МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
“Национальный исследовательский университет ИТМО”

**ФАКУЛЬТЕТ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**
**Кафедра проектирования и безопасности компьютерных систем**

# ЛАБОРАТОРНАЯ РАБОТА №2
по дисциплине
**"ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА"**

по теме:
**«Вычисление функций»**

**Выполнил:** Нгуен Ле Минь
**Группа:** N3251
**Преподаватель:** Гришенцев А.Ю.

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург
2021

# 1 Задание

### Задание 2.1: Схема Горнера

Разработать и написать программу реализующую схему Горнера для деления многочленов на двучлены и вычисляющую коэффициенты многочлена-частного и остаток. Результат выводить в виде многочлена-частного и остатка. Входные параметры: коэффициенты многочлена, величина $\xi$, являющаяся свободным членом делящего двучлена $(x - \xi)$. Программу снабдить проверкой правильности расчетов через умножение.

### Задание 2.2: Ряд Тейлора и Маклорена

Разработать и написать программу для вычисления логарифмической и гармонической функции с помощью рядов Тейлора (Маклорена), с возможностью выбора числа членов рядов вычисления функций, точность вычисления определить с помощью расчёта остаточного члена. Осуществить проверку вычислений: путем вычисления значений функции в точке и сравнение ошибки вычисления и величины остаточного члена. Построить график зависимости ошибки, точного значения и остаточного члена в зависимости от числа членов ряда.

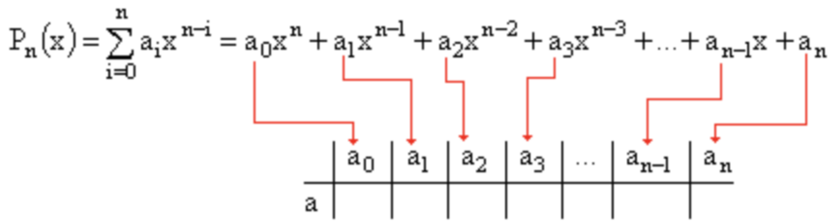
# 2 Теория

## 2.1 : Схема Горнера

Схема Горнера - способ деления многочлена

$P_n(x) = \sum_{i=0}^{n} a_i x^{n-1} = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + ... + a_{n-1} x + a_n$

на бином $x - a$. Работать придётся с таблицей, первая строка которой содержит коэффициенты заданного многочлена. Первым элементом второй строки будет число $a$, взятое из бинома $x - a$:




После деления многочлена $n$-ой степени на бином $x - a$, получим многочлен, степень которого на единицу меньше исходного, т.е. равна $n - 1$. Непосредственное применение схемы Горнера проще всего показать на примерах.

## 2.2 : Ряд Тейлора и Маклорена

Если функция $f(x)$ имеет непрерывные производные вплоть до $(n+1)$-го порядка, то ее можно разложить в степенной ряд по *формуле Тейлора*:

$$f(x) = \sum_{n=0}^{\infty} f^{(n)}(a) \frac{(x-a)^n}{n!} = f(a) + f'(a)(x-a) + \frac{f''(a)(x-a)^2}{2!} + \ldots + \frac{f^{(n)}(a)(x-a)^n}{n!} + R_n,$$

где $R_n$ − *остаточный член* в форме Лагранжа определяется выражением

$$R_n = \frac{f^{(n+1)}(\xi)(x-a)^{n+1}}{(n+1)!}, \quad a < \xi < x.$$

Если приведенное разложение сходится в некотором интервале $x$, т.е. $\lim_{n\to\infty} R_n = 0$, то оно называется *рядом Тейлора*, представляющим разложение функции $f(x)$ в точке $a$.

Если $a = 0$, то такое разложение называется *рядом Маклорена*:

$$f(x) = \sum_{n=0}^{\infty} f^{(n)}(0) \frac{x^n}{n!} = f(0) + f'(0)x + \frac{f''(0)x^2}{2!} + \ldots + \frac{f^{(n)}(0)x^n}{n!} + R_n.$$

Ряды Маклорена для вычисления логарифмической и гармонической функции $cos(x)$:

$cos(x) = 1 - \frac{1}{2!}x^2 + \frac{1}{4!}x^4 ... + \frac{(-1)^n}{(2n)!}x^{2n} + o(x^{2n})$

$ln(1 + x) = x - \frac{1}{2}x^2 + \frac{1}{3}x^3 - \frac{1}{4}x^4 + ... + \frac{(-1)^{n+1}}{n}x^n + o(x^n)$

# 3 Программы

## 2.1 : Схема Горнера

```
#include <iostream>
using namespace std;
int main() {
    cout << "Method Horner for division" << endl;
    cout << "a0*x^n + a1*x^(n-1) +....+an" << endl;
    int n;
    cout << "Value n:";
    cin >> n;
    while (n < 0) {
        cout << "Invalid value n.Value n >0 " << endl;
        cout << "Value n:";
        cin >> n;
    }
    auto  *pA = new float[n + 1];
    auto  *pB = new float[n + 1];
    cout << "Coefficient:" << endl;
    for (int i = 0; i <= n; i++)
    {
        cout << "a" << i << " =";
        cin >> pA[i];
    }
    int c;
    cout << "Divide x-c.\nValue c:";
    cin >> c;
    pB[0] = pA[0];
    cout << "Result:" << pB[0] << "*x^" << (n - 1);
    for (int i = 1; i <= n; i++)
    {
        pB[i] = pA[i] + c * pB[i - 1];
        if (i <= n - 2) cout << "+" << pB[i] << "*x^" << n - i - 1;
        else if (i == n - 1) cout << "+" << pB[i];
        else cout << "\nOctatki:" << pB[i];
    }

    float x;
    cout << "\nCheck result division with x= ";
    cin >> x;
    float value1 = 0;
    float value2 = 0;
    for (int i = 0; i <= n; i++) value1 += pA[i] * pow(x, n - i);
    for (int i = 0; i < n; i++) value2 += pB[i] * pow(x, n - i - 1);
    value2 = value2 * (x - c) + pB[n];
    if (value1 == value2) cout << "Result division :True";
    else cout << "Result division :False";
    return 0;
}

}
```

**Результаты программы :**

```
Method Horner for division
a0*x^n + a1*x^(n-1) +....+an
Value n:9
Coefficient:
a0 =2
a1 =3
a2 =4
a3 =
5
a4 =2
a5 =92
a6 =3
a7 =1
a8 =2
a9 =3
Divide x-c.
Value c:3
Result:2*x^8+9*x^7+31*x^6+98*x^5+296*x^4+980*x^3+2943*x^2+8830*x^1+26492
Octatki:79479
Check result division with x= 3
Result division :True
```

## 2.2 : Ряд Тейлора и Маклорена

```
#include<iostream>
using namespace std;
double factorial(int n)
{
    return (n == 1 || n == 0) ? 1 : n * factorial(n - 1)*1.0;
}
int main() {
    double m_ln = 0;
    double m_cos = 0;
    float x;
    int n;
    cout << "Value n in maclaurin series: ";
    cin >> n;
    cout << "Use maclaurin series to calculate cosx" << endl;
    cout << "x value: ";
    cin >> x;
    for (int i = 0; i <= n; i++) {
        m_cos += pow(-1, i)* pow(x, 2 * i) / factorial(2 * i)*1.0;
    }
    cout << "Calculate value cos x:" << m_cos << endl;
    cout << "True value cosx:" << cos(x) << endl;
    cout << "Error when use series: " << pow(-1, n + 1)* pow(x, 2 * (n + 1)) / factorial(2 * (n +
    cout << "Real error value:" << abs(cos(x) - m_cos) << endl;
    cout << "\nUse maclaurin series to calculate ln(x+1)" << endl;
    do {
        cout << "Input value\|x\|<1" << endl;
        cout << "x value: ";
        cin >> x;
    } while (abs(x) >= 1);

    for (int i = 1; i <= n; i++)
    {
        m_ln += pow(-1, (i + 1))*pow(x, i) / i * 1.0;
    }
```

```
    cout << "Calculate value ln(x+1):" << m_ln << endl;
    cout << "True value ln(x+1):" << log(x + 1) << endl;
    cout << "Error when use series: " << pow(-1, n + 2)*pow(x, n + 1) / (n + 1)*1.0 << endl;
    cout << "Real error value:" << abs(log(x + 1) - m_ln) << endl;
    return 0;
}
```

**Результаты программы**

```
Value n in maclaurin series: 3
Use maclaurin series to calculate cosx
x value: 2
Calculate value cos x:-0.422222
True value cosx:-0.416147
Error when use series: 0.00634921
Real error value:0.00607538

Use maclaurin series to calculate ln(x+1)
Input value|x|<1
x value: 3
Input value|x|<1
x value: 0.45
Calculate value ln(x+1):0.379125
True value ln(x+1):0.371564
Error when use series: -0.0102516
Real error value:0.00756141
```

# 4 Вывод

В ходе работе мы научились вычисления логарифмической и гармонической функции с помощью рядов, ввод значение число рядов и значение $x$,сравнение получит результат и реально величина, также расчета остаточного члена.Важно отметим при расчет значение функция $ln(x+1)$ только использовать рядов при $|x|< 1$,иначе ряд расходиться.