Blatt 4

Aufgabe 13

```
> restart:
> A := [ 1, Pi, exp(1), pi, x+y ];
> op(A);
> 'whattype'('op'(1, 'A')) = whattype(op(1, A));
> 'whattype'('op'(2, 'A')) = whattype(op(2, A));
> 'whattype'('op'(3, 'A')) = whattype(op(3, A));
> 'whattype'('op'(4, 'A')) = whattype(op(4, A));
> 'whattype'('op'(5, 'A')) = whattype(op(5, A));
> op(op(3, A));
> 'whattype'('op'('op'(3, 'A'))) = whattype(op(op(3, A)));
> op(op(5, A));
> 'whattype'('op'(1, 'op'(5, 'A'))) = whattype(op(1, op(5, A)));
> 'whattype'('op'(2, 'op'(5, 'A'))) = whattype(op(2, op(5, A)));
> B := { x * y / z, u / v * w, (alpha + beta) / (beta * delta) };
> op(B);
> whattype(op(1, B));
> whattype(op(2, B));
> whattype(op(3, B));
> op(op(1, B));
> whattype(op(1, op(1, B)));
> whattype(op(2, op(1, B)));
> whattype(op(3, op(1, B)));
> op(op(2, op(1, B)));
> whattype(op(1, op(2, op(1, B))));
> whattype(op(2, op(2, op(1, B))));
> op(op(2, B));
> whattype(op(1, op(2, B)));
> whattype(op(2, op(2, B)));
> whattype(op(3, op(2, B)));
> op(op(3, op(2, B)));
> whattype(op(1, op(3, op(2, B))));
> whattype(op(2, op(3, op(2, B))));
> op(op(3, B));
> whattype(op(1, op(3, B)));
> whattype(op(2, op(3, B)));
> whattype(op(3, op(3, B)));
> op(op(1, op(3, B)));
> whattype(op(1, op(1, op(3, B))));
> whattype(op(2, op(1, op(3, B))));
> op(op(2, op(3, B)));
> whattype(op(1, op(2, op(3, B))));
> whattype(op(2, op(2, op(3, B))));
> op(op(3, op(3, B)));
> whattype(op(1, op(3, op(3, B))));
> whattype(op(2, op(3, op(3, B))));
```

Aufgabe 14

```
> restart:
> f := x -> x^5 - 5*x^4 - 10*x^3 + 50*x^2 + 9*x - 45;

> factor(f); # Test: Nullstellen bei -3, -1, 1, 3, 5
> L1 := [ seq(i - 5, i = 0..7) ];
> L1 := [ seq(i, i = -5..2) ]; # Wahlweise
> L1 := [ -5, -4, -3, -2, -1, 0, 1, 2 ]; # Wahlweise
> Liste1 := map(f, L1);
> L2 := [ seq(i - 1, i = 0..8) ];
> L2 := [ seq(i, i = -1..7) ]; # Wahlweise
> L2 := [ -1, 0, 1, 2, 3, 4, 5, 6, 7 ]; # Wahlweise
> Liste2 := map(f, L2);
> A := convert(Liste1, set);
> B := convert(Liste2, set);
> ?union
> C := A union B: 'A' union 'B' = C;
> nops(C);
```

Aufgabe 15

```
> restart:
```

(a)

```
> P[0] := 1;
> P[1] := x;
> for n from 1 to 9 do
    P[n+1] := factor(1/(n+1) * ((2*n+1)*x*P[n] - n*P[n-1]));
  od;
```

(b)

```
> Integral := (m, n) -> int(P[m]*P[n], x = -1..1);
> # Results := seq(seq(Integral(m, n), m = 0..10), n = 0..10);
> allOkay := true:
> for m from 0 to 10 do
    for n from 0 to 10 do
      if (m = n) then
        answer := is(Integral(m, n) = 2 / (2*n + 1));
      else
        answer := is(Integral(m, n) = 0);
       fi;
     allOkay := allOkay and answer;
    end do;
  end do:
> if allOkay then
    print("Formel korrekt.");
  else
    error("Formel nicht korrekt.");
  end if;
```

(c)

```
> with(plots):
> colors := [ 'green', 'blue', 'yellow', 'red', 'black' ];
> p = []:
> for n from 0 to 4 do
    p[n] := plot(P[n], x = -1..1, color = colors[n + 1]);
  end do:
> display([ seq(p[n], n = 0..4) ]);
> # oder direkter
> plot([ P[0], P[1], P[2], P[3], P[4] ], x = -1..1, color = [
  'green', 'blue', 'yellow', 'red', 'black' ]);
> # oder
> plot([ seq(P[i], i = 0..4) ], x = -1..1, color = colors);
```

Aufgabe 16

```
> restart:
```

(a)

```
> # Ausgabe wird durch ':' unterdrückt
> f[0] := 1:
> f[1] := 1:
> for kk from 2 to 100 do
    f[kk] := f[kk-1] + f[kk-2];
  od:
> # # Ausgabe, wenn man möchte
> # [ seq(f[kk], kk = 0..100) ];
```

(b)

```
> ?ithprime
> Start := 9; Ende := 25;
> primes := [ seq(ithprime(kk), kk = Start..Ende) ];
> # Anzahl testen
> nops(primes) = Ende - Start + 1;
> for kk in primes do
    f[kk];
  od;
```