Допълнителни задачи по СНИ към упражнения 3 и 4

Зад. 1 Да се напише функция, която приема като параметър списък с числа. Да се обходят всички елементи на списъка и за всеки от тях да се провери дали е естествено число.
Ако елементът е естествено число, да се отпечата на екрана неговият факториел. В противен случай, да се изведе подходящо съобщение на екрана, например “The input is not a natural number”.
Упътване: Можете да проверите дали едно число е цяло като използвате ***IntegerQ.***

Зад. 2 Стандартните компютри могат да извършват само основни аритметични операции (събиране, изваждане, умножение, деление). Поради тази причина, стойностите на повечето математически функции не могат да бъдат пресметнати точно. Вместо това, те се приближават с помощта на различни алгоритми, включващи в себе си голям брой аритметични операции – например като се използва развитие в ред на Тейлър за съответната функция. Тези операции могат да имат голяма изчислителна сложност, особено ако се изисква висока точност при приближението. Например в някои приложения на компютърната графика, във всяка секунда се налага пресмятането на стойността на функцията синус в хиляди точки, което, имайки предвид казаното по-горе, може значително да забави работата на програмата. Едно класическо решение на този проблем е употребата на т.нар. ***lookup table*** – масив от данни, съдържащ предварително пресметнати стойности на функцията в краен брой точки. Когато е необходима стойността на функцията в конкретна точка, тя не се пресмята в реално време, а вместо това се използват данните от lookup таблицата. Например за приближение на търсената стойност, може да се използва стойността от таблицата, съответстваща на най-близката налична точка. По този начин същинските пресмятания се заместват с операция за достъпване на елемент от масив, която е значително по-бърза (за повече информация вж. Lookup table - Wikipedia).

Задача:
Да се напише функция, която да намира приближено стойността на *sin x* в интервала $[0, 2\pi]$. За целта да се използва предварително генериран списък от *N+1* наредени двойки от вида $(\alpha_i, \sin \alpha_i)$, където $\alpha_i = i \frac{2\pi}{N}$, $i = 0, \ldots, N$. Функцията да приема като параметри *x* и *N* и да връща приближено стойността на *sin x,* като използва най-близката налична в таблицата точка. Сравнете получените резултати със стойността, която системата *Mathematica* връща. Експерериментирайте със стойността на параметъра *N*, за да получите по-добро приближение.

Зад. 3 Правилото на Крамер е метод за решаване на системи линейни уравнения. Методът работи със системи, при които броят на уравненията и броят на неизвестните съвпадат и системата има единствено решение.
Да разгледаме система от *n* линейни уравнения с *n* неизвестни, представена във векторно-матрична форма: $A\boldsymbol{x} = \boldsymbol{b}$. Нека матрицата *A* има размерност *n x n* и $det\ A \neq 0$, а векторът $\boldsymbol{x} = (x_1 \ldots, x_n)^T$ е вектор-стълб, съдържащ неизвестните. Тогава системата има

единствено решение и то се задава по формулата:

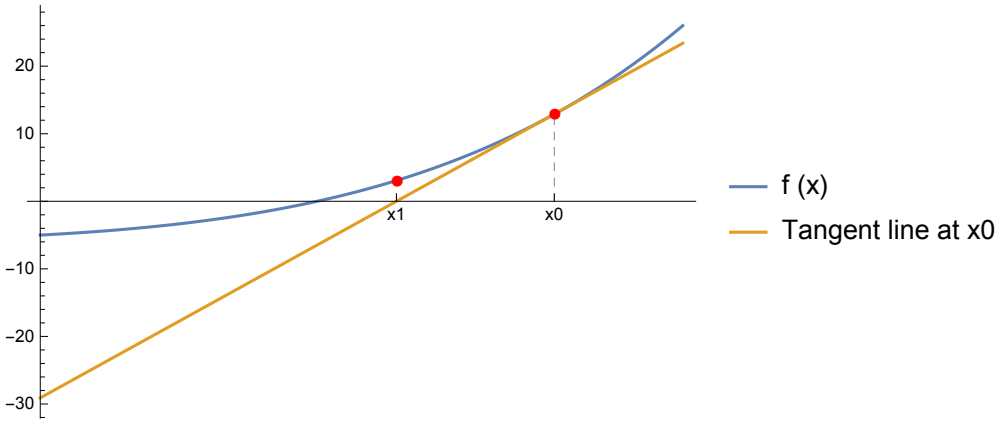
$$x_i = \frac{det(A_i)}{det(A)}, \qquad i = 1,...,n,$$

където A_i е матрица, образувана чрез заместването на i-тия стълб на матрицата A с вектора $\boldsymbol{b}$.

Задача:
Да се напише функция, която намира решението на линейна система от горепосочения вид по правилото на Крамер. Функцията да приема като параметри квадратната матрица на системата и вектора на десните части. В случай че матрицата A има нулева детерминанта, да се изведе подходящо съобщение. В противен случай, функцията да връща вектора с решението на системата.

Зад.4 На практика често възниква необходимостта от решаване на нелинейни уравнения. В общия случай, обаче, те не могат да бъдат решени аналитично. Затова възниква необходимостта от използване на методи за приближеното им решаване. Един от най-популярните алгоритми за числено пресмятане на корени на уравнения е методът на Нютон. Да разгледаме следната задача:
Искаме да намерим корен на уравнението $f(x) = 0$ в интервала [a,b] (*вж. забележката по-долу*). Да предположим, че можем да намерим някакво начално приближение на стойността на търсения корен - да го означим с x_0. Целта ни е да построим редица от приближения, всяко следващо от които е по-добро от предходното (по-близко е до действителната стойност на корена). Като за начало, нека построим допирателната към графиката на f(x) в т. x_0:

$$y = f(x_0) + f'(x_0)(x - x_0)$$



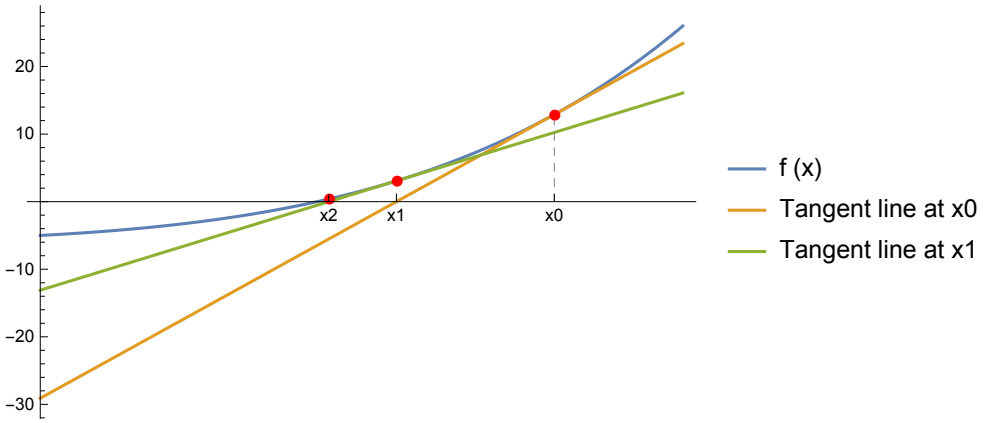
От графиката виждаме, че оранжевата линия (допирателната към графиката на f(x) в т. x_0), пресича абсцисната ос доста по-близо до действителното решение, отколкото е x_0. Нека означим пресечната точка на допирателната с абцисната ос с x_1 и да я използваме за следващо приближение. Как можем да намерим тази точка? Знаем, че координатите и са $(x_1, 0)$ и че лежи на допирателната y:

$$0 = f(x_0) + f'(x_0)(x_1 - x_0)$$

$$x_1 - x_0 = -\frac{f(x_0)}{f'(x_0)}$$

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Сега можем да повторим процедурата, като построим допирателната към графиката на f(x) в т. x_1, очаквайки да получим още по-добро приближение:



Метод на Нютон:

Следвайки горния подход, можем да построим редица от приближения x_1, x_2 , x_3,..., като използваме връзката:

$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$.

Повтаряме процедурата, докато две съседни приближеня станат достатъчно близки, или достигнем максимален брой итерации на алгоритъма.

Забележка: За съществуването на корен в интервала [a,b], е необходимо f(a)f(b)<0. За да има задачата единствено решение, е необходимо още да е изпълнено условието f'(x)f''(x) ≠ 0.

Задача:

(а) Да се напише функция, която намира корен на уравнение от вида f(x)= 0 по метода на Нютон. Функцията да приема като параметри *f(x)*, x_0 - начално приближение, ϵ - толеранс (допустима разлика между две съседни приближения), maxIterations - максимален брой итерации;

(б) Като изпозлвате написаната функция, намерете приближено решения на следните уравнения с точност до 6 знака след десетичната запетая:

(i) $\cos x = x$, $x \in [0, 2]$;

(ii) $x^2 = e^{2-x^2}$, $x \in [0, 2]$;

(в) Намерете всички корени на уравнението ln x = 2 cos x с точност до 6 знака след дестеичната запетая;

(г) Като използвате метода на Нютон, пресметнете стойността на $\sqrt{50}$ с точност до 5 знака

след десетичната запетая.

Сигурно се досещате, че СКА Mathematica, както и всяка друга съвременна СКА разполага с вградени алгоритми за намиране на корените на уравнения. Защо тогава е необходимо да имплементираме методът на Нютон сами? Едната причина е, че това е универсален метод, на чиято идея се базират много други, по-сложни техники за намиране на корени на уравнения, които не са вградени в компютърните програми. Нещо повече, често на практика, вградените фунцкии в СКА не могат да ни дадат оптимални резултати. Причината за това е, че те прилагат универсални подходи, с които могат да се решат широк клас от задачи, но биха могли да изпитат затруднения с някои по-специални частни случаи. Накрая, но не на последно място, за да използваме интелигентно възможностите на компютърните системи, е важно да познаваме математическите алгоритми, които стоят зад тях.