```
> 
# ####################
# Uebung 11, Aufgabe 1
# ####################

> 
restart:
with(linalg):

> 
# Die niedrigsten beiden Zustaende des 1-dimensionalen HO.

assume(m>0):
assume(omega>0):
assume(h>0):

eta_0 := (m*omega/(Pi*h))^(1/4) * exp(-(m*omega/(2*h))*u^2):
eta_1 := (m*omega/(Pi*h))^(1/4) * exp(-(m*omega/(2*h))*u^2) * sqrt
(2*m*omega/h)*u:

> 
phi := array(1..4):

> 
# "Kartesische Eigenfunktionen".

phi[1] := subs(u=x, eta_0) * subs(u=y, eta_0) * subs(u=z, eta_0):

phi[2] := subs(u=x, eta_1) * subs(u=y, eta_0) * subs(u=z, eta_0):
phi[3] := subs(u=x, eta_0) * subs(u=y, eta_1) * subs(u=z, eta_0):
phi[4] := subs(u=x, eta_0) * subs(u=y, eta_0) * subs(u=z, eta_1):

> 
# Matrixdarstellung des 1-Operators (Test der Orthonormalitaet).

M_1 := matrix(4, 4):

for i1 from 1 to 4 do
  for i2 from 1 to 4 do
    M_1[i1,i2] := int(int(int(phi[i1] * 1 * phi[i2], x=-infinity..
+infinity),
      y=-infinity..+infinity), z=-infinity..+infinity);
  od:
od:

evalm(M_1);
```

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

```
> 
# Matrixdarstellung des Hamilton-Operators.

M_H := matrix(4, 4):

for i1 from 1 to 4 do
  for i2 from 1 to 4 do
    M_H[i1,i2] := int(int(int(phi[i1] *
      (-(h^2/(2*m)) * (diff(diff(phi[i2], x), x) +
                      diff(diff(phi[i2], y), y) +
                      diff(diff(phi[i2], z), z)) +
      (m*omega^2/2) * (x^2 + y^2 + z^2) * phi[i2]),
      x=-infinity..+infinity), y=-infinity..+infinity), z=-
infinity..+infinity);
  od:
od:

evalm(M_H);
```

$$\begin{bmatrix} \frac{3}{2}\,\omega\sim h\sim & 0 & 0 & 0 \\ 0 & \frac{5}{2}\,\omega\sim h\sim & 0 & 0 \\ 0 & 0 & \frac{5}{2}\,\omega\sim h\sim & 0 \\ 0 & 0 & 0 & \frac{5}{2}\,\omega\sim h\sim \end{bmatrix} \tag{2}$$

```
> 
# Matrixdarstellung von L_x, L_y und L_z.

M_L_x := matrix(4, 4):
M_L_y := matrix(4, 4):
M_L_z := matrix(4, 4):

for i1 from 1 to 4 do
  for i2 from 1 to 4 do

    M_L_x[i1,i2] := int(int(int(phi[i1] *
      (y * (-I*h*diff(phi[i2], z)) - z * (-I*h*diff(phi[i2], y))),
      x=-infinity..+infinity), y=-infinity..+infinity), z=-
infinity..+infinity);

    M_L_y[i1,i2] := int(int(int(phi[i1] *
      (z * (-I*h*diff(phi[i2], x)) - x * (-I*h*diff(phi[i2], z))),
      x=-infinity..+infinity), y=-infinity..+infinity), z=-
infinity..+infinity);

    M_L_z[i1,i2] := int(int(int(phi[i1] *
      (x * (-I*h*diff(phi[i2], y)) - y * (-I*h*diff(phi[i2], x))),
      x=-infinity..+infinity), y=-infinity..+infinity), z=-
infinity..+infinity);

  od:
od:

evalm(M_L_x);
evalm(M_L_y);
evalm(M_L_z);
```

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\mathrm{I}\,h\sim \\ 0 & 0 & \mathrm{I}\,h\sim & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathrm{I}\,h\sim \\ 0 & 0 & 0 & 0 \\ 0 & -\mathrm{I}\,h\sim & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -\mathrm{I}\,h\sim & 0 \\ 0 & \mathrm{I}\,h\sim & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3}$$

```
> 
# Matrixdarstellung von L^2.

M_L_2 := multiply(M_L_x, M_L_x) + multiply(M_L_y, M_L_y) +
multiply(M_L_z, M_L_z):

evalm(M_L_2);
```

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2\,h\sim^2 & 0 & 0 \\ 0 & 0 & 2\,h\sim^2 & 0 \\ 0 & 0 & 0 & 2\,h\sim^2 \end{bmatrix} \tag{4}$$

```
> 
# Kommutatoren checken (1).

# [H, L^2]
evalm(multiply(M_H, M_L_2) - multiply(M_L_2, M_H));

# [H, L_z]
evalm(multiply(M_H, M_L_z) - multiply(M_L_z, M_H));

# [L^2, L_z]
evalm(multiply(M_L_2, M_L_z) - multiply(M_L_z, M_L_2));
```

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5}$$

```
> 
# Kommutatoren checken (2).

# [L_x, L_y]
evalm(multiply(M_L_x, M_L_y) - multiply(M_L_y, M_L_x) - I*h*M_L_z)
;

# [L_y, L_z]
evalm(multiply(M_L_y, M_L_z) - multiply(M_L_z, M_L_y) - I*h*M_L_x)
;
```

```
# [L_z, L_x]
evalm(multiply(M_L_z, M_L_x) - multiply(M_L_x, M_L_z) - I*h*M_L_y)
;
```

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{6}$$

```
> # L_z diagonalisieren.


# Eigenwerte und Eigenvektoren von L_z in obiger Darstellung sind
offensichtlich:

#    0 -->               ( 1,  0,  0,  0)
#   +h --> (1/sqrt(2)) * ( 0,  1, +I,  0)
#    0 -->               ( 0,  0,  0,  1)
#   -h --> (1/sqrt(2)) * ( 0,  1, -I,  0)


# Transformationsmatrix.

T := matrix(4, 4, [
  [ 1,          0,  0,          0],
  [ 0,  1/sqrt(2),  0,  1/sqrt(2)],
  [ 0, +I/sqrt(2),  0, -I/sqrt(2)],
  [ 0,          0,  1,          0]]):

evalm(T);

# Diagonalisiertes L_z.

evalm(multiply(multiply(htranspose(T), M_L_z), T));
```

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2}\sqrt{2} & 0 & \frac{1}{2}\sqrt{2} \\ 0 & \frac{1}{2}I\sqrt{2} & 0 & -\frac{1}{2}I\sqrt{2} \\ 0 & 0 & 1 & 0 \end{bmatrix}$$
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & h\sim & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -h\sim \end{bmatrix} \tag{7}$$

```
> # Vergleich mit "sphaerischen Eigenfunktionen"

# Eigenfunktionen zu L_z sind durch T^T phi gegeben; diese muessen
[bis auf Phasenfaktoren]
# mit den Ergebnissen der 6. Uebung uebereinstimmen.


# Kugelflaechenfunktionen.

Y_00 := 1 / sqrt(4*Pi):

Y_1_p1 := sqrt(3/(8*Pi)) * (x + I*y) / sqrt(x^2+y^2+z^2):
Y_1__0 := sqrt(3/(4*Pi)) *         z / sqrt(x^2+y^2+z^2):
Y_1_m1 := sqrt(3/(8*Pi)) * (x - I*y) / sqrt(x^2+y^2+z^2):


# Radialwellenfunktionen.

R_00 := simplify(
 subs(b=sqrt(h/(m*omega)), subs(y=r/b, y * exp(-y^2/2))) / r
  ):

norm_00 := simplify(sqrt(int((r*R_00)^2, r=0..infinity))):

R_1_ := simplify(
 subs(b=sqrt(h/(m*omega)), subs(y=r/b, y^2 * exp(-y^2/2))) / r
  ):

norm_1_ := simplify(sqrt(int((r*R_1_)^2, r=0..infinity))):


# Die vollstaendigen Wellenfunktionen (T * phi).

# l=0, m=0
phi_00 := subs(r=sqrt(x^2+y^2+z^2), R_00 * Y_00 / norm_00):
simplify(phi_00 - phi[1]);

# l=1, m=+1
phi_1_p1 := subs(r=sqrt(x^2+y^2+z^2), R_1_ * Y_1_p1 / norm_1_):
simplify(phi_1_p1 - ( (1/sqrt(2)) * phi[2] + (I/sqrt(2)) * phi[3]
));

# l=1, m=0
phi_1__0 := subs(r=sqrt(x^2+y^2+z^2), R_1_ * Y_1__0 / norm_1_):
simplify(phi_1__0 - phi[4]);

# l=1, m=-1
phi_1_m1 := subs(r=sqrt(x^2+y^2+z^2), R_1_ * Y_1_m1 / norm_1_):
simplify(phi_1_m1 - ( (1/sqrt(2)) * phi[2] - (I/sqrt(2)) * phi[3]
));
```

$$0$$
$$0$$
$$0$$
$$0 \tag{8}$$

```
>
```