**Лабораторная работа №2**

**Интерполирование**

Выполнил студент 2 курса 3 группы ФПМИ

Сараев Владислав Максимович

Минск, 2020

**Теоретические сведения**

Дана функция . Необходимо построить интерполяционный многочлен данной функции в барицентрической форме на отрезке . Интерполирование необходимо провести по узлам () как по равноотстоящим узлам, так и по чебышевским. Для каждого построение необходимо построить графики получившихся приближений и экспериментально определить максимум-норму погрешности (максимум величины ).

Чебышевские узлы вычисляются по формуле:

где n – число точек, а a и b – концы отрезка, а котором проводится интерполяция.

Интерполяционный многочлен в барицентрической форме имеет вид:

Соответственно, для его нахождения необходимо предварительно вычислить только нормирующие множители , которые вычисляются по формуле:

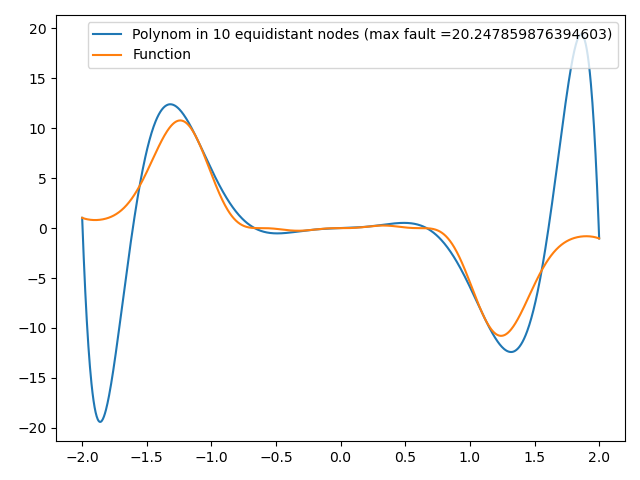
Вычислив один раз нормирующие множители , можно найти значение интерполяционного многочлена во всех интересующих точках.

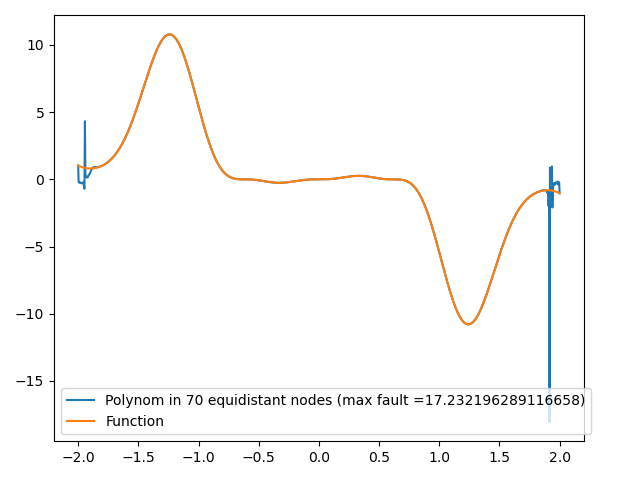
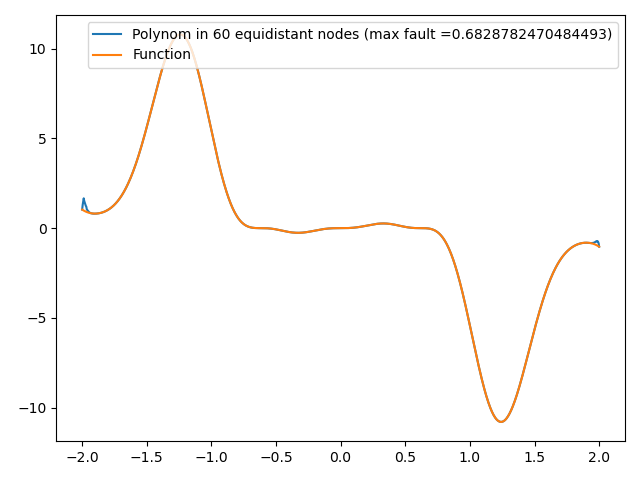
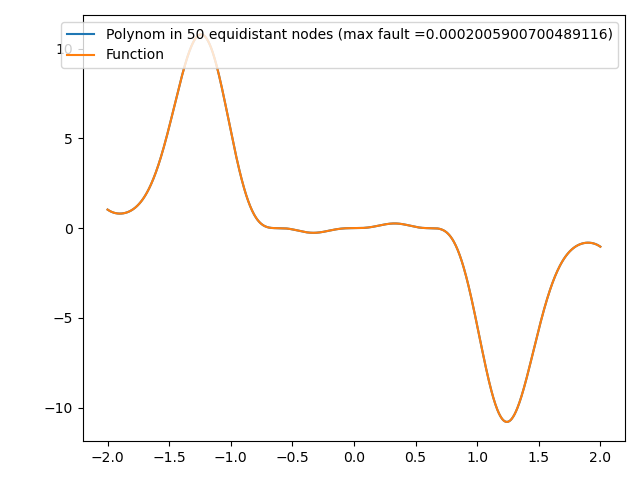
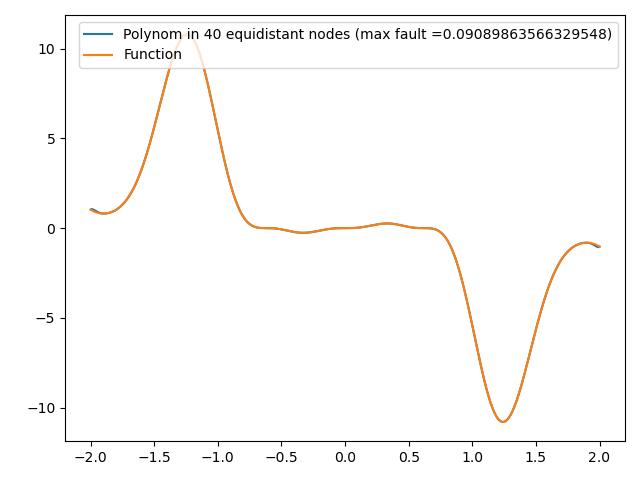
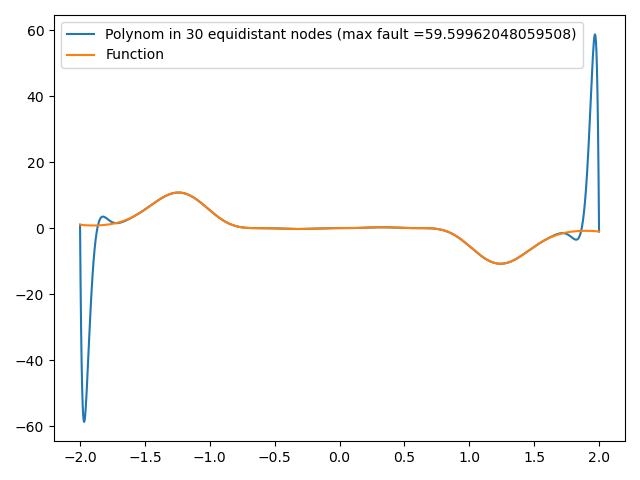
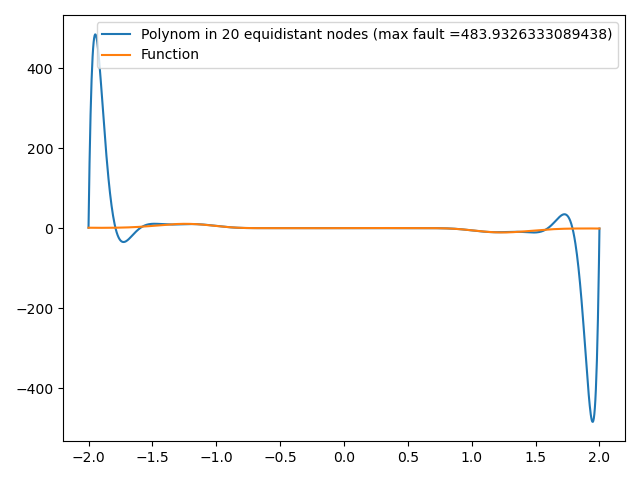
Соответственно, план выполнения задания следующий:

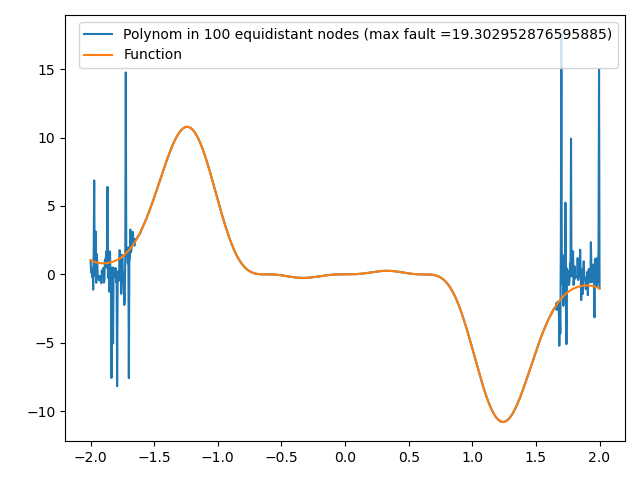
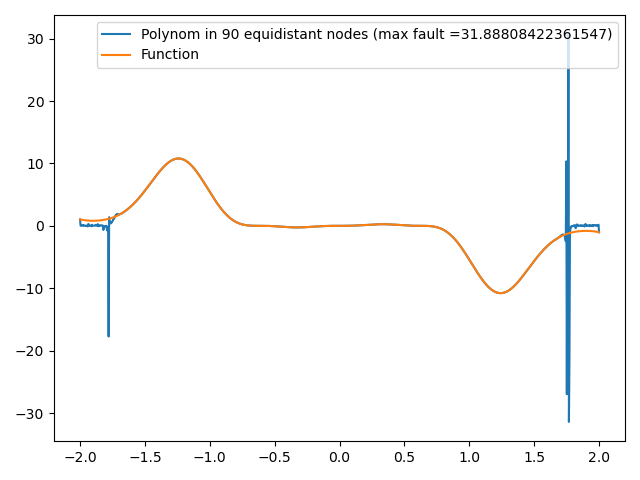
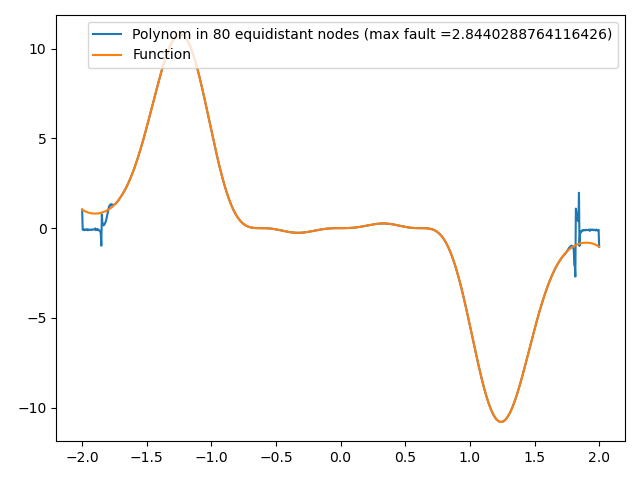
1. Разделить отрезок на нужное количество равноотстоящих узлов или вычислить необходимое количество чебышевских узлов по формуле (1).
2. Вычислить коэффициенты по формуле (3) и, соответственно построить многочлен (2).
3. По формуле (2) найти значения в искомых точках
4. Построить график и вычислить максимум-норму погрешности

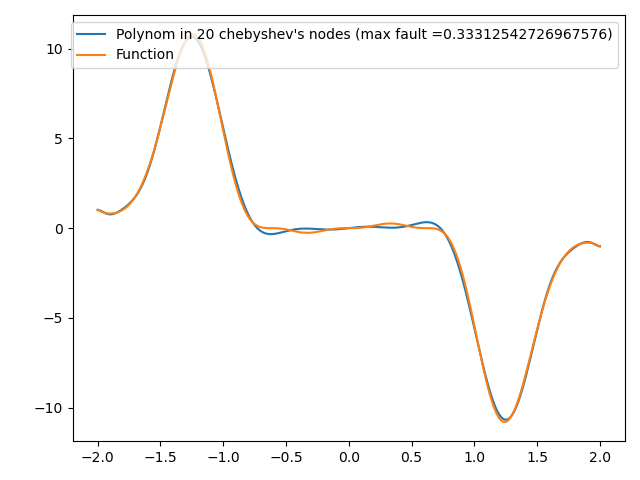
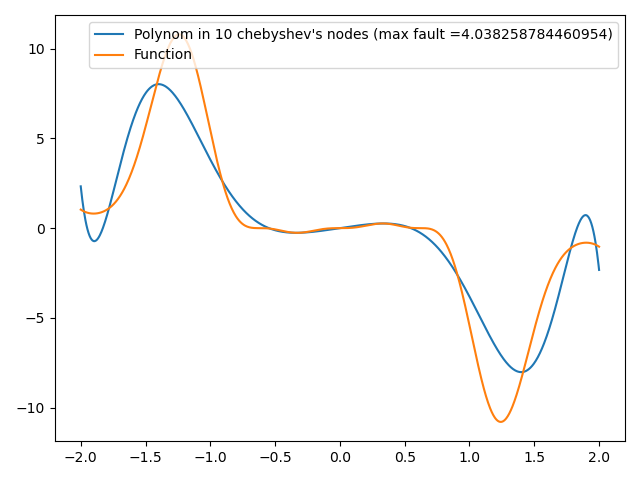
**Полученное решение**

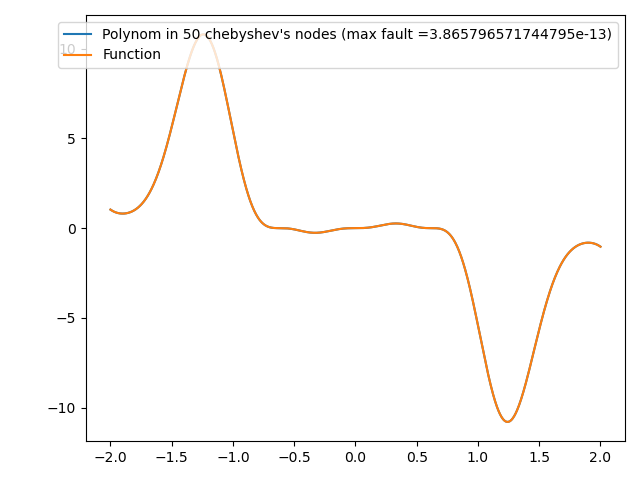
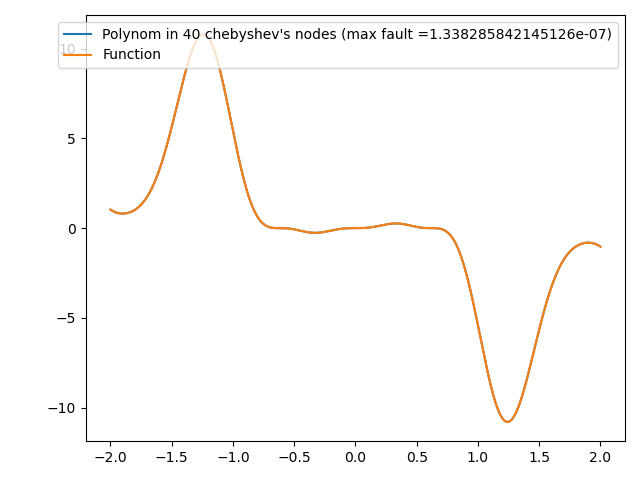
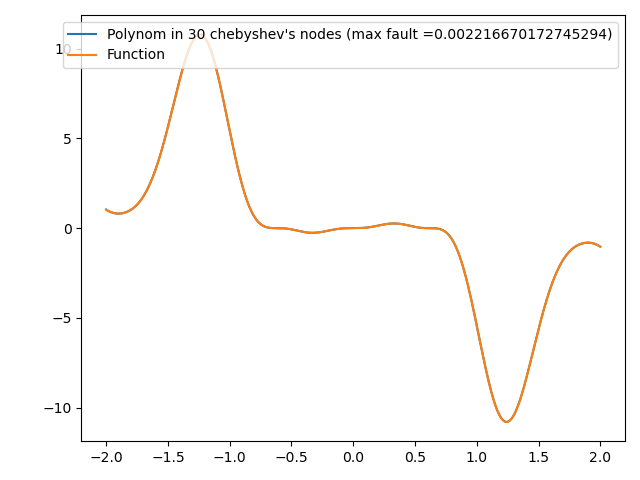
**Графики, полученные по равноотстоящим узлам**

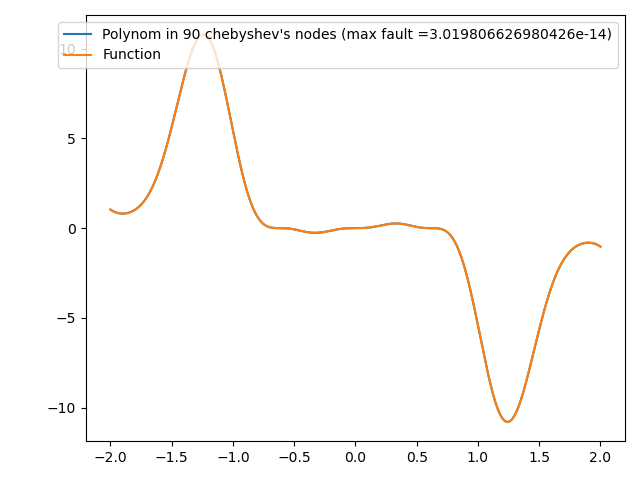
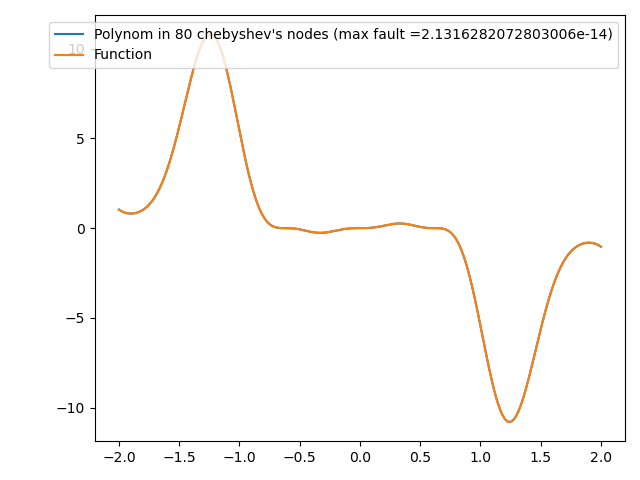
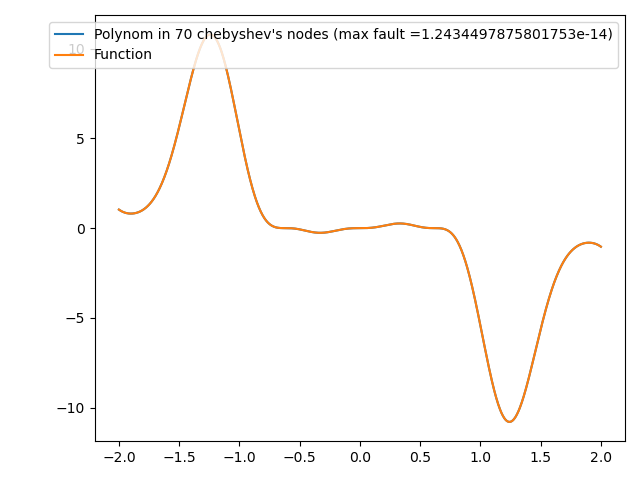
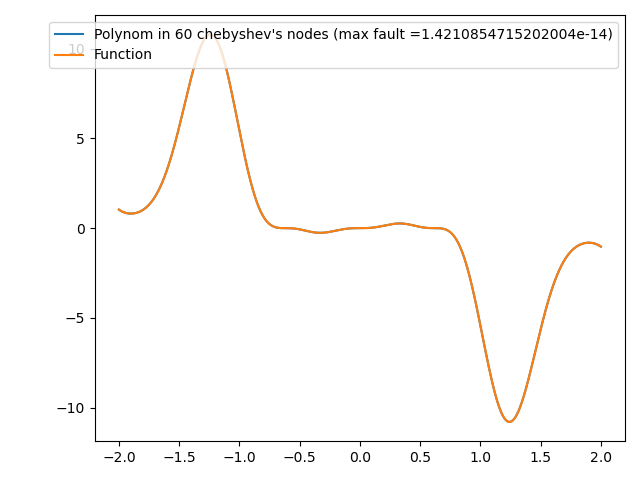
****

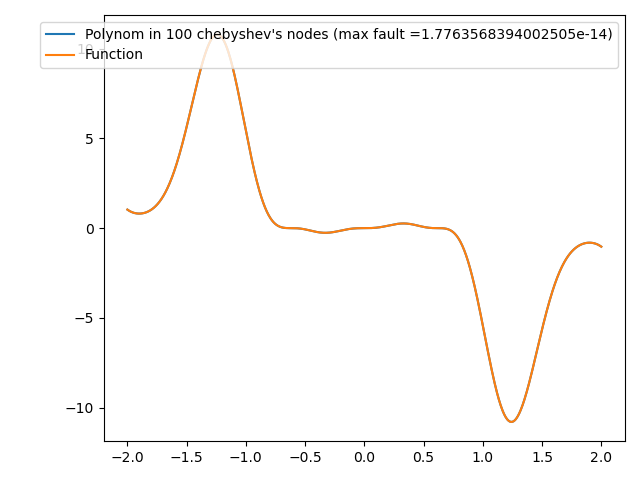
****

****

**Графики, полученные по чебышевским узлам**







**Нормы**

|  |  |  |
| --- | --- | --- |
| N | Норма (равноотстоящие узлы) | Норма (чебышевские узлы) |
| 10 | 20.247859876394603 | 4.038258784460954 |
| 20 | 483.9326333089438 | 0.33312542726967576 |
| 30 | 59.59962048059508 | 0.002216670172745294 |
| 40 | 0.09089863566329548 | 1.338285842145126e-07 |
| 50 | 0.0002005900700489116 | 3.865796571744795e-13 |
| 60 | 0.6828782470484493 | 1.4210854715202004e-14 |
| 70 | 17.232196289116658 | 1.2434497875801753e-14 |
| 80 | 2.8440288764116426 | 2.1316282072803006e-14 |
| 90 | 31.88808422361547 | 3.019806626980426e-14 |
| 100 | 19.302952876595885 | 1.7763568394002505e-14 |

**Исходный код**

**import** numpy **as** np

**from** math **import** sin**,** cos**,** pi

**import** matplotlib**.**pyplot **as** plot

**def** function**(**t**):**

**return** **(**sin**(**4 **\*** t**)** **-** t**)** **\*\*** 3

**def** polynom**(**t**):**

**global** x**,** y**,** c

**for** i **in** **range(len(**x**)):**

**if** t **==** x**[**i**]:**

**return** y**[**i**]**

den **=** **[**c**[**i**]** **/** **(**t **-** x**[**i**])** **for** i **in** **range(len(**c**))]**

**return** **sum(**den**[**i**]** **\*** y**[**i**]** **for** i **in** **range(len(**c**)))** **/** **sum(**den**)**

L **=** **-**2

R **=** 2

dots **=** np**.**linspace**(**L**,** R**,** 1000**)**

count\_of\_dots **=** **[**10 **\*** i **for** i **in** **range(**1**,** 11**)]**

type\_of\_nodes **=** "equidistant"

**for** l **in** **range(**2**):**

**for** i **in** **range(len(**count\_of\_dots**)):**

N **=** count\_of\_dots**[**i**]**

**if** type\_of\_nodes **==** "equidistant"**:**

x **=** np**.**linspace**(**L**,** R**,** N**)**

**else:**

x **=** np**.**empty**(**N**)**

**for** j **in** **range(**1**,** N **+** 1**):**

x**[**j **-** 1**]** **=** **(**L**+**R**)/**2 **+** **(**R**-**L**)/**2**\***cos**((**2**\***j**-**1**)\***pi**/(**2**\***N**))**

y **=** np**.**empty**(**N**)**

**for** j **in** **range(**0**,** N**):**

y**[**j**]** **=** function**(**x**[**j**])**

c **=** np**.**ones**(**N**)**

**for** j **in** **range(len(**x**)):**

**for** k **in** **range(len(**x**)):**

**if** j **!=** k**:**

c**[**j**]** **\*=** x**[**j**]** **-** x**[**k**]**

c**[**j**]** **=** 1 **/** c**[**j**]**

polynom\_dots **=** **[**polynom**(**t**)** **for** t **in** dots**]**

function\_dots **=** **[**function**(**t**)** **for** t **in** dots**]**

max\_fault **=** **max(abs(**polynom\_dots**[**i**]** **-** function\_dots**[**i**])** **for** i **in** **range(len(**polynom\_dots**)))**

**if** type\_of\_nodes **==** "equidistant"**:**

label\_info **=** "Polynom in " **+** **str(**N**)** **+** " equidistant nodes (max fault =" **+** **str(**max\_fault**)** **+** ")"

**else:**

label\_info **=** "Polynom in " **+** **str(**N**)** **+** " chebyshev's nodes (max fault =" **+** **str(**max\_fault**)** **+** ")"

plot**.**plot**(**dots**,** polynom\_dots**,** label**=**label\_info**)**

plot**.**plot**(**dots**,** function\_dots**,** label**=**"Function"**)**

plot**.**legend**()**

plot**.**show**()**

type\_of\_nodes **=** "chebyshev's"