матричные операции +, -, \*, /, \, ^, &, |, ==, ~=, <=, >=, <, > , .\*, ./, \., .^ (хотя бы некоторые из них, т.к. реализовать операции /, \ и, особенно, ^ в полном объеме не так просто), используя только операции со скалярными значениями.

29. Дана числовая матрица x. Требуется найти максимальное значение среди минимумов ее строк.

Возможно такое решение с помощью встроенных матричных операций:

max\_min = max(min(x,2));

Написать соответствующую функцию, не использующую матричных операций (использующую только скалярные операции)

30. Используя встроенную функцию line (или plot) и векторизацию операций, построить все (или хотя бы только некоторые) графики из задания 2.

Например, пусть требуется построить семейство концентрических окружностей радиусов 1,2,3,…,n. Решение:

%n - входной параметр

t = [0:0.1:2\*pi, 0];

w=1; color = ’r’

x = cos(t); y = sin(t); h = line(x,y);

set(h, ’linewidth’, w, ’color’, color)

**for** a=2:n

w=2\*w;

xa = a\*x; ya = a\*y; h = line(x,y);

set(h, ’linewidth’, w, ’color’, color)

**end**

Или, пусть требуется построить график кардиоиды, параметрические уравнения которой имеют вид: . Решение:

t = [0:0.1:2\*pi, 0];

r = (1-sin(t));

x = r.\*cos(t); % поэлементное перемножение вектора на вектор

y = r.\*sin(t);

line(x,y)

Или, пусть требуется построить график зависимости

от x, например, для и, соответственно, **.** Решение:

dx = 0.1 % стоит попробовать и другие значения, например, 1e-16

x = 0:0.1:pi;

delta=abs(cos(x)-(sin(x+dx)-sin(x))/dx);

line(x,delta)

31. Дан числовой вектор, требуется подсчитать число нулей. Возможно такое решение с помощью встроенных матричных операций: sum(x==0). Написать соответствующую функцию, не использующую матричных операций (использующую только скалярные операции)

32. Дан числовой вектор х, требуется проверить,

А) все ли его элементы не равны нулю (специально предназначенную для этого встроенную функцию all(x) не использовать)

Б) хотя бы один его элемент не равен нулю (специально предназначенную для этого встроенную функцию any(x) не использовать)

33. Дана числовая матрица x, требуется подсчитать число нулевых столбцов (строк) матрицы

Возможно такие решения с помощью встроенных матричных операций:

sum(all(x,1)) – в случае столбцов sum(all(x,2)))

Написать соответствующую функцию, не встроенных использующую матричных операций (использующую только скалярные операции)

34. Дана числовая матрица x, требуется подсчитать число НЕ нулевых столбцов (строк) матрицы

Возможно такие решения с помощью встроенных матричных операций:

sum(any(x,1)) – в случае столбцов, sum(any(x,2)) – в случае строк (почему следующее решение не правильное: size(x,2) - sum(all(x,1) (?));

Написать соответствующую функцию, не использующую встроенных матричных операций (использующую только скалярные операции)

35. Используя только скалярные операции проверить, равны ли 2 вектора или 2 матрицы (решение с помощью встроенных матричных операций: all(x(:)==y(:)))

36. Используя только скалярные операции реализовать операции (написать соответствующие функции) транспонирования матрицы, поворота матрицы вокруг вертикальной или горизонтальной оси (встроенные матричные функции fliplr, flipud), поворота матрицы на 90 градусов (встроенная функция rot90), переформатирования матрицы (встроенная функция reshape)

37. Используя только скалярные операции реализовать следующие встроенные матричные функции: mean, max, min, find, sum, prod, cumsum, cumprod для матричного аргумента

(случай векторного аргумента уже должен быть разобран, см. задачу 3\*; при выполнении данного задания также все еще допускается сужать функциональность этих встроенных функций до разумных пределов)

38. Написать функцию, которая бы с помощью встроенной функции rand генерировала координаты *n* случайных точек плоскости и отображала их в координатных осях в виде маркеров в форме ‘\*’ с помощью встроенной функции line, распределяя их «равномерно»

А) в прямоугольнике: ()

Б) в круге единичного радиуса (лишние генерируемые точки отсеивать)

В) в квадрате: (воспользоваться преобразованием поворота на 45 градусов)

39. Написать функцию, которая бы с помощью встроенной функции randi генерировала целочисленные координаты *n* случайных точек плоскости и отображала их в координатных осях в виде маркеров в форме ‘\*’, с помощью встроенной функции line, распределяя их «равномерно»

в прямоугольнике: ()

40. Написать функцию, которая бы с помощью встроенной функции randn генерировала координаты случайных точек плоскости, располагающихся в окрестностях заданных точек, по n случайных точек в каждой окрестности, и отображала их в координатных осях в виде маркеров в форме ‘\*’ с помощью встроенной функции line, распределяя их не равномерно (по так называемому гауссовскому или, что - то же, нормальному закону). Координаты заданных точек, как и число n, должны передаваться в эту функцию через соответствующие входные параметры.

41. Реализовать встроенную функцию meshgrid, формирующую массивы с координатами точек узлов декартовой координатной сетки в заданных пределах.

42. Воспользовавшись встроенными функциями meshgrid и surf построить график поверхности

А) ;

Б) сферы единичного радиуса в пересечении с цилиндром, радиуса 0.5, ось которого проходит через точку (0.5,0,0)

43. Даны две числовых вектор-строки x и y равной длины, требуется сформировать вектор-строку z удвоенной длины, в котором элементы данных векторов чередовались бы: z(1)=x(1), z(2)=y(1), z(3)=x(2), z(4)=y(2),…

С использованием матричных операций этого можно добиться так: z=[x; y]; z=z(:).’;

(пожалуйста, разберитесь, почему это так). Написать соответствующую функцию, использующую только скалярные операции.

44. Используя только скалярные операции реализовать следующие встроенные матричные функции для работы с множествами: операции с множествами union, intersect, setdiff, unique, ismember

(при реализации этих функций рассмотреть два случая: (1) множество представлено отсортированным массивом и (2) – не отсортированным, в связи с этим см. также задачи 11, 12; в данном пункте допускается также несколько сужать функциональность указанных встроенных функций до разумных пределов)

45. Написать функцию, проверяющую (возвращающую соответствующий вектор логических значений), какие из элементов заданного целочисленного вектора являются простыми. (Предназначенную для этого встроенную функцию isprime не использовать.)

46. Написать функцию, осуществляющую факторизацию (разложение на простые множители) заданного натурального числа. Полученные множители должны быть элементами возвращаемого функцией вектора.

(Предназначенную для этого встроенную функцию factor не использовать.)

47. Реализовать алгоритм быстрого вычисления *n*-го числа Фибоначчи, воспользовавшись соотношением .

(Свести к «быстрому» возведению соответствующей матрицы в степень n. Использовать встроенные матричные операции разрешается.)

48. Используя элементарные преобразования матрицы, написать функцию

А) возвращающую матрицу, обратную к данной (в случае, если матрица не имеет обратной, возвращать пустой массив)

Б) вычисление определителя

В) возвращающую решение системы линейных уравнений, заданной своей расширенной матрицей

(в случае, если система несовместна или имеет более одного решения возвращать пустой массив, а также значение, содержащee информацию о числе решений, т.е. 0, 1 или inf)

Во всех трех случаях надо иметь в виду следующее. Напомним, что для приведения матрицы x к ступенчатому виду, используется элементарная операция, состоящая в замене некоторой, скажем i-ой, строки суммой этой строки с другой, скажем j-ой, строкой, умноженной на некоторый коэффициент, назовем его t. На языке МАТЛАБ это преобразование записывается в виде:

x(i,:) = x(i,:) + t\*x(j,:); % x(i,:) – это вся i-я строка матрицы целиком

Чтобы после такого преобразования при некотором j< i на месте x(j,i) получился ноль, необходимо положить t = - x(j,i)/x(i,i). Однако не достаточно просто потребовать, чтобы

x(i,i)~=0 (как это обычно бывает на упражнениях по алгебре). Это было бы достаточным, если бы вычисления выполнялись с абсолютной точностью, но для компьютерных вычислений в формате с плавающей точкой это не так. Поэтому всегда нужно предполагать, что каждое хранимое в памяти значение x(i,j) имеет некоторую погрешность. При выполнении преобразования эта погрешность умножается на коэффициент t, поэтому если допускается, что |t| >1, то при многократных преобразованиях погрешность может быстро нарастать до катастрофически больших значений.

Поэтому необходимо соблюдать условие . Этого можно добиться путем такой транспозиции строк матрицы, после которой будет иметь место равенство = max{ } (n - размер матрицы). Эта необходимая процедура называется выбором ведущего элемента.

Кроме того, в случае, если матрица вырождена, то на каком-то шаге приведения ее к ступенчатому виду где-то на главной диагонали должен получиться ноль. Однако в силу приближенного характера вычислений, вместо ожидаемого нуля может получиться только достаточно малое по модулю значение не равное нулю. Поэтому важно установить порог, при достижении которого получаемое значение должно считаться равным нулю. В качестве такого порогового значения можно взять, например, величину 1e-7 (читается: 10 в минус 7-ой степени).

49. Написать функцию, возвращающую НОД (или, как вариант, НОК) данного массива целых чисел (входной параметр).

(Свести задачу к вычислению НОД пары чисел. Для вычисление НОД пары чисел реализовать алгоритм Евклида, для чего сначала сформулировать инвариант цикла; решение см., например, в книге В.В. Борисенко «Основы программирования» <http://www.intuit.ru/studies/courses/2193/67/lecture/1968?page=2> ).

50. Написать функцию, реализующую расширенный алгоритм Евклида.

(Сначала сформулировать инвариант цикла; решение см., например, в книге В.В. Борисенко «Основы программирования» <http://www.intuit.ru/studies/courses/2193/67/lecture/1968?page=5>)

51. Дан массив коэффициентов некоторого многочлена, расположенных по убыванию степеней, и числовое значение его аргумента (входные параметры функции). Написать функцию, вычисляющую

A) значение этого многочлена в заданной точке по схеме Горнера (решение см., например, в книге В.В. Борисенко «Основы программирования» <http://www.intuit.ru/studies/courses/2193/67/lecture/1966?page=2>)

Б) одновременно, значение этого многочлена и его производной в заданной точке; к проектированию алгоритма подойти на основе теории индуктивных функций, получив необходимое индуктивное расширение (решение см., например, в книге В.В. Борисенко «Основы программирования» <http://www.intuit.ru/studies/courses/2193/67/lecture/1966?page=4> )

52. Дан одномерный массив x с некоторыми данными. Требуется вычислить так называемое среднее квадратическое отклонение (СКО) этих данных, определяемое формулой

, где . Написать функцию, получающую на вход массив данных и возвращающую соответствующее СКО и реализующую однопроходный алгоритм.

(К проектированию алгоритма подойти на основе теории индуктивных функций, получив необходимое индуктивное расширение; решение разбирается в лекционном курсе)

53. Обобщенной частичной суммой числовой последовательности будем называть сумму вида , где . Требуется написать функцию, возвращающую максимальную обобщенную частичную сумму данной числовой последовательности, представленной вектором (входной параметр) и реализующую однопроходный алгоритм (сложность которого будет оцениваться как O(n)).

(К проектированию алгоритма подойти на основе теории индуктивных функций, получив необходимое индуктивное расширение; решение разбирается в лекционном курсе)

Дополнительные задачи.

- Cформировать матрицу, строки которой содержат все перестановки чисел 1,2,…,n

(можно рекурсивно, не рекурсивное решение см., например, в <http://www.e-academy7.narod.ru/COURSES/PROGRAM/LITERATURA/01shen.PDF> , задача 2.2.1)

- Реализовать алгоритм «решето Эратосфена» для поиска всех простых чисел, не превосходящих заданного n.

(Описание алгоритма см, например, на странице <https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D1%88%D0%B5%D1%82%D0%BE_%D0%AD%D1%80%D0%B0%D1%82%D0%BE%D1%81%D1%84%D0%B5%D0%BD%D0%B0> )

- Вычислить коэффициенты многочлена Чебышева n-го порядка

- Вычислить коэффициенты произведения двух многочленов, заданных своими коэффициентами (решение см., например, в <http://www.e-academy7.narod.ru/COURSES/PROGRAM/LITERATURA/01shen.PDF> , задача 1.2.15)

- Вычислить коэффициенты частного и остатка при делении двух многочленов, заданных своими целочисленными коэффициентами

- Вычислить НОД двух многочленов, заданных своими целочисленными коэффициентами

- Полиномиальная интерполяция (на равномерной сетке, на чебышевской сетке)

- Даны координаты некоторого конечного множества точек плоскости. Вычислить

А) наибольшее расстояние, определяемое между парами этих точек

Б) наибольшую площадь треугольника с вершинами в этих точках

В) наибольшую длину замкнутой ломаной с n узловыми точками (возможно имеющей точки самопересечения), где n – входной параметр.

- Лабиринт, граница которого представляет собой плоскую ломаную, задан массивами с координатами узловых точек границы (для формирования данных использовать ginput – встроенную функцию графического ввода). Вычислить длину границы лабиринта.

- Проверить, находится ли точка внутри лабиринта

- Дано множество отрезков на плоскости (даны массивы с координатами их концов). Найти все пересекающиеся отрезки (отметить точки пересечений)

- Построить выпуклую оболочку заданного конечного множества точек плоскости.

- Вычислить площадь лабиринта – это уже точно в следующем семестре, когда студенты будут знать формулу Грина