COMPUTER SCIENCE 2

Lab meeting I: Polynomial interpolation

1. Write a function **`double lagrange(double x_int[], double y_int[], int n, double x)`** which calculates the value of the n-th order Lagrange's polynomial $P_n(x)$ defined for the set of nodes {x(0), y(0)}, ..., {x(n), y(n)}, for a given value of x.

$$P_n(x) = y(0) \cdot L_0(x) + \ldots + y(n) \cdot L_n(x) \quad , \quad L_k(x) = \prod_{\substack{j=0 \\ j \neq k}}^{n} \frac{x - x_j}{x_k - x_j} \quad , \quad k = 0,1,..,n$$

2. Create an input file **interpolation_data.ini** on the disk containing

```
5            // order of Pn(x)
0.0   1.0    //  x_int(0) , y_int(0)
1.0   3.0             .
3.0   -2.0            .
5.0  -5.0             .
8.0  -1.0             .
10.0  2.0    //  x_int(n) , y_int(n)
```

3. Write the main function which performs the following actions

- It opens and reads the file interpolation_data.ini
- It calculates the interpolation polynomial $P_n(x)$ at 101 equally spaced points in the closed interval [x(0),x(n)].
- It opens and writes the output disk file **interpolation_results.dat** in the following manner

x(0)	$P_n(x(0))$
x(0) + h	$P_n(x(0) + h)$
x(0) + 2h	$P_n(x(0) + 2h)$
....	
x(n)	$P_n(x(n))$

4. Make a plot of the tabulated function using Excel of Grapher. Check whether the conditions of interpolation are satisfied.

5. Write the function ***double Bad(double x)*** according to the formula $g(x) = \frac{1}{1 + 10x^2}$ and compute the interpolating polynomial defined for uniformly distributed nodes in the interval [-1,1]. Choose tabulation points as in the Section 3. Repeat calculations for n = 6, 20 and 40. Make plots of the function g(x) and the computed polynomials in Excel or Grapher. Repeat calculations using the Chebyshev nodes $x_k^{CH} = \cos\left(\frac{2*k+1}{2*n+2}\pi\right)$, $k = 0,1,..,n$.

6. Write the function double ***Newton***(double x_int[], double y_int[], int n, double x) implementing the Newton's method of polynomial interpolation. Note that the first stage of the calculations is to compute the vector of divided differences {y(0), y(0,1), y(0,1,2), …, y(0,..,n)}. This stage depends on the interpolation nodes, but it is independent of the choice of particular value of x. Think how to avoid multiple re-calculation of finite differences while tabulating the interpolating polynomial $P_n(x)$. Modify the main function so that it uses ***Newton*** instead of ***Lagrange*** and compare obtained plots.