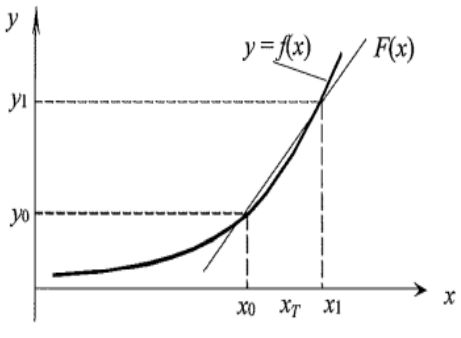


## Задание

Написать программы для нахождения промежуточных значений и построить графики функции, заданной в  $n$  точках:

### 1) с использованием кусочно-линейного интерполирования:

При кусочно-линейном интерполировании функция  $f(x)$  на интервале  $x_i \leq x \leq x_{i+1}$ , ( $i=0,1,\dots,n-1$ ) аппроксимируется отрезком прямой

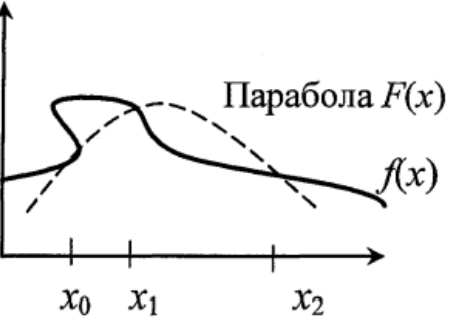
	$y = a_i x + b_i,$ $a_i = \frac{y_i - y_{i-1}}{x_i - x_{i-1}},$ $b_i = y_{i-1} - a_i x_{i-1}$	$R_n(x) = f(x) - P_n(x)$ $R_1(x) = \frac{M_2}{8} h^2.$ $M_2 = \max  f''(x) ,$ $h = (x_i - x_{i-1})$
---	---	---

Причем, вначале ищем, на каком отрезке  $[x_i, x_{i+1}]$ . находится искомая точка  $x$ .

### 2) с использованием кусочно-параболического интерполирования.

При кусочно-параболическом интерполировании полином строится на интервале  $[x_{i-1}, x_i, x_{i+1}]$  по 3-м узловым точкам, ближайшим к заданному значению аргумента.

Алгоритм вычисления, реализующий кусочно-параболическое интерполирование, может быть записан:

	$y = a_i x^2 + b_i x + c_i$ $\begin{cases} a_i x_{i-1}^2 + b_i x_{i-1} + c_i = y_{i-1} \\ a_i x_i^2 + b_i x_i + c_i = y_i \\ a_i x_{i+1}^2 + b_i x_{i+1} + c_i = y_{i+1} \end{cases}$ $x \in (x_{i-1}, x_i) \quad \text{или} \quad x \in (x_i, x_{i+1}).$	$R_2(x) = (x - x_0)(x - x_1)(x - x_2) \frac{M_3}{6}$ $M_3 = \max  f'''(x) .$
---	---	--

Система уравнений для нахождения неизвестных коэффициентов  $a_i, b_i, c_i$ , решается с использованием встроенного модуля **Python**.

3) **посредством построения полинома Лагранжа.** Для построения полиномов Лагранжа возрастающих степеней может быть применена следующая итерационная схема (схема Эйткена).

**Разобрать самостоятельно, пример построения приведен:**

Полиномы, проходящие через две точки  $(x_i, y_i)$ ,  $(x_j, y_j)$

$(i=0,1,\dots,n-1 ; j=i+1,\dots,n)$ , могут быть представлены таким образом:

$$L_{ij} = \frac{1}{x_j - x_i} \begin{vmatrix} x - x_i & y_i \\ x - x_j & y_j \end{vmatrix}$$

Полиномы, проходящие через три точки  $(x_i, y_i)$ ,  $(x_j, y_j)$ ,  $(x_k, y_k)$   $(i=0,\dots,n-2 ; j=i+1,\dots,n-1 ; k=j+1,\dots,n)$ , могут быть выражены через полиномы  $L_{ij}$  и  $L_{jk}$ :

$$L_{ijk} = \frac{1}{x_k - x_i} \begin{vmatrix} x - x_i & L_{ij} \\ x - x_k & L_{jk} \end{vmatrix}$$

Полиномы для четырёх точек  $(x_i, y_i)$ ,  $(x_j, y_j)$ ,  $(x_k, y_k)$ ,  $(x_l, y_l)$  строятся из полиномов  $L_{ijk}$  и  $L_{jkl}$ :

$$L_{ijkl} = \frac{1}{x_l - x_i} \begin{vmatrix} x - x_i & L_{ijk} \\ x - x_l & L_{jkl} \end{vmatrix}$$

Процесс продолжается до тех пор, пока не будет получен полином, проходящий через  $n$  заданных точек.

Здесь массив  $L$  – это промежуточные значения полинома Лагранжа. Первоначально следует положить значение  $L$  равными  $y_i$ . После выполнения  $n$  циклов – это значение полинома Лагранжа степени  $n$  в точке  $x$ .

4) **с использованием кубического сплайн интерполирования ( на основании встроенного модуля Python).**