Education and Culture DG
Lifelong Learning Programme

This project has been funded with support from the European Commission.
This publication reflects the views only of the author, and the Commission cannot be held for any use which may be made of the information contained therein.

Approximation of a function with Taylor-Polynom

Bettina-von-Arnim Gymnasium, Dormagen, Germany
Vihannin lukio, Vihanti, Finland

PC based Math
Project for High School Students

1 *Description of the Project (team1)*

Open/Close Print

The intention of our project was to get to know the basics of *Mathematica* and to use the program in the following.
For this we got a short introduction into *Mathematica* an continuatively an exercise we had to solve with our knowledge.
The main focus was on the approximation of functions with the aid of Taylorpolynoms.
To this we approximated at first simple polynoms and later a sinus function. Concluding we build the difference between the original and the approximated function.

2 *Brainstorming and Theory*

Open/Close Print

Brainstorming

Open / Close

- to approximate a sinus function we should use polynoms
- first approximation must be a parallel line to the x- axis
- to get a more exact approximation the degree of the polynominal function has to be higher

What Mathematics do we need:

Open / Close

1. In our project we worked with function, plots and tables.

Input ▹
```
Clear[f, x];
f[x_] = ⅇ^x
```

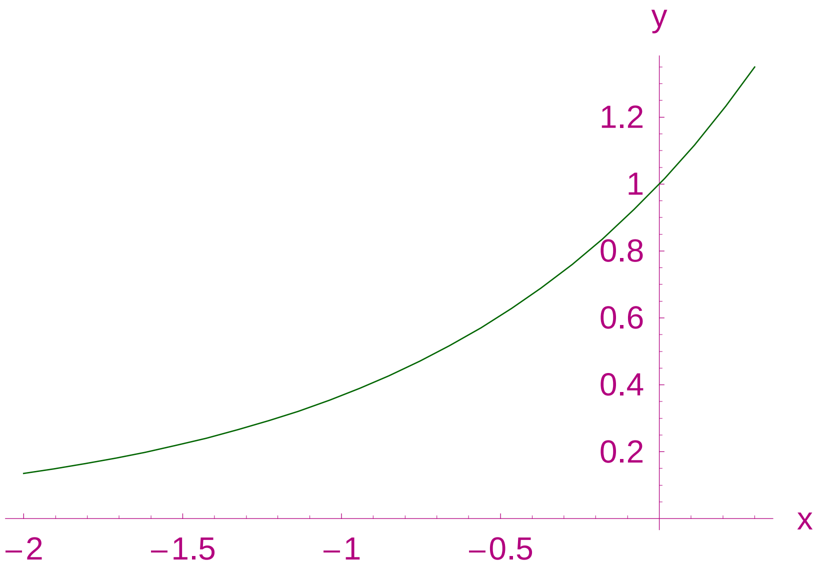
e^x

Input ▹
```
Clear[fTable, x];
start = -3;
stop = 3;
step = 0.5;

data = Table[{x, f[x] // N}, {x, start, stop, step}] // Chop;
MDShowTable[data, {"x", f[x] }];
```

x	e^x
−3	0.0497871
−2.5	0.082085
−2.	0.135335
−1.5	0.22313
−1.	0.367879
−0.5	0.606531
0	1.
0.5	1.64872
1.	2.71828
1.5	4.48169
2.	7.38906
2.5	12.1825
3.	20.0855

Input ▹
```
MDPlot[{f[x] }, {x , -2 , 1 × .3} ]
```



- Graphics -

2. Also we used derivation.

Input ▹
```
Clear[f, x];
f[x_] := x^5 + 5 x^4 + 0.7 x^3
f[x]
```

$0.7\,x^3 + 5\,x^4 + x^5$

```
f'[x]
f''[x]
```

```
f'''[x]
D[f[x] , {x, 4}]
```

$2.1\, x^2 + 20\, x^3 + 5\, x^4$

$4.2\, x + 60\, x^2 + 20\, x^3$

$4.2 + 120\, x + 60\, x^2$

$120 + 120\, x$

3 Developing Models

Open/Close Print

Approximation of a polynominal function:

Open / Close

1. We chose the polynominal function f[x]= $-x + 0.5\, x^2 + 0.5\, x^3$ and want to approximate the function on the point x0.

Input ▹
```
Clear[f, x];
x0 = -1
Expand[f[x_] = x (0.5 x + 1) (x - 1)]
```

-1

$-x + 0.5\, x^2 + 0.5\, x^3$

2. The name of our approximated function is p0.
At first the degree of p0 is 0, also the function is a parallel to x-axes.

Input ▹
```
Clear[p0, x];
p0[x_] = a
```

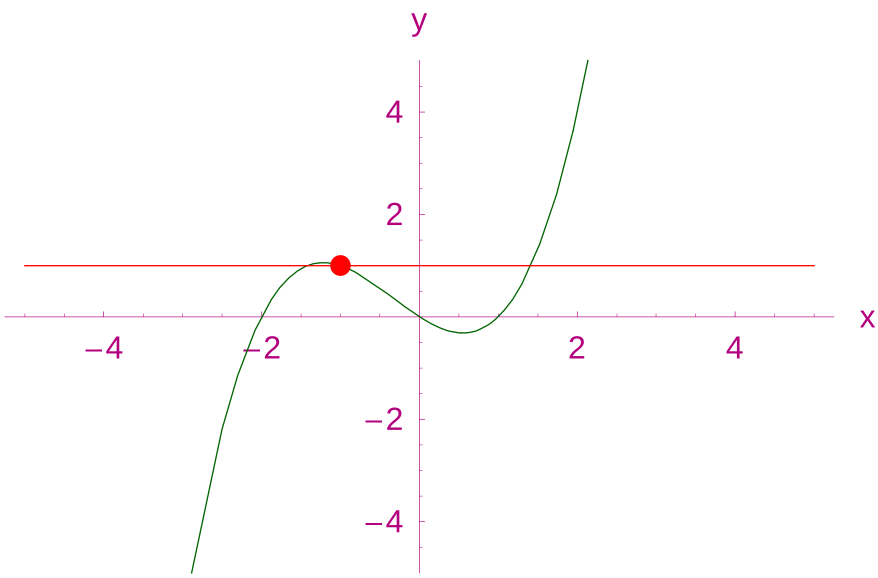
a

```
p0[x_] = 1
```

Input ▹

1

```
MDPlot[{f[x], p0[x]}, {x , -5 , 5 }, PlotRange → { -5, 5},
 Epilog → { Red, PointSize[0.025] , Point[{- 1, f[-1] }] } ]
```




```
- Graphics -
```

Input ▷
```
→ MDRealOnly;
Clear[a];
Solve[{ f[x0] == p0[x] }, { a} ]
```

```
{{}}
```

3. The next degree is 1. We want to find the tangent line with this degree.

Input ▷
```
Clear[p1, x];
Clear[a, b];
p1[x_] = a * x + b
```

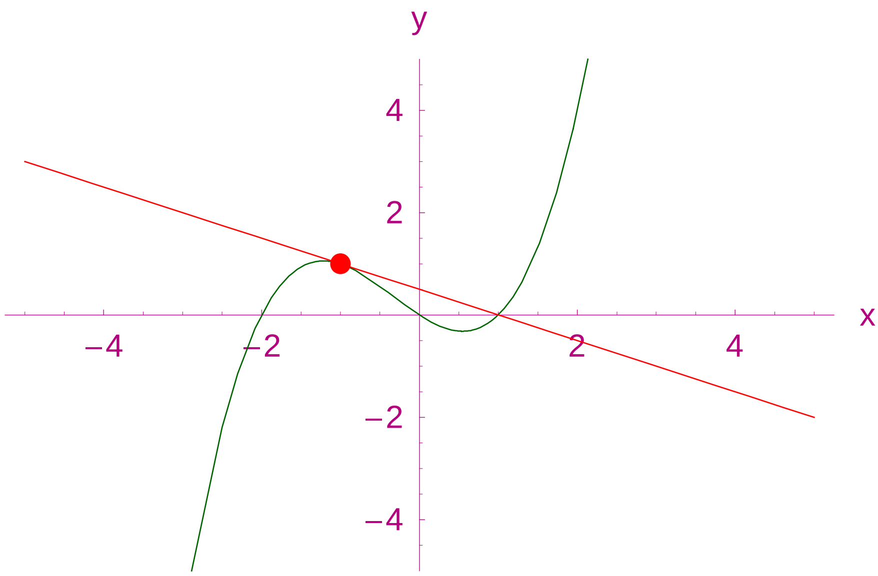
```
b + a x
```

Input ▷
```
→ MDRealOnly;
Clear[a, b];
Solve[{ f[x0] == p1[x0], f'[x0] == p1'[x0] }, { a, b } ]
p1[x_] = -0.5 * x + 0.5
```

```
{{a → -0.5, b → 0.5}}
```

```
0.5 - 0.5 x
```

Input ▷
```
Clear[a, b];
MDPlot[{f[x], p1[x]}, {x , -5 , 5 }, PlotRange → { -5, 5},
 Epilog → { Red, PointSize[0.025] , Point[{- 1, f[-1] }] } ]
Null
```



- Graphics -

4. Now we want to find the curveness with the degree 2.

Input ▷
```
Clear[p2, x];
Clear[a, b, c];
p2[x_] = a * x^2 + b * x + c
```

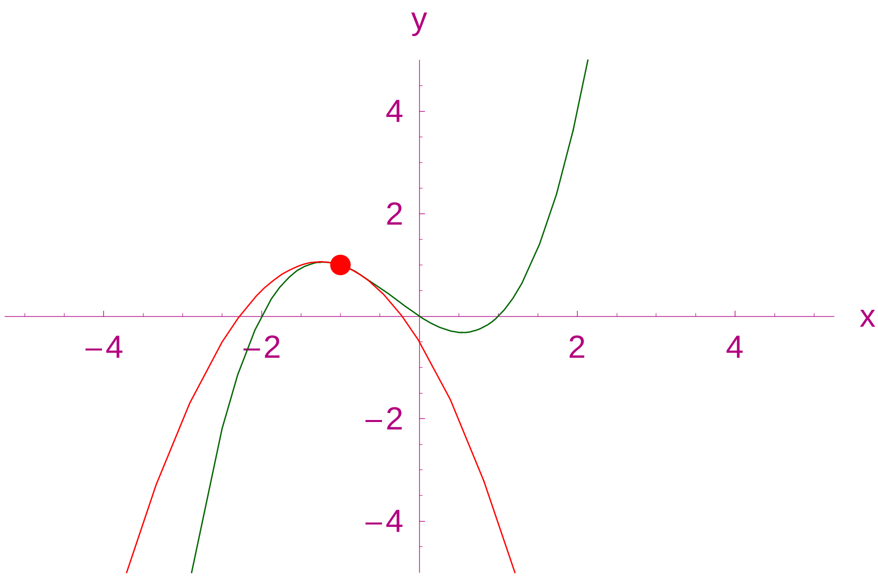
$c + b\,x + a\,x^2$

Input ▷
```
→ MDRealOnly;
Clear[a, b, c];
Solve[{ f[x0] == p2[x0],
  f'[x0] == p2'[x0], f''[x0] == p2''[x0] }, { a, b, c } ]
p2[x_] = -1 * x^2 -2.5 * x -0.5
```

{{a → -1., b → -2.5, c → -0.5}}

$-0.5 - 2.5\,x - x^2$

Input ▷
```
Clear[a, b, c];
MDPlot[{f[x], p2[x]}, {x , -5 , 5 }, PlotRange → { -5, 5},
 Epilog → { Red, PointSize[0.025] , Point[{- 1, f[-1] }] } ]
Null
```



- Graphics -

5. In the third degree the functions are the same, because f[x] has the degree 3.

Input ▷
```
Clear[p3, x];
Clear[a, b, c, d]
p3[x_] = a*x^3 + b*x^2 + c*x + d
```

$d + c\,x + b\,x^2 + a\,x^3$

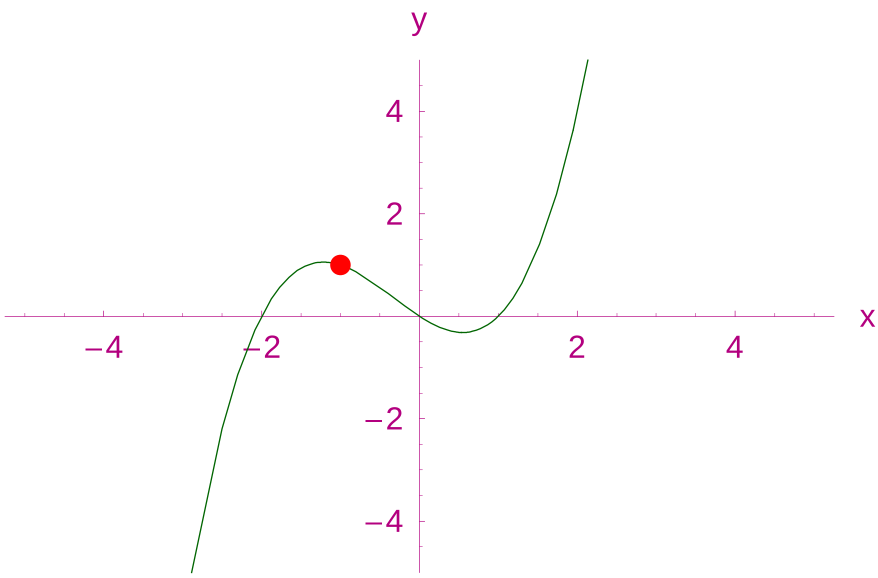
Input ▷
```
→ MDRealOnly;
Clear[a, b, c, d];
Solve[{ f[x0] == p3[x0], f'[x0] == p3'[x0],
  f''[x0] == p3''[x0], f'''[x0] == p3'''[x0] }, { a, b, c, d } ]
```

```
p3[x0] = 0.5*x^3 +0.5*x^2 -1*x - 1.1102230246251565`*^-16
```

$\{\{a \to 0.5,\ b \to 0.5,\ c \to -1.,\ d \to -1.11022 \times 10^{-16}\}\}$

$-1.11022 \times 10^{-16} - x + 0.5\,x^2 + 0.5\,x^3$

Input ▷
```
Clear[a, b, c, d];
MDPlot[{f[x], p3[x]}, {x , -5 , 5 }, PlotRange → { -5, 5},
 Epilog → { Red, PointSize[0.025] , Point[{- 1, f[-1] }] } ]
Null
```



- Graphics -

Approximation of a sinus function:

Öffnen / Schließen

6. The function we want to approximate is f[x]=Sin[x].

Input ▹
```
Clear[f, x];
x0 = 1
f[x_] := Sin[x]
```

1

Input ▹
```
p0[x_] = a
```

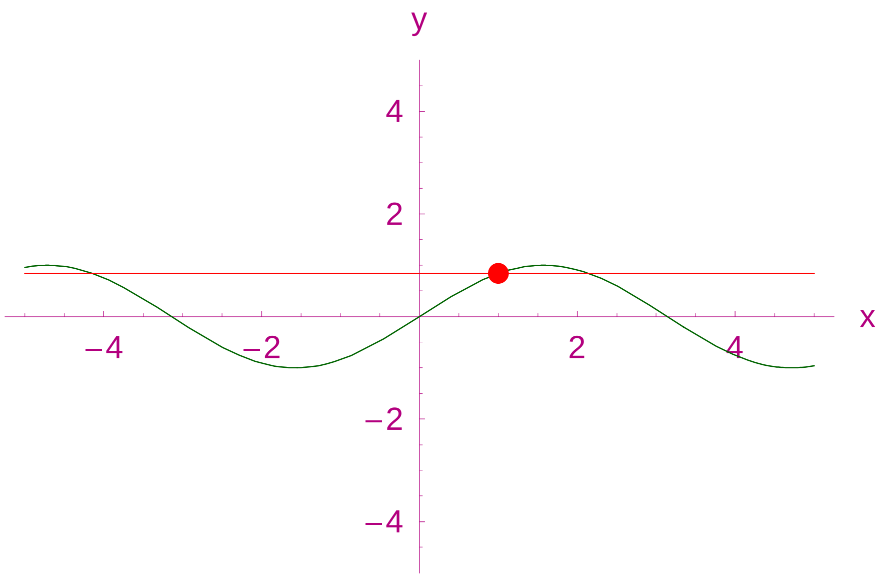
a

Input ▹
```
→ MDRealOnly;
Clear[a];
Solve[{ f[x0] == p0[x] }, { a} ]
p0[x] = Sin[1]
```

{{a → Sin[1]}}

Sin[1]

Input ▹
```
MDPlot[{f[x], p0[x]}, {x , -5 , 5 }, PlotRange → { -5, 5},
 Epilog → { Red, PointSize[0.025] , Point[{ 1, f[1] }] } ]
```




```
- Graphics -
```

Input ▹
```
Clear[p1, x];
Clear[a, b];
p1[x_] = a * x + b
```

```
b + a x
```

Input ▹
```
Solve[{ f[x0] == p1[x0], f'[x0] == p1'[x0] }, { a, b } ]
p1[x] = Cos[1] * x - Cos[1] + Sin[1]
a = f'[1]
b = f''[1] + f[1]
```

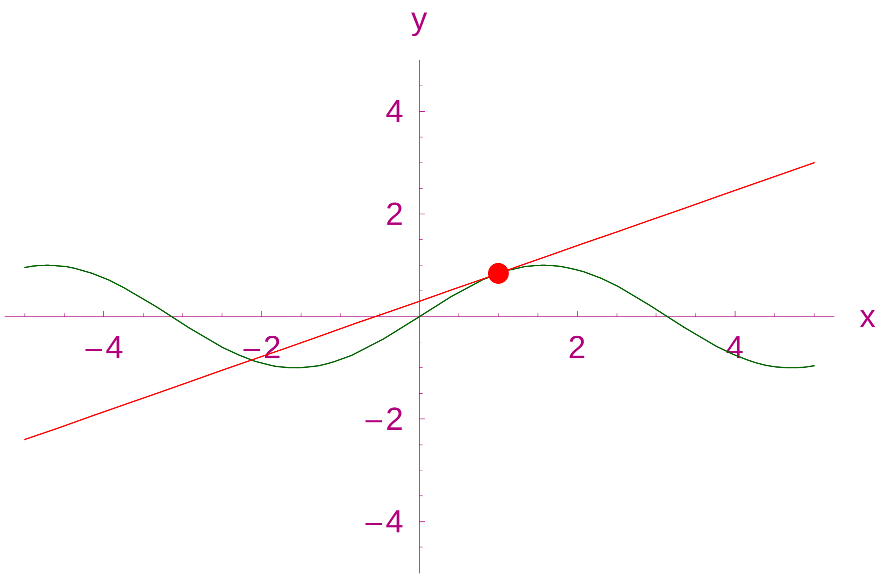
```
{{a → Cos[1], b → -Cos[1] + Sin[1]}}

-Cos[1] + x Cos[1] + Sin[1]

Cos[1]

0
```

Input ▹
```
MDPlot[{f[x], p1[x]}, {x , -5 , 5 }, PlotRange → { -5, 5},
 Epilog → { Red, PointSize[0.025] , Point[{ 1, f[1] }] } ]
```




```
- Graphics -
```

Input ▹
```
Clear[p2, x];
Clear[a, b, c];
p2[x_] := a*x^2 + b*x + c
```

Input ▹
```
→ MDRealOnly;
Clear[a, b, c];
{ a, b, c } = { a, b, c } /.
   Flatten@Solve[{ f[x0] == p2[x0], f'[x0] == p2'[x0],
      f''[x0] == p2''[x0] }, { a, b, c } ];
p2[x]
Null
```

$$-\text{Cos}[1] + \frac{\text{Sin}[1]}{2} - \frac{1}{2} x^2 \,\text{Sin}[1] + x\,(\text{Cos}[1] + \text{Sin}[1])$$

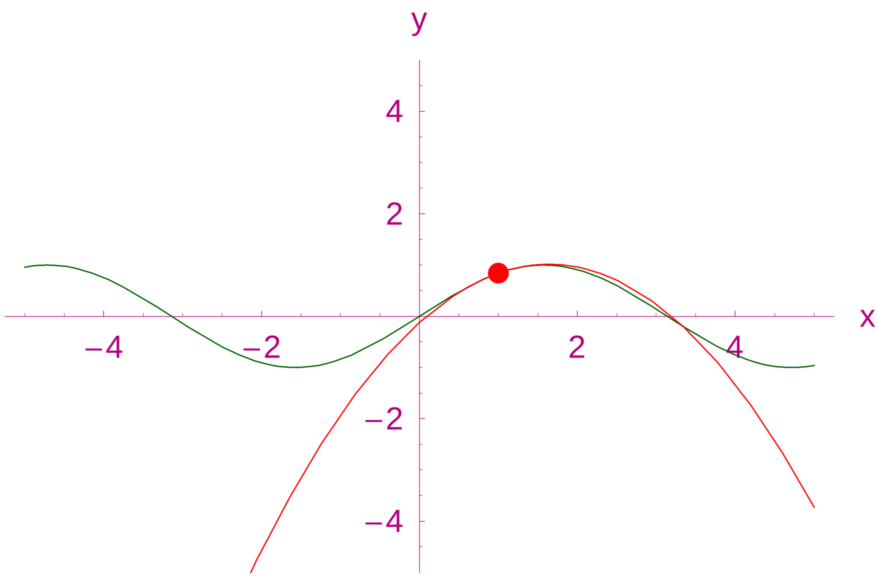
$$a = \frac{f''[1]}{2}$$

$$b = f'[1] + f[1]$$

$$c = f'''[1] + \frac{f[1]}{2}$$

Null

Input ▹
```
MDPlot[{f[x], p2[x]}, {x , -5 , 5 }, PlotRange → { -5, 5},
  Epilog → { Red, PointSize[0.025] , Point[{ 1, f[1] }] } ];
```



Input ▹
```
Clear[p3, x];
Clear[a, b, c, d]
p3[x_] := a*x^3 + b*x^2 + c*x + d
p3[x]
```

$d + c\,x + b\,x^2 + a\,x^3$

Input ▹
```
→ MDRealOnly;
Clear[a, b, c, d];
{ a, b, c, d } = { a, b, c, d } /. Flatten@
    Solve[{ f[x0] == p3[x0], f'[x0] == p3'[x0], f''[x0] ==
       p3''[x0], f'''[x0] == p3'''[x0] }, { a, b, c, d } ];
p3[x]
Null
```

$$-\frac{1}{6} x^3 \text{Cos}[1] + \frac{1}{2} x^2 (\text{Cos}[1] - \text{Sin}[1]) + \frac{1}{2} x (\text{Cos}[1] + 2\,\text{Sin}[1]) + \frac{1}{6} (-5\,\text{Cos}[1] + 3\,\text{Sin}[1])$$

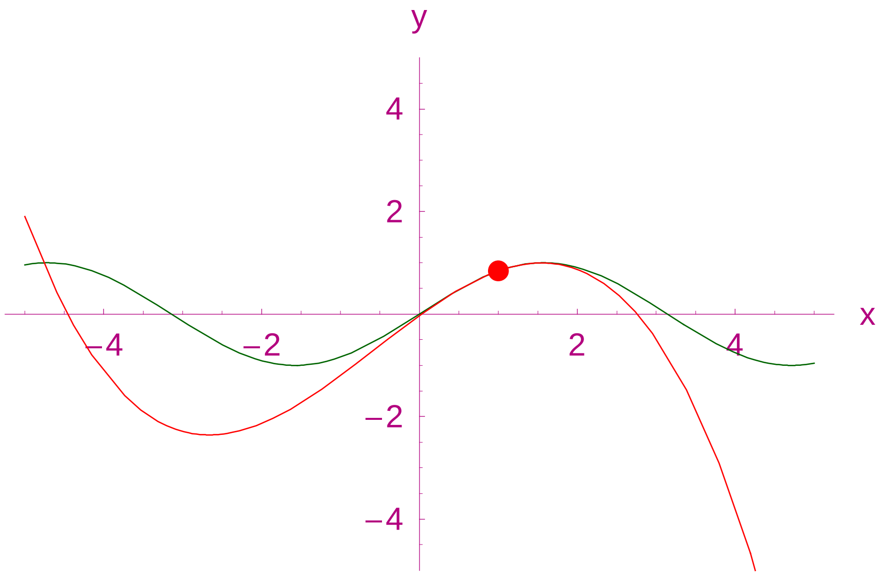
$a=\dfrac{f'''[1]}{6}$

$b=\dfrac{1}{2}(f'[1]+f''[1])$

$c=\dfrac{1}{2}(f'[1]+2*f[1])$

$d=\dfrac{1}{6}(5*f'''[1]+3*f[1])$

```
MDPlot[{f[x], p3[x]}, {x , -5 , 5 }, PlotRange → { -5, 5},
 Epilog → { Red, PointSize[0.025] , Point[{ 1, f[1] }] } ]
```



- Graphics -

Input ▷
```
Clear[p4, x];
Clear[a, b, c, d, e]
p4[x_] := a * x^4 + b * x^3 + c * x^2 + d * x + e
p4[x]
```

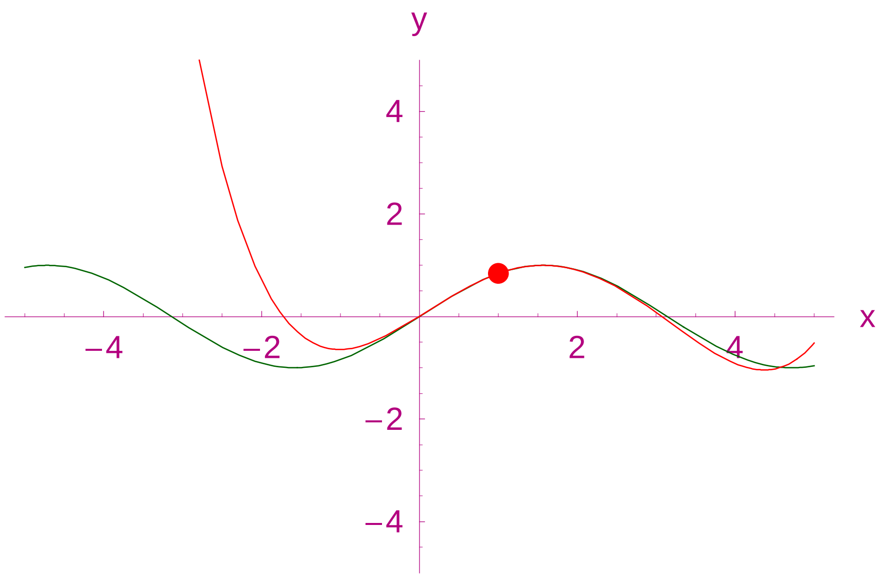
$e + d\,x + c\,x^2 + b\,x^3 + a\,x^4$

Input ▷
```
→ MDRealOnly;
Clear[a, b, c, d, e];
{ a, b, c, d, e } = { a, b, c, d, e } /.
   Flatten@Solve[{ f[x0] == p4[x0], f'[x0] == p4'[x0],
      f''[x0] == p4''[x0], f'''[x0] == p4'''[x0],
      f''''[x0] == p4''''[x0] }, { a, b, c, d, e } ];
```

$$p4[x] = \frac{Sin[1]}{24} * x\text{^}4 + \frac{1}{6}(-Cos[1] - Sin[1]) * x\text{^}3 + \frac{1}{4}(2\,Cos[1] - Sin[1]) * x\text{^}2 + \frac{1}{6}(3\,Cos[1] + 5\,Sin[1]) * x + \frac{1}{24}(-20\,Cos[1] + 13\,Sin[1])$$

$$\frac{1}{6}x^3(-Cos[1] - Sin[1]) + \frac{1}{4}x^2(2\,Cos[1] - Sin[1]) + \frac{1}{24}x^4\,Sin[1] + \frac{1}{6}x\,(3\,Cos[1] + 5\,Sin[1]) + \frac{1}{24}(-20\,Cos[1] + 13\,Sin[1])$$

Input ▷
```
MDPlot[{f[x], p4[x]}, {x , -5 , 5 }, PlotRange → { -5, 5},
  Epilog → { Red, PointSize[0.025] , Point[{ 1, f[1] }] } ];
```



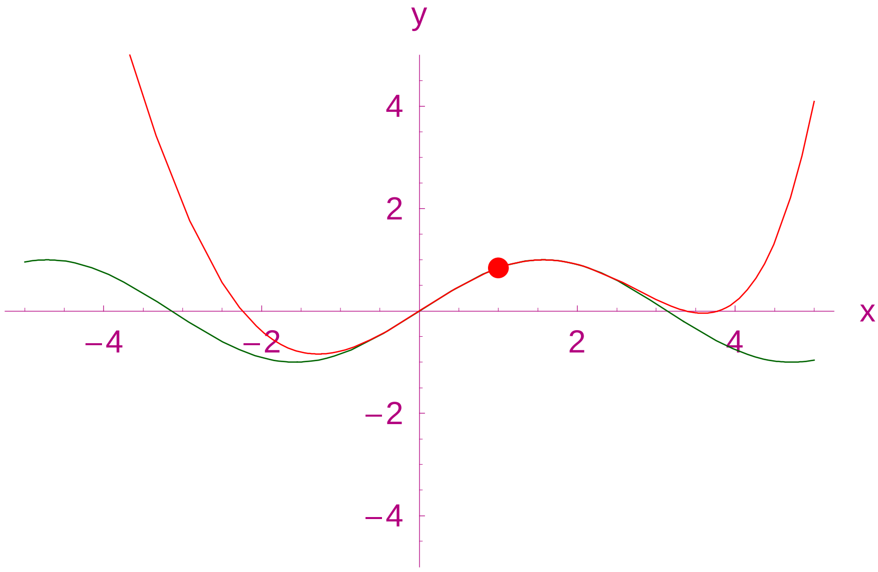
Input ▷
```
Clear[p5, x];
Clear[a, b, c, d, e, g]
p5[x_] := a*x^5 + b*x^4 + c*x^3 + d*x^2 + e*x + g
```

```
→ MDRealOnly;
Clear[a, b, c, d, e, g];
```

Input ▷
```
{a, b, c, d, e, g} = {a, b, c, d, e, g} /. Flatten@
    Solve[{f[x0] == p5[x0], f'[x0] == p5'[x0],
      f''[x0] == p5''[x0], f'''[x0] == p5'''[x0],
      f''''[x0] == p5''''[x0], (D[f[x], {x, 5}] /. x -> 1) ==
       (D[p5[x], {x, 5}] /. x -> 1)}, {a, b, c, d, e, g}];
p5[x]
Null
```

$$\frac{1}{120} x^5 \text{Cos}[1] + \frac{1}{12} x^2 (5\,\text{Cos}[1] - 3\,\text{Sin}[1]) + \frac{1}{12} x^3 (-\text{Cos}[1] - 2\,\text{Sin}[1]) + \frac{1}{24} x^4 (-\text{Cos}[1] + \text{Sin}[1]) + \frac{1}{24} x\,(13\,\text{Cos}[1] + 20\,\text{Sin}[1]) + \frac{1}{120} (-101\,\text{Cos}[1] + 65\,\text{Sin}[1])$$

Input ▷
```
MDPlot[{f[x], p5[x]}, {x, -5, 5}, PlotRange → {-5, 5},
 Epilog → {Red, PointSize[0.025], Point[{1, f[1]}]}]
```



- Graphics -

Difference between the original function and the approximated function:

Öffnen / Schließen

7. Now we want to know the difference between the approximation and the function, because they are only in one point identical.

```
Clear[f, x, n, k, c];
c = 2
x0 = 1
n = 1
k = 1
```

Input ▷
```
f[x_] := Sin[x]
```

$$t[z\_] := \sum_{n=0}^{k} \frac{D[f[x], \{x, n\}] \;/.\; x \to x_0}{n\,!} (x - x_0)^n \;/.\; x \to z$$

```
t[x]
```

Input ▷
```
Abs[f[c] - t[c]] // N
```

This is our result of the difference.

Input ▷
```
f[c]
```

Input ▷
```
t[2]
```

8. Now we have to choose the higher degree of the polynominal function.

```
Clear[f, x, n, k, c];
c = 2
x0 = 1
n = 2
k = 2
f[x_] := Sin[x]
```

$$t[z\_] := \sum_{n=0}^{k} \frac{D[f[x], \{x, n\}] \,/.\, x \to x_0}{n!} (x - x_0)^n \,/.\, x \to z$$

```
t[x]
```

Input ▹ `Abs[f[c] - t[c]] // N`

The difference between the two functions is lower when the degree is higher.

9. Now we have to choose the higher degree of the polynominal function again.

```
Clear[f, x, n, k, c];
c = 2
x0 = 1
n = 3
k = 3
```

Input ▹ `f[x_] := Sin[x]`

$$t[z\_] := \sum_{n=0}^{k} \frac{D[f[x], \{x, n\}] \,/.\, x \to x_0}{n!} (x - x_0)^n \,/.\, x \to z$$

```
t[x]
```

Input ▹ `Abs[f[c] - t[c]] // N`

10. Now we chose a really high degree of the polynominal function.

```
Clear[f, x, n, k, c];
c = 2
x0 = 1
n = 100
k = 100
```

Input ▹ `f[x_] := Sin[x]`

```
t[z_] := Sum[(D[f[x], {x, n}] /. x -> x0)/n! (x - x0)^n, {n, 0, k}] /. x -> z
```

$$t[z\_] := \sum_{n=0}^{k} \frac{D[f[x], \{x, n\}] \;/.\; x \to x_0}{n!} (x - x_0)^n \;/.\; x \to z$$

```
t[x]
```

Input ▹
```
Abs[f[c] - t[c]] // N
```

Now the degree is high enough! ☺

```
Clear[f, x, n, k, c];
c = 2
```

```
x0 = 1
```

Input ▹

```
f[x_] := Sin[x]
```

$$t[z\_] := \sum_{n=0}^{n} \frac{D[f[x], \{x, n\}] \;/.\; x \to x_0}{n!} (x - x_0)^n \;/.\; x \to z$$

Input ▹
```
g[n_] := Abs[f[c] - t[c]] // N
```

```
start = 0;
stop = 10;
step  = 1;
```

Input ▹

```
data = Table[{n, g[n] // N}, {n, start, stop, step}] // Chop;
MDShowTable[data,
  {"n is the degree of the approximated function",
   "Difference from f[c] and t[c]" }];
```

n is the degree of the approximated function	Difference from f[c] a
0	0.0678264
1	0.472476
2	0.0517404
3	0.03831
4	0.00324872
5	0.0012538
6	0.0000850877
7	0.0000221152
8	1.24534×10^{-6}
9	2.43589×10^{-7}
10	1.17022×10^{-8}

4

Our Team and Experiences

Open / Close Print

Jere Junttila (Vihanti, Finland)
Miina Honkala (Vihanti, Finland)
Angela Brück (Dormagen, Germany)
Frederike Franken (Dormagen, Germany)

Open / Close

Our experiences with the project:

Open / Close

In the time of our project we made a lot of new experiences.
For example our journeys to Finland and accordingly to Germany where we lived in host families were a highlight.
During our visit in Vihanti/Dormagen we worked on our *Mathematica* project. It was very exciting to work with *Mathematica* instead of the ordinary mathematic lessons. We learned something about the handling with the program and solved our exercise successful. It was a great time for all of us...

1

Description of the Project (team2)

Open / Close Print

The aim of our project was to find a way to define a Taylorpolynominal.
With the help of the approximation of polynominals and the sinus-function, the difference between Taylorpolynominals and the original function can be calculated.

2 Brainstorming and Theory

Open/Close Print

Brainstorming

Open / Close

To get a Taylorpolynominal you need to start with the approximation of polynominal function of a low degree.
The results of these approximations can be assigned to the approximation of any function included the sinus-function.
This will lead us to find a way to define a Taylorpolynominal and so it is the basic idea of forming a Taylorpolynominal.

What Mathematics do we need

Open / Close

We made use of folowing mathematics:
- addition, subtraction, multiplication and division
- derivation

3 Developing Models

Open/Close Print

Part 1.1: Solve of the approximation of a polynominal function

Open / Close

Input ▹
```
Clear[f, x0, x];
f[x_] := 3 x^3 - 5 x^2 + 7
x0 = 2;
```

Derivation

Input ▹
```
f[x0]
```

Input ▹
```
f'[x0]
```

Input ▹
```
f''[x0]
```

Input ▹
```
f'''[x0]
```

Input ▹
```
f''''[x0]
```

Solve p0

Input ▹
```
Clear[p0, x, a];
p0[x_] := a
```

```
Clear[a]
```

Input ▹
```
Switch → Pure Solve;
Clear[x];
{a} = {a} /.
   Flatten[Solve[{ f[x0] == p0[x0] }, {a} ]];
```

```
p0[x]
```

Graph f[x] and p0[x]

Input ▹
```
MDPlot[{f[x] , p0[x]}, {x , -5, 5}, PlotRange → { -1, 12 } ]
```

This is just a straight line through the point p0.

Solve p1

Input ▹
```
Clear[p1, x, b, c];
p1[x_] := b * x + c
```

```
Clear[b, c]
```

Input ▹
```
Switch → Pure Solve;
Clear[x];
{b, c} = {b, c} /.
   Flatten[
    Solve[{ f[x0] == p1[x0], f'[x0] == p1'[x0] }, { b, c } ]];
```

```
p1[x]
```

Graph f[x] and p1[x]

Input ▹
```
MDPlot[{f[x] , p1[x]}, {x , -5, 5}, PlotRange → { -1, 12 } ]
```

This is the tangent line and it shows the gradient at the point p0.

Solve p2

Input ▹
```
Clear[p2, x, d, e, i];
p2[x_] := d * x^2 + e * x + i
```

```
Clear[d, e, i]
```

Input ▹
```
Switch → Pure Solve;
Clear[x];
{d, e, i} = {d, e, i} /.
   Flatten[Solve[{ f[x0] == p2[x0],
      f'[x0] == p2'[x0], f''[x0] == p2''[x0] }, { d, e, i }]];
```

```
p2[x]
```

Graph f[x] and p2[x]

Input ▹
```
MDPlot[{f[x] , p2[x]}, {x , -5, 5}, PlotRange → { -1, 12 } ]
```

This graph shows the curvature at the point p0.

Solve p3

Input ▹
```
Clear[p3, x, j, k1, l, m];
p3[x_] := j * x^3 + m2 * x^2 + l * x + m
```

```
Clear[j, m2, l, m]
```

Input ▹
```
Switch → Pure Solve;
Clear[x];
{j, m2, l, m} = {j, m2, l, m} /.
   Flatten[Solve[
     { f[x0] == p3[x0], f'[x0] == p3'[x0], f''[x0] == p3''[x0],
      f'''[x0] == p3'''[x0] }, { j, m2, l, m } ]];
```

```
p3[x]
```

Graph f[x] and p3[x]

Input ▹
```
MDPlot[{f[x], p3[x]}, {x, -5, 5}, PlotRange → {-1, 12}]
```

The function p3 has the same degree as the original function has. The approximation for the function f(x) is completed now.

Part 1.2: Movie

Open / Close

This movie shows that the approximation get more and more accurate with an increasing of the degree of the polynom.

```
More... ;
Clear[j, y, a, k];
great[j_, y_] :=
```

```
Expand[Sum[(D[f[x], {x, ß}] /. x → x0)/ß! * (x - x0)^ß, {ß, 0, y}]] /. x → j // N ;
(* enter your f *)

MDMovie[ MDPlot[{f[x] /. x → j, great[j, y]},
  {j, -4, 6}, PlotRange → {0, 12},
  Background → $MDNotebookBackground, PlotLabel →
   "y = " <> ToString[NumberForm[y, {5, 3}]] <> "\n"], {y, 0, 3, 1}]
```

Part 1.3: Comparison of the coefficients of the approximations p0→ p5 of the function f

Open / Close

p0

Input ▹
```
f[x0]
p0[x]
```

The first approximation of f is just the function value of f at the point x0.

p1

Input ▹
```
f'[x0]
p1[x]
p1[x] - p0[x]
```

Input ▹
```
Expand[f'[x0] * (x - x0)]
```

The result of p1-p0 is the product of the first derivation of f at the x value x0 and (x-x0). That is = f'[x0]*(x-x0). We subtract both approximations because there is the same constraint in both orders, which says that the function value of p0 and p1 at x0 has to be the same. If we add p0 we ought to get the function we got when we solved the function the way we did in chapter 1.1 (f[x0]+f'[x0]*(x-x0)).

Input ▹
```
f[x0] + f'[x0] * (x - x0)
```

p2

Input ▹
```
f''[x0]
p2[x]
p2[x] - p1[x]
```

Input ▹
```
Expand[f''[x0] * (x - x0)]
```

This does not work the way we did it above because we need x^2 in the formular so we have to put it into brackets and square the bracket.

Input ▹
```
Expand[f''[x0] * (x - x0) ^2]
```

This is not the result we are looking for; but we see by comparing this result to the result of p2-p1 that this is two times the result of p2-p1. Now we will divide the hole term by 2.

Input ▹ $\text{Expand}\left[\frac{f''[x0]*(x-x0)^{\wedge}2}{2}\right]$

Now we have exactly the same result like 3 steps before.

Input ▹ $\text{Expand}\left[f[x0]+f'[x0]*(x-x0)+\frac{f''[x0]*(x-x0)^{\wedge}2}{2}\right]$

This is the function of p2 evaluated in a different way than in 1.1.

p3

```
f'''[x0]
p3[x]
p3[x] - p2[x]
```

Input ▹ $\text{Expand}\left[\frac{f'''[x0]*(x-x0)^{\wedge}2}{2}\right]$

Now we can see that we have got exactly the same problem like above. We try to change the exponent to 3 to get x^3.

Input ▹ $\text{Expand}\left[\frac{f'''[x0]*(x-x0)^{\wedge}3}{2}\right]$

Now we have got x^3 but all the coefficients are not right.
If we compare the coefficients of x^3 we can see that there is a 3 in the original term and a 9 in the one above. But 9 divided by 3 is 3 so we try to divide the term above by 2 times 3.

Input ▹ $\text{Expand}\left[\frac{f'''[x0]*(x-x0)^{\wedge}3}{2*3}\right]$

And now we get the right function so we can see that this formular is right and if we add the terms of p2, p1 and p0 we should get the function of f because we reached the degree of f and p3 is equal to f.

Input ▹ $\text{Expand}\left[f[x0]+f'[x0]*(x-x0)+\frac{f''[x0]*(x-x0)^{\wedge}2}{2}+\frac{f'''[x0]*(x-x0)^{\wedge}3}{2*3}\right]$

If we look at this funktion we can see that the second summand can also be written as $\frac{f'[x0]*(x-x0)^{\wedge}1}{1}$ and the third summand can be written as $\frac{f''[x0]*(x-x0)^{\wedge}2}{1*2}$ and the fourth summand can be written as $\frac{f'''[x0]*(x-x0)^{\wedge}3}{1*2*3}$. The first summand can be written as f[x0]×(x-x0)^0.

If we look at the denominators of the 3 terms we see 1, 1*2 and 1*2*3. This can be shorten to 1!, 2! and 3! and the number in front od the ! is always the number of the derivation. We can also see that the exponent of (x-x0) is the number of the derivation.

Part 2.1:Solve of the approximation of a sinus function

Open / Close

approximation at point π of f[x]=sin[x]

Input ▹
```
Clear[k, x, x0]
k[x_] := Sin[x]
x0 = 3;
```

Solve t0

Input ▹
```
Clear[t0, x, a1]
t0[x_] := a1
```

Input ▹
```
→ MDRealOnly;
Clear[x, p0, a1];
{a1} = {a1} /.
Flatten[Solve[{ k[x0] == t0[x0]}, { a1 } ]]
```

Graph k[x] and t0[x]

Input ▹
```
Plot[{k[x], t0[x]}, {x, -2 π, 2 π}, PlotRange → {-2, 2}]
```

This is just a straight line through the point p0.

Solve t1[x]

Input ▹
```
Clear[t1, b1, c1]
t1[x_] := b1 x + c1
```

Input ▹
```
Clear[x, b1, c1]
{b1, c1} = {b1, c1} /.
  Flatten[Solve[{k[x0] == t1[x0], k'[x0] == t1'[x0]}, {b1, c1}]];
t1[x]
```

Graph k[x] and t1[x]

Input ▹
```
Plot[{k[x], t1[x]}, {x, -2 π, 2 π}, PlotRange → {-2, 2}]
```

This is the tangent line and it shows the gradient at the point p0.

Solve t2[x]

Input ▹
```
Clear[t2, x, d1, e1, i1]
t2[x_] := d1 x^2 + e1 x + i1
```

Input ▷
```
Clear[d1, e1, i1, x]
{d1, e1, i1} = {d1, e1, i1} /.
  Flatten[Solve[{k[x0] == t2[x0], k'[x0] == t2'[x0],
     k''[x0] == t2''[x0]}, {d1, e1, i1}]]; t2[x]
```

Graph k[x] and t2[x]

Input ▷
```
Plot[{k[x], t2[x]}, {x, -2 π, 2 π}, PlotRange → {-2, 2}]
```

This graph shows the curvature at the point p0.

Solve t3[x]

Input ▷
```
Clear[t3, j1, l2, l1, m1]
t3[x_] := j1 x^3 + l2 x^2 + l1 x + m1
```

Input ▷
```
Clear[j1, l2, l1, m1, x]
{j1, l2, l1, m1} = {j1, l2, l1, m1} /.
  Flatten[
   Solve[{k[x0] == t3[x0], k'[x0] == t3'[x0], k''[x0] == t3''[x0],
     k'''[x0] == t3'''[x0]}, {j1, l2, l1, m1}]]; t3[x]
```

Graph k[x] and t3[x]

Input ▷
```
Plot[{k[x], t3[x]}, {x, -2 π, 2 π}, PlotRange → {-2, 2}]
```

Solve t4[x]

Input ▷
```
Clear[t4, n1, o1, p1, q1, r1, x]
t4[x_] := n1 * x^4 + o1 * x^3 + p1 * x^2 + q1 * x + r1
```

Input ▷
```
Clear[n1, o1, p1, q1, r1, x]
{n1, o1, p1, q1, r1} = {n1, o1, p1, q1, r1} /.
  Flatten[
   Solve[{k[x0] == t4[x0], k'[x0] == t4'[x0], k''[x0] == t4''[x0],
     k'''[x0] == t4'''[x0], k''''[x0] == t4''''[x0]},
    {n1, o1, p1, q1, r1}]]; t4[x]
```

Graph k[x] and t4[x]

Input ▷
```
Plot[{k[x], t4[x]}, {x, -2 π, 2 π}, PlotRange → {-2, 2}]
```

Solve t5[x]

Input ▷
```
Clear[t5, s1, u1, v1, w1, z1, h1, a2, x]
t5[x_] := s1 * x^5 + u1 * x^4 + v1 * x^3 + w1 * x^2 + h1 * x + z1
```

Input ▷
```
Clear[s1, u1, v1, w1, z1, h1, x]
{s1, u1, v1, w1, z1, h1} = {s1, u1, v1, w1, z1, h1} /.
  Flatten[
   Solve[{k[x0] == t5[x0], k'[x0] == t5'[x0], k''[x0] == t5''[x0],
     k'''[x0] == t5'''[x0], k''''[x0] == t5''''[x0],
     k'''''[x0] == t5'''''[x0]}, {s1, u1, v1, w1, z1, h1}]]; t5[x]
```

Graph k[x] and t5[x]

Input ▷
```
Plot[{k[x], t5[x]}, {x, -2 π, 2 π}, PlotRange → {-2, 2}]
```

Input ▷
```
Clear[t6, s2, u2, v2, w2, z2, h2, a3, x]
t6[x_] := s2 * x^6 + u2 * x^5 + v2 * x^4 + w2 * x^3 + h2 * x^2 + z2 * x + a3
```

Input ▷
```
{s2, u2, v2, w2, z2, h2, a3} = {s2, u2, v2, w2, z2, h2, a3} /.
  Flatten[
   Solve[{k[x0] == t6[x0], k'[x0] == t6'[x0], k''[x0] == t6''[x0],
     k'''[x0] == t6'''[x0], k''''[x0] == t6''''[x0],
     k'''''[x0] == t6'''''[x0], k''''''[x0] == t6''''''[x0]},
    {s2, u2, v2, w2, z2, h2, a3}]]; t6[x]
```

Graph of k[x] and t6[x]

Input ▷
```
Plot[{k[x], t6[x]}, {x, -2 π, 2 π}, PlotRange → {-2, 2}]
```

The graphs of the functions t3 to t6 show that the approximation gets more accurate the higher the degree of the function is.

Part 2.2: Movie

Open / Close

This movie shows how the approximation get more and more accurate.

Input ▷

[More...](#) ;
```
Clear[x, z, n];
polynom[z_, n_] :=
  Expand[ Sum[ (D[k[x], {x, ß}] /. x → x0) / ß! * (x - x0)^ß, {ß, 0, n}] ] /. x → z // N   ;
(* enter your f *)

MDMovie[ MDPlot[ {polynom[z, n], Sin[z]},
  {z, -2 π, 2 π}, PlotRange → {-2.1, 2.1},
  Background → $MDNotebookBackground, PlotLabel →
   "n = " <> ToString[NumberForm[n, {5, 3}]] <> "\n"], {n, 0, 5, 1}]
```

$$\text{polynom}[z, n] := \text{Expand}\left[\sum_{ß=0}^{n} \frac{(D[k[x], \{x, ß\}] /. x \to x0)}{ß!} * (x - x0)^{ß}\right] /. x \to z // N$$

Part 2.3: Compare of the coefficients of the approximations of the sinus function

Open / Close

t0

Input ▹
```
k[x0]
t0[x]
```

The first approximation is just the function value of the original function

t1

We will try to use our results from 1.3 to see if it works at this function, too.

Input ▹
```
k'[x0]
t1[x]
t1[x] - t0[x]
Expand[k'[x0] * (x - x0)]
k[x0] + Expand[k'[x0] * (x - x0)]
```

We can see that our result from 1.3 works for the second approximation.

t2

Now we try it for the next approximation.

Input ▹
```
k''[x0]
t2[x]
t2[x] - t1[x]
Expand[(k''[x0] * (x - x0)^2)/2]
k[x0] + Expand[k'[x0] * (x - x0)] + Expand[(k''[x0] * (x - x0)^2)/2]
```

We can see this works for the second approximation as well as for the third approximation.

We will not try this with all the approximations we as did in chapter 2.1. We will just try it with the tenth approximation.

```
Clear[a5, b5, c5, d5, e5, f5, g5, h5, i5, j5, k5, t10]
t10[x_] := a5 * x^10 + b5 * x^9 + c5 * x^8 + d5 * x^7 +
  e5 * x^6 + f5 * x^5 + g5 * x^4 + h5 * x^3 + i5 * x^2 + j5 * x + k5
{a5, b5, c5, d5, e5, f5, g5, h5, i5, j5, k5} =
 {a5, b5, c5, d5, e5, f5, g5, h5, i5, j5, k5} /.
```

```
Flatten[Solve[{k[x0] == t10[x0], k'[x0] == t10'[x0],
   k''[x0] == t10''[x0], k'''[x0] == t10'''[x0],
   k''''[x0] == t10''''[x0], k'''''[x0] == t10'''''[x0],
   k''''''[x0] == t10''''''[x0], k'''''''[x0] == t10'''''''[x0],
   k''''''''[x0] == t10''''''''[x0], k'''''''''[x0] ==
    t10'''''''''[x0], k''''''''''[x0] == t10''''''''''[x0]},
  {a5, b5, c5, d5, e5, f5, g5, h5, i5, j5, k5}]]; t10[x]
```

Input ▹

$$\text{Expand}\Big[k[x0] + k'[x0]\,(x - x0) + \frac{1}{2}k''[x0]\,(x - x0)^2 + \frac{k^{(3)}[x0]\,(x - x0)^3}{2\times 3} + \frac{k^{(4)}[x0]\,(x - x0)^4}{4!} + \frac{k^{(5)}[x0]\,(x - x0)^5}{5!} + \frac{k^{(6)}[x0]\,(x - x0)^6}{6!} + \frac{k^{(7)}[x0]\,(x - x0)^7}{7!} + \frac{k^{(8)}[x0]\,(x - x0)^8}{8!} + \frac{k^{(9)}[x0]\,(x - x0)^9}{9!} + \frac{k^{(10)}[x0]\,(x - x0)^{10}}{10!}\Big]$$

Now we can see that the new way produces the same result like the old way.

4 Result and Summary

Open/Close Print

Section 1.1: Taylor

Open / Close

Now we look for a shorter way to write this term :

$$\text{Simplify}\Big[f[x0] + f'[x0]*(x - x0) + \frac{f''[x0]*(x - x0)\verb|^|2}{2} + \frac{f'''[x0]*(x - x0)\verb|^|3}{2*3} + \frac{f''''[x0]*(x - x0)\verb|^|4}{4!} + \frac{f'''''[x0]*(x - x0)\verb|^|5}{5!} + \frac{f''''''[x0]*(x - x0)\verb|^|6}{6!} + \frac{f'''''''[x0]*(x - x0)\verb|^|7}{7!} + \frac{f''''''''[x0]*(x - x0)\verb|^|8}{8!} + \frac{f'''''''''[x0]*(x - x0)\verb|^|9}{9!} + \frac{f''''''''''[x0]*(x - x0)\verb|^|10}{10!}\Big]$$

We can see this is a sum. So now we will try to write this with a $\sum$ to get a shorter term.
We will define the number of derivation as a.

$$\sum_{a=0}^{n} \frac{f^{(a)}[x0]}{a!}\,(x - xo)^a$$

Now we will compare these two results to check if the new term is the right term to approximat functions.

Input ▹

$$\texttt{k[x0]} + \texttt{Expand}[\texttt{k}'\texttt{[x0]}\ (\texttt{x} - \texttt{x0})] + \texttt{Expand}\left[\frac{1}{2}\ \texttt{k}''\texttt{[x0]}\ (\texttt{x} - \texttt{x0})^2\right] +$$
$$\texttt{Expand}\left[\frac{\texttt{k}^{(3)}\texttt{[x0]}\ (\texttt{x} - \texttt{x0})^3}{2 \times 3}\right] + \texttt{Expand}\left[\frac{\texttt{k}^{(4)}\texttt{[x0]}\ (\texttt{x} - \texttt{x0})^4}{4\,!}\right] +$$
$$\texttt{Expand}\left[\frac{\texttt{k}^{(5)}\texttt{[x0]}\ (\texttt{x} - \texttt{x0})^5}{5\,!}\right] + \texttt{Expand}\left[\frac{\texttt{k}^{(6)}\texttt{[x0]}\ (\texttt{x} - \texttt{x0})^6}{6\,!}\right] +$$
$$\texttt{Expand}\left[\frac{\texttt{k}^{(7)}\texttt{[x0]}\ (\texttt{x} - \texttt{x0})^7}{7\,!}\right] + \texttt{Expand}\left[\frac{\texttt{k}^{(8)}\texttt{[x0]}\ (\texttt{x} - \texttt{x0})^8}{8\,!}\right] +$$
$$\texttt{Expand}\left[\frac{\texttt{k}^{(9)}\texttt{[x0]}\ (\texttt{x} - \texttt{x0})^9}{9\,!}\right] + \texttt{Expand}\left[\frac{\texttt{k}^{(10)}\texttt{[x0]}\ (\texttt{x} - \texttt{x0})^{10}}{10\,!}\right]$$

Input ▹

```
Clear[n, ß]
n = 10;
```
$$\texttt{Expand}\left[\sum_{\texttt{ß}=0}^{\texttt{n}} \frac{(\texttt{D[k[x], \{x, ß\}] /. x -> x0})}{\texttt{ß}\,!} * (\texttt{x} - \texttt{x0})^{\texttt{ß}}\right]$$

Input ▹

$$\texttt{Plot}\left[\left\{\texttt{Expand}\left[\sum_{\texttt{ß}=0}^{\texttt{n}} \frac{(\texttt{D[k[x], \{x, ß\}] /. x -> x0})}{\texttt{ß}\,!} * (\texttt{x} - \texttt{x0})^{\texttt{ß}}\right] \texttt{ /. x} \to \texttt{t},\ \texttt{k[x] /. x} \to \texttt{t}\right\}, \{\texttt{t}, -2\,\pi, 2\,\pi\}, \texttt{PlotRange} \to \{-2, 2\}\right]$$

If we compare the results we can see that the second term gives the same result as the first term; but it is shorter.

This is the Taylor term to approximate any function at a point in its domain.

Section 1.2: Taylor approximation at any point with any function.

Open / Close

In this chapter you can create a Taylorpolynominal with any function you like.

Input ▹

```
Clear[ü, y0, n, ß]
ü[x_] := Sin[x] * ⅇ^2 x    (*enter your function ü *)
y0 := 2    (*enter your point as y0 *)
n := 10
(*enter your n this is the number of approximation*)
```
$$\texttt{Expand}\left[\sum_{\texttt{ß}=0}^{\texttt{n}} \frac{(\texttt{D[ü[x], \{x, ß\}] /. x} \to \texttt{y0})}{\texttt{ß}\,!} * (\texttt{x} - \texttt{y0})^{\texttt{ß}}\right]$$

Section 2.1: Accuracy of a Taylorpolynominal

Open / Close

We defined the Taylorpolynominal as a function "polynom" depending on z (difference to the approximated point) and n(Degree of the Taylorpolynominal). Then we defined the function "diff" which is depending on n and u (z changed into x0+u). "Diff" is the absolute value of the original function k and and the Taylorpolynominal at the point x0+u.

```
Clear[n]
polynom[z_, n_] :=
```

Input ▹ $$\text{Expand}\left[\sum_{ß=0}^{n} \frac{(D[k[x], \{x, ß\}] \;/.\; x \to x0)}{ß\,!} * (x - x0)^{ß}\right] \;/.\; x \to z \;//\; N$$

```
diff[n_, u_] := Abs[k[x0 + u] - polynom[x0 + u, n]] // N
```

Here we created the table to compare the differences of both functions at different points on the left and the right side of the approximation. This shows the accuracy of the approximation.

```
More... ;
start = 0;
stop = 10;
step = 1;
```

Input ▹
```
data = Table[{n, diff[n, -2] , diff[n, -1], diff[n, 1] ,
    diff[n, 2]  // N}, {n, start, stop, step}] // Chop;
MDShowTable[data, {"n", "Diff in x=x0-2\n          x0=3",
   "Diff in x=x0-1\n          x0=3",
   "Diff in x=x0+1\n          x0=3",
   "Diff in x=x0+2\n          x0=3"}];
```

n	Diff in x=x0−2 x0=3	Diff in x=x0−1 x0=3	Diff in x=x0+1 x0=3	Diff in x=x0+2 x0=3
0	0.700351	0.768177	0.897923	1.10004
1	1.27963	0.221815	0.09207	0.879941
2	0.997394	0.151255	0.16263	1.16218
3	0.322596	0.0137437	0.00236875	0.157809
4	0.228516	0.00786368	0.00824875	0.251889
5	0.035482	0.000386262	1.18497×10^{-6}	0.0121087
6	0.022938	0.000190262	0.000197185	0.0246527
7	0.00220466	6.16492×10^{-6}	7.57894×10^{-7}	0.000489941

8	0.00130866	2.66491×10^{-6}	2.74211×10^{-6}	0.00138594
9	0.0000881569	6.32391×10^{-8}	1.39527×10^{-8}	0.0000108737
10	0.0000483347	2.43502×10^{-8}	2.49362×10^{-8}	0.000050696

This table shows the difference of both functions on the left and the right sides of the approximated function and thus the accuracy of the Taylorpolynominal.
You can see that the differences are minor:
There is only a difference of 32/100 at the point x0-2 when the degree of the function is 3 and 15/100 at the point x0+2.
When the degree is ten there is only a difference of 5/100000 to both sides.

Section 3.1: Use of Taylorpolynominals

Open / Close

Taylorpolynominals are used to explain the Bernoulli-Effect (stochastics) and the derivation of the sentence of L´Hospital. Furthermore, it is used in calculators.

5 *Our Team*

Open / Close Print

Participants

Open / Close

Participants:

Uta Geratz, Bettina-von-Arnim-Gymnasium, Dormagen, Germany
Carl Philip Heising, Bettina-von-Arnim-Gymnasium, Dormagen, Germany

Reetta Lumiaho, Vihannin lukio, Vihanti, Finland
Leena Luikko, Vihannin lukio, Vihanti, Finland

1 Description of the Project (team3)

Open / Close Print

The topic of this project is to find a proper way to approximate every kind of a function with a polynomial function.
The idea behind this is, that it is very easy to analyze a polynomial function, e.g. the root of the function, the maxima and minima or the limit. But there are many different functions in mathematics, e.g. exponential functions, logarithm functions, sine and cosine functions and many more.
We want to develop an accurate formula with an as small as possible error, which allows to display any function as an polynomial function, to make the analysis of the function easier.

2 Brainstorming and Theory

Open / Close Print

Brainstorming

Open / Close

Our goal is to develop a formula to approximate any function with a polynomial function. To do that we first need a general polynomial function without values, e.g. f(x)=
$a x^3 + b x^2 + c x + d$.
Then we have to compare different values of the function we want to approximate with our polynomial function. We can check the values on different positions and we can also check the values of the derivations of the function, because the values of the derivations of the function, which we want to approximate, have also to be the same as the values of the derivations of the polynomial function.
For the first try it is a good idea to approximate any polynomial function with another polynomial function. Because we are approximating a polynomial function, our function has to be in the end the same as the polynomial function, we are approximating.

What mathematics do we need?

Open/Close

Our task in this section is to make a approximation of a polynomial function. To find the polynomial function, which is going to approximate the given function we have to compare the values of the functions, as well as the values of the derivations of the functions.
We need to know how to calculate function values. We need to know how to create derevations of functions.
To see the results in a graphic we need to know how to plot our functions into a coordinate system.
It is possible, that we will need more mathematics after we find out, how the approximation works.

Approximation of a polynomial function

Open/Close

We are trying to approximate a polynomial function with another polynomial function.

1. We are looking for a polynomial which has the same function value like f(x) at x0.

Input ▹
```
Clear[f, x, x0];
x0 = 2;
f[x_] := 5 x^3 - 7 x^2 + 4 x^1 - 3
```

Now we plot the function f(x).

Input ▹
```
MDPlot[{f[x]}, {x , -5 , 5 } ];
```

The new function has to be a polynomial function of degree 0. The name is p0.

Input ▹
```
Clear[p0, x, a0];
p0[x_] := a0

Clear[a]
```

Input ▹
```
 Switch → Pure Solve ;
Clear[x, a0];
{a0} = {a0} /.
  Flatten[MDRealOnly[Solve[{ f[x0] == p0[x0] }, { a0 } ]]]
```

We have now solved the value of a0. It is 17. So p0(x)=17
Now we plot f(x) and the new function.

Input ▹
```
MDPlot[{f[x], p0[x]}, {x , -5 , 5 } ];
```

The taylor-polynomial of the zeroth degree have the same value as the functuin f(x) at the point x0.

2. The next approximation is the tangent line at x0.

The new function has to be a polynomial function of degree 1. The name is p1.

Input ▷
```
Clear[p1, a1, b1, x];
p1[x_] := a1 * x + b1
```

Input ▷
```
→ Pure Solve;
Clear[a1, b1];
{a1, b1 } = {a1, b1 } /. Flatten[MDRealOnly[
    Solve[{ f[x0] == p1[x0], f'[x0]  == p1'[x0] }, {a1, b1 } ]]]
```

We have now solved the value of a1 and b1. They are 36 and -55. So p1(x)=36x-55
Now we plot the two functions.

Input ▷
```
MDPlot[{f[x], p1[x]}, {x , -2 , 5 } ];
```

The first taylor-polynomial hase the same slope as the function f(x).

3. We are doing this for the next polynomial degrees.

The new function has to be a polynomial function of degree 2. The name is p2.

Input ▷
```
Clear[a2, b2, c2, x]
p2[x_] := a2 * x^2 + b2 * x + c2
```

Input ▷
```
→ Pure Solve;
Clear[x, a2, b2, c2];
{a2, b2, c2} = {a2, b2, c2} /. Flatten[
MDRealOnly[Solve[{ f[x0] == p2[x0], f'[x0]  == p2'[x0],
      f''[x0]  == p2''[x0] }, {a2,  b2, c2 } ]]]
```

We plot both functions.

Input ▷
```
MDPlot[{f[x], p2[x]}, {x , -2 , 5 } ];
```

The second taylor-polynomial give us the curveness of f(x)
The next one is a polynomial function of degree 3. The name is p3.

Input ▷
```
Clear[a3, b3, c3, d3, x]
p3[x_] := a3 * x^3 + b3 * x^2 + c3 * x + d3
```

Input ▷
```
Clear[x, a3, b3, c3, d3];
{a3, b3, c3, d3} = {a3, b3, c3, d3} /. Flatten[
MDRealOnly[Solve[
     { f[x0] == p3[x0], f'[x0] == p3'[x0], f''[x0] == p3''[x0],
      f'''[x0] == p3'''[x0] }, { a3, b3, c3, d3} ]]]
```

We plot both functions.

Input ▹
```
MDPlot[{f[x], p3[x]}, {x , -5 , 5 } ];
```

The new function is equal to f(x). The approximation is done. The degree of the taylor-polynomial is the same as the degree of f(x).

Input ▹
```
MDPlot[{f[x], p0[x], p1[x], p2[x], p3[x]}, {x , -5 , 5 } ];
```

We plot all taylor-polynomials in one coordinate system.

Section: Comparison of Coefficients

Open / Close

4. We are now trying to find a structure in the build of a taylor-polynomial

We compare the coefficients of the taylor-polynomial with the derivations of the function to find a formula how to create a taylor-polynomial.

Input ▹
```
f[x0]
p0[x]
```

The coefficient of the zeroth taylor-polynomial is the function at the point x0.

Input ▹
```
f'[x0]
p1[x]
```

The first derivation at the point x0 is the value of the coefficient with the x of the first taylor-polynomial. To analyse only the second summand of the taylor-polynomial we subtract the zeroth taylor-polynomial from the first.

Input ▹
```
p1[x] - p0[x]
```

36 is the value of the first derivation at the point x0. -72 is this value multiplicated with -2. We now do the same for the next approximation.

Input ▹
```
f''[x0]
p2[x]
p2[x] - p1[x]
```

The coefficient with the x^2 is the value of the second derivation of the function at the point x0 devided by 2. The coefficient of the x is the value of the second derivation of the function multiplicated with 2x, the linear part is the value of the second derivation of the function devided by 2 and then multiplicated with 4. We continue our analysis.

Input ▹
```
f'''[x0]
p3[x]
p3[x] - p2[x]
```

The coefficient of the x^3-part is the value of the third derivation of the function at the point x0 devided by 6. The x^2-part is the value multiplicated with x^2. The next one is the value multiplicated with 2x. The last one is the value devided by 6 multiplicated with 8.
You can see a structure in it. The coefficient of the part with the highest degree is created through building the n-derivation at the point x0 and then devide it with the faculty of n (n!).
To get the other coefficients the experssion $\frac{f^{(n)}[x0]}{n!}$ is multiplicated with the experssion $(x - x0)^n$. n is the degree of the taylor-polynomial. To get the taylor-polynomial you have to sum up all the taylor-polynomials of lower degrees. So to get the third taylor-polynomial you have to sum up the expression $\frac{f^{(n)}[x0]}{n!} * (x - x0)^n$ for n={0;1;2;3}. Therefore the formula for the taylor-polynomial is $\sum_{i=0}^{n} \frac{f^{(n)}[x0]}{n!} * (x - x0)^n$.

3 Developing Models

Open / Close Print

Section: Transfer

Open / Close

In this section we are going to transfer the results from the second section on a different kind of function, to see if the formula we develop still works with this functions.
We will take an exponential function and approximate it with our polynomial function.
Because the polynomial function is only an approximation we are going to calculate the error rate between the exponential function and the polynomial function. With this information we will find out how many approximations are necesarry to get a proper result.

Section: Approximation of a exponentional function

Open / Close

We now approximate a exponentional function with a taylor-polynomial. For that purpose we are doing the same action as in the first part.

Input ▹
```
Clear[g, x];
g[x_] := ⅇ^x
x0 := 2
```

Input ▹
```
Clear[s0, x, a0];
s0[x_] := a0
```

```
Switch → Pure Solve ;
Clear[x, a0];
{a0} = {a0} /.
  Flatten[MDRealOnly[Solve[{ g[x0] == s0[x0] }, { a0 } ]]]
```

We start again with a polynomial of the zeroth degree and continue to increase the degree, until we get a good approximation.

```
Input ▷ MDPlot[{g[x], s0[x]}, {x , -5 , 5 } ];

Input ▷ Clear[s1, a1, b1, x];
        s1[x_] := a1 * x + b1

        → Pure Solve;
Input ▷ Clear[a1, b1];
        {a1, b1 } = {a1, b1 } /. Flatten[MDRealOnly[
            Solve[{ g[x0] == s1[x0], g'[x0] == s1'[x0] }, {a1, b1 } ]]]

Input ▷ MDPlot[{g[x], s1[x]}, {x , -5 , 5 } ];

Input ▷ Clear[a2, b2, c2, x]
        s2[x_] := a2 * x^2 + b2 * x + c2

        → Pure Solve;
        Clear[x, a2, b2, c2];
Input ▷ {a2, b2, c2} = {a2, b2, c2} /. Flatten[
        MDRealOnly[Solve[{ g[x0] == s2[x0], g'[x0] == s2'[x0],
              g''[x0] == s2''[x0] }, {a2, b2, c2 } ]]]

Input ▷ MDPlot[{g[x], s2[x]}, {x , -5 , 5 } ];

Input ▷ Clear[a3, b3, c3, d3, x]
        s3[x_] := a3 * x^3 + b3 * x^2 + c3 * x + d3

        Clear[x, a3, b3, c3, d3];
        {a3, b3, c3, d3} = {a3, b3, c3, d3} /. Flatten[
Input ▷ MDRealOnly[Solve[
             { g[x0] == s3[x0], g'[x0] == s3'[x0], g''[x0] == s3''[x0],
              g'''[x0] == s3'''[x0] }, { a3, b3, c3, d3} ]]]

Input ▷ MDPlot[{g[x], s3[x]}, {x , -5 , 5 } ];

Input ▷ Clear[x, a4, b4, c4, d4, e4]
        s4[x_] := a4 * x^4 + b4 * x^3 + c4 * x^2 + d4 * x + e4

        → Pure Solve;
        Clear[x, a4, b4, c4, d4, e4];
Input ▷ {a4, b4, c4, d4, e4} = {a4, b4, c4, d4, e4} /. Flatten[
```

```
MDRealOnly[Solve[{ g[x0] == s4[x0], g'[x0] == s4'[x0],
      g''[x0] == s4''[x0], g'''[x0] == s4'''[x0],
      g''''[x0] == s4''''[x0]}, { a4, b4, c4, d4, e4 } ]]]
```

Input ▷
```
MDPlot[{g[x], s4[x]}, {x , -5, 5} ];
```

Input ▷
```
Clear[x, a5, b5, c5, d5, e5, f5]
s5[x_] := a5 * x^5 + b5 * x^4 + c5 * x^3 + d5 * x^2 + e5 * x + f5
```

Input ▷
```
→ Pure Solve;
Clear[x, a5, b5, c5, d5, e5, f5];
{a5, b5, c5, d5, e5, f5} = {a5, b5, c5, d5, e5, f5} /. Flatten[
MDRealOnly[Solve[
     { g[x0] == s5[x0], g'[x0] == s5'[x0], g''[x0] == s5''[x0],
      g'''[x0] == s5'''[x0], g''''[x0] == s5''''[x0],
      g'''''[x0] == s5'''''[x0]}, { a5, b5, c5, d5, e5 , f5} ]]]
```

Input ▷
```
MDPlot[{g[x], s5[x]}, {x , -5 , 5 } ];
```

Input ▷
```
Clear[x, a6, b6, c6, d6, e6, f6, g6]
s6[x_] := a6 * x^6 + b6 * x^5 + c6 * x^4 + d6 * x^3 + e6 * x^2 + f6 * x + g6
```

Input ▷
```
→ Pure Solve;
Clear[x, a6, b6, c6, d6, e6, f6, g6];
{a6, b6, c6, d6, e6, f6, g6} =
 {a6, b6, c6, d6, e6, f6, g6} /. Flatten[
MDRealOnly[Solve[{ g[x0] == s6[x0],
      (D[g[x], {x, 1}] /. x -> x0) == (D[s6[x], {x, 1}] /. x → x0),
      (D[g[x], {x, 2}] /. x -> x0) == (D[s6[x], {x, 2}] /. x → x0),
      (D[g[x], {x, 3}] /. x -> x0) == (D[s6[x], {x, 3}] /. x → x0),
      (D[g[x], {x, 4}] /. x -> x0) == (D[s6[x], {x, 4}] /. x → x0),
      (D[g[x], {x, 5}] /. x -> x0) == (D[s6[x], {x, 5}] /. x → x0),
      (D[g[x], {x, 6}] /. x -> x0) == (D[s6[x], {x, 6}] /. x → x0)},
     { a6, b6, c6, d6, e6 , f6, g6} ]]]
```

Input ▷
```
MDPlot[{g[x], s6[x]}, {x , -5, 5 } ];
```

As the degree of the taylor-polynomial increase the approximation becomes more accurate. With a high degree the function values in the enviroment of the approximate value x0 have only a small difference in comparison to the value of the approximated function.

The difference to the approximation of a polynomial function is, that the taylor-polynomial will not be the same as the approximated function. With each degree the approximation becomes more accurate, but to reach the approximated function you have to calculate the taylor-polynomial infinitely.

Input ▷

```
More... ;
Clear[tay, n];
tay[n_] :=
  Sum[((D[g[x], {x, i}] / i!) /. x → x0) (x - x0)^i, {i, 0, n}];

MDMovie[MDPlot[{g[x], tay[n]},
  {x, -2 π, 2 π}, PlotRange → {{-2, 4}, {-1, 7}},
  Background → $MDNotebookBackground, PlotLabel →
   "n = " <> ToString[NumberForm[n, {5, 3}]] <> "\n"], {n, 0, 6, 1}]
```

In this movie you can see how the approximation becomes more accurate step by step.

Section: Difference of the both function

Open/Close

We first define a function with which we can calculate the value of any point of an taylor-polynomial for any function.

Input ▷

```
tay[z_, n_] :=
 Sum[((D[g[x], {x, i}] / i!) /. x → x0) (x - x0)^i, {i, 0, n}] /. x → z
```

Then we define a function which solve the absolute value of the difference of the approximated function and the taylor-polynomial in a certain point.

Input ▷

```
difference[j_, n_] := Abs[g[x0 + j] - tay[x0 + j, n]] // N
```

Now we table the results.

Input ▷

```
More... ;
Clear[difference, x, j, n];
difference[j_, n_] := Abs[g[x0 + j] - tay[x0 + j, n]] // N
start = 0;
stop = 15;
step = 1;

data = Table[{xq, difference[2, xq], difference[-2, xq] // N},
```

```
    {xq, start, stop, step}] // Chop;
MDShowTable[data, {"Taylor-Polynominal degree",
   "Difference at x0+2", "Difference at x0-2"}];
```

Taylor–Polynominal degree	Difference at x0+2	Difference at x0–2
0	47.2091	6.38906
1	32.431	8.38906
2	17.6529	6.38906
3	7.80079	3.46302
4	2.87476	1.46302
5	0.904342	0.507396
6	0.247537	0.149409
7	0.0598788	0.0382498
8	0.0129642	0.0086648
9	0.00253871	0.00176068
10	0.000453614	0.00032442
11	0.0000745055	0.0000546882
12	0.0000113208	8.49651×10^{-6}
13	1.60004×10^{-6}	1.22421×10^{-6}
14	2.11362×10^{-7}	1.64461×10^{-7}
15	2.62054×10^{-8}	2.06953×10^{-8}

In this table you can see the degree of the taylor-polynomial and the difference of the approximated function and the taylor-polynomial at the points x0-2 and x0+2. The approximated function is the exponentional functuon f(x)= e^{x}. The point x0=2.

4 Result and Summary

Open/Close Print

Section: Approximation of any function with a polynomial function

Open/Close

With the formula $\sum_{i=0}^{n} \frac{f^{(n)}[x0]}{n!} * (x - x0)^{n}$ you can calculate the taylor-polynomial for any function f(x). n is a natural and gives the degree of the taylor-polynomial. x0 is the point at which you calculate the derivations of the function. n! is 1*2*3*4*...*n. To calculate a taylor-polynomial of the nth degree you have to sum up all taylor-polynomials from 0 to n.

Section: Use of the taylor-polynomial

Open/Close

The taylor-polynom is used in normal calculators. The calculations which a normal calculator can do well is multiplication, devision, addition and subtraction. So if a calculator calculates e.g. the value of a sinus, it just uses a taylor-polynomial to calculate this value.

5 Our Team

Open/Close Print

Participants

Open/Close

Our team consisted of two Finnish students, Mikko Huhtala and Anu Sandvik, and two German students, Jens Schatten and Mark Smoliar.

Our Experience with the Project

Open/Close

This project was an interesting event. It was exciting to work with students from another country on a mathematical topic. Another interesting aspect of the project was, that there was a meeting in Finland as well as in Germany, so we get to know the country of the other students. It was a bit complicated to work all the time in English, because no one of us was used to it, but we managed this problem as well and in the end completed our task in the project.

6

Picture of the whole group

Open / Close Print

Teachers of the group: Wolfgang Breivogel, Ari Tranberg.
Students: Jere Junttila, Miina Honkala, Angela Brück, Frederike Franken,
Uta Geratz, Carl Philip Heising, Reetta Lumiaho, Leena Liukko, Mikko Huhtala, Anu Sandvik, Jens Schatten and Mark Smoliar.

Also in the picture: Mr. Schieren (headmaster of the Bettina-von-Arnim-Gymnasium in Dormagen Germany).

New Chapter Cut Last Chapter

INTERNET Open / Close Print