Numerical Analysis HW8

1) Use the Lagrange interpolation process to obtain a polynomial of least degree that assumes these values:

X	0	2	3	4
Y	5	4	-7	10

4 data points gives us $P_3(x)$

$P_3(x) = y_0L_0 + y_1L_1 + y_2L_2 + y_3L_3$

$\Rightarrow P_3(x) = 5L_0 + 4L_1 - 7L_2 + 10L_3$

$\Rightarrow P_3(x) = 5 \cdot \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} \rightarrow 5 \cdot \frac{(x-2)(x-3)(x-4)}{\text{constant}}$

+ same process for L_1, L_2, & L_3.

In the end we get $P(x) = ax^3 + bx^2 + cx + d$

so $\boxed{3}$ is our answer.

---

2) Verify that the polynomials $p(x) = 5x^3 - 27x^2 + 45x - 21$ and $q(x) = x^4 - 5x^3 + 8x^2 - 5x + 3$ interpolate the data

X	1	2	3	4
Y	2	1	6	47

and explain why this doesn't violate the uniqueness part of the thm on existence of polynomial interpolation.

$5(1)^3 - 27(1)^2 + 45(1) - 21 = (1)^4 - 5(1)^3 + 8(1)^2 - 5(1) + 3 = 2$

$5(2)^3 - 27(2)^2 + 45(2) - 21 = (2)^4 - 5(2)^3 + 8(2)^2 - 5(2) + 3 = 1$

$5(3)^3 - 27(3)^2 + 45(3) - 21 = (3)^4 - 5(3)^3 + 8(3)^2 - 5(3) + 3 = 6$

$5(4)^3 - 27(4)^2 + 45(4) - 21 = (4)^4 - 5(4)^3 + 8(4)^2 - 5(4) + 3 = 47$

Verified

This doesn't violate the uniquness thm b/c the max order polynomial given 4 points is order 3. q(x) is order 4, hence it not qualifying to take the uniquness from the thm.

3) It is suspected that the table comes from a cubic polynomial. How can this be tested? Explain.

x	-2	1	0	1	2	3
y	1	4	11	16	13	-4

We can use Finite differences method.

x	y	$\Delta^1 y$	$\Delta^2 y$	$\Delta^3 y$
-2	1	3 $(y_2 - y_1)$	9 $(\Delta^1 y_2 - \Delta^1 y_1)$	-6 $(\Delta^2 y_2 - \Delta^2 y_1)$
-1	4	7	-2	-6
0	11	5	-8	-6
1	16	-3	-14	
2	13	-17		
3	-4			

At $\Delta^3 y$ all constants are same so we can conclude that the table does indeed come from a cubic polynomial. We can Furthe see this if we take derivatives from

$p_3(x) = ax^3 + bx^2 + cx + d \rightarrow p_3'(x) = 3ax^2 + 2bx + c \rightarrow p_3''(x) = 6ax + 2b \rightarrow p_3'''(x) = 6a.$

At the 3rd derivative we get a constant term which is similar to the finite differences method having 1 value at all points when $\Delta^3 y$ (constant).

4) There exists a unique polynomial $p(x)$ of degree 2 or less s.t. $p(0)=0$, $p(1)=1$, and $p'(\alpha)=2$ for any value of $\alpha \in [0,1]$ except one value of α say, α_0. Determine α_0, and give this polynomial for $\alpha \neq \alpha_0$.

$P_2(x) = ax^2+bx+c$ $\quad p(0) = a(0)^2+b(0)+c=0$, so $c=0$

$p(1) = a(1)^2+b(1)+c=1 \Rightarrow a+b=1$ $\quad P_2(x) = ax^2+bx$

$p_2'(x) = 2ax+b \rightarrow p_2'(\alpha) = 2a\alpha+b$ b/c $p'(\alpha)=2$, get

$2a\alpha+b=2$. $a+b=1$ becomes $b=1-a$, subsitute.

$2a\alpha+(1-a)=2 \Rightarrow 2a\alpha+1-a=2 \Rightarrow a(2\alpha-1)=1$

$\Rightarrow a=\frac{1}{2\alpha-1}$ if $\alpha=\frac{1}{2}$ then a is not defined so $\boxed{\alpha_0=\frac{1}{2}}$

$a=\frac{1}{2\alpha-1}$ & $b=1-a=1-\frac{1}{2\alpha-1}$, so $\boxed{\frac{x^2}{2\alpha-1}+x-\frac{x}{2\alpha-1}=P_2(x)}$

5) An interpolating polynomial $p(x)$ of degree 20 with uniform nodes is used to approximate e^{-x} on the interval $[0,2]$. Find an upper bound on $\max_{x\in[0,2]}|e^{-x}-p(x)|$ to determine how accurately this polynomial approximates e^{-x} on this interval.

$|f(x)-p(x)| \leq \frac{M}{4(n+1)} \cdot \left(\frac{b-a}{n}\right)^{n+1}$ $\quad$ where $\max_{[a,b]}|f^{(n+1)}(x)| \leq M$

degree 20 allows us to have $n=20$, $f^{(21)}(x) = -e^{-x}$, so

$\max_{[a,b]}|-e^{-x}| \leq M$ $\quad x=0$ gives us the max, $1=M$

Now $|f(x)-p(x)| \leq \frac{1}{4(n+1)} \cdot \left(\frac{b-a}{n}\right)^{n+1}$ plug in values $\frac{1}{4\cdot 21}\cdot\left(\frac{2}{20}\right)^{21}$

$\Rightarrow |f(x)-p(x)| \leq \frac{1}{84}\cdot(10^{-1})^{21} \Rightarrow \frac{1}{84}\cdot 10^{-21}$

so $\boxed{|e^{-x}-p(x)| \leq \frac{1}{84}\cdot 10^{-21}}$ (we use this equation b/c the nodes are in uniform)

6) Suppose $\cos(x)$ is approximated by interpolating polynomial of degree n, using $n+1$ nodes equally spaced nodes in the interval $[0,1]$. How accurate is the approximation? Express your answer in terms of n. How accurate is the approximation when $n=9$? For what values of n is the error less than 10^{-10}?

$\text{Max}_{[0,1]} |\cos^{(n+1)}(x)| \le M$

$$\overset{0}{\cos(x)} \to \overset{1}{-\sin(x)} \to \overset{2}{-\cos(x)} \to \overset{3}{\sin(x)} \to \overset{4}{\cos(x)}$$

b/c equally distributed nodes, $|\cos(x) - p(x)| \le \frac{M}{4(n+1)} \cdot \left(\frac{1}{n}\right)^{n+1}$

$\text{Max}_{[0,1]} |\cos(x)| = 1$, $\text{Max}_{[0,1]} |-\cos(x)| = 1$

$\text{Max}_{[0,1]} |\sin(x)| = 0.841$, $\text{Max}_{[0,1]} |-\sin(x)| = 0.841$ b/c everything must be less than equal to M, we will make $M=1$.

$\boxed{|\cos(x) - p(x)| \le \frac{1}{4(n+1)} \cdot \left(\frac{1}{n}\right)^{n+1}}$ in terms of n.

When $n=9$, $n+1=10$, which is an even derivative so 1 must be used for M. Regardless $\boxed{|\cos(x) - p(x)| \le \frac{1}{40} \cdot \left(\frac{1}{9}\right)^{10}}$ for $n=9$.

For $n=9$, $\frac{1}{4} \cdot 10^{-1} \cdot (9^{-1})^{10} \Rightarrow \frac{1}{4} \cdot 10^{-1} \cdot 9^{-10} < 10^{-10}$. Let's now solve

$\frac{1}{4(n+1)} \cdot \left(\frac{1}{n}\right)^{n+1} \le 10^{-10} \Rightarrow (n^{-1})^{n+1} \le 10^{-10} \cdot (4(n+1)) \Rightarrow n^{-n-1} \le 10^{-10} \cdot (4n+4)$

$\Rightarrow \frac{1}{n^{n+1}} \le 4 \cdot 10^{-10} + 4n \cdot 10^{-10} \Rightarrow \frac{1}{n^{n+1}} \le 4 \cdot 10^{-10} \cdot (n+1) \Rightarrow 1 \le 4 \cdot 10^{-10} \cdot (n+1) \cdot n^{n+1}$

$\Rightarrow \frac{10^{10}}{4} \le n^{n+2} + n^{n+1}$. We know $n=9$ has an error $< 10^{-10}$, so we can plug and chug in reverse order $n=8 < 10^{-10}$ b/c $\frac{10^{10}}{4} \le 8^{10} + 8^{9}$ is true. $n=7 \ge 10^{-10}$ b/c $\frac{10^{10}}{4} < 7^{9} + 7^{8}$ is false. Meaning

$\boxed{\text{When } n \ge 8 \text{, the error} < 10^{-10}}$.

Problem P1) Write pseudocode for a function that computes the divided differences. The input function consists of two arrays. The first array $X_{0:n}$ contains the n+1 values of x at which we interpolate the function f whose values at the interpolation points are given by the second array $Y_{0:n}$. The function returns the divided differences which are the coefficients of the Newton's polynomial.

```
def interpolate_coefs(x, y):
        n_plus_1 = len(x)
        make a n_plus_1 by n_plus_1 matrix called coef
        turn the first column of coef into y.
         Loop through values 1 to n_plus_1 - 1 (i)
             Inner Loop through values 0 to n_plus_1 - 1 - i (j)
                 coef[j][i] = (coef[j+1][i-1] - coef[j][i-1]) / (x[j+i] - x[j])

        return the first row of coef
```

Problem P2) Write pseudocode for a function that uses nested multiplication to evaluate Newton's polynomial at any point x. Note that this function also needs as its input the array $X_{0:n}$ from Problem P1 & and divided differences which are the coefficients of the Newton's polynomial.

```
def eval_polynomial(coef, x):
        n = len(x)-1
        total = coef[n]
        from 1 to n inclusive (i)
              total = coef[n-i] + x · total
        return total
```