Wykład 4

Listy, Tabele etc.

In[1]:=
```
ClearAll["Global`*"]
```

Listy

Uwagi ogólne

Uwaga: często wyniki operacji są przedstawiane w postaci listy

In[2]:=
```
Head[{}]
```

Out[2]=
List

In[3]:=
```
roz = Solve[x^2 + 2 x + 5 == 1, x]
```

Out[3]=
$$\{\{x \to -1 - i\sqrt{3}\}, \{x \to -1 + i\sqrt{3}\}\}$$

In[4]:=
```
roz[[1]][[1]][[2]]
```

Out[4]=
$$-1 - i\sqrt{3}$$

In[5]:=
```
x /. Part[roz, 1]
```

Out[5]=
$$-1 - i\sqrt{3}$$

In[6]:=
```
Last[Last[Part[roz, 1]]]
```

Out[6]=
$$-1 - i\sqrt{3}$$

In[7]:=
```
roz = DSolve[{x''[t]^2 + 2 x'[t] + x[t] == 1, x'[0] == 1, x[0] == 0}, x[t], t]
```

Out[7]=
$$\text{DSolve}\left[\{x''(t)^2 + 2\,x'(t) + x(t) = 1,\ x'(0) = 1,\ x(0) = 0\},\ x(t),\ t\right]$$

Operacje na listach można wykonywać w niezmiernie prosty sposób

In[8]:=
```
{1, 2, 3, 4}^3 - 1
```
Out[8]=
$\{0, 7, 26, 63\}$

In[9]:=
```
{1, 2, 3, 4} - 2
```
Out[9]=
$\{-1, 0, 1, 2\}$

In[10]:=
```
#^3 - 1 & /@ {1, 2, 3, 4}
```
Out[10]=
$\{0, 7, 26, 63\}$

In[11]:=
```
{1, 2, 3, 4} - {2, 4}
```
Thread: Objects of unequal length in $\{1, 2, 3, 4\} + \{-2, -4\}$ cannot be combined.

Out[11]=
$\{-2, -4\} + \{1, 2, 3, 4\}$

In[12]:=
```
1 / {1, 2, 3, 4}
```
Out[12]=
$\left\{1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}\right\}$

In[13]:=
```
{0.1, 0.5, 1, 2} + {1, 2, 3, 4}^3 - 1
```
Out[13]=
$\{0.1, 7.5, 27, 65\}$

In[14]:=
```
Log[%] // N
```
Out[14]=
$\{-2.30259, 2.0149, 3.29584, 4.17439\}$

In[15]:=
```
? List
```
$\{e_1, e_2, ...\}$ is a list of elements. >>

In[16]:=
```
{A, B} // FullForm
```
Out[16]//FullForm=
List[A, B]

In[17]:=
```
ListQ[{1, 2}]
```
Out[17]=
True

In[18]:=
```
ListQ[1]
```

Out[18]=
```
False
```

Tabl

In[19]:=
```
? Table
```

Table[*expr*, *n*] generates a list of *n* copies of *expr*.
Table[*expr*, {*i*, i_{max}}] generates a list of the values of *expr* when *i* runs from 1 to i_{max}.
Table[*expr*, {*i*, i_{min}, i_{max}}] starts with *i* = i_{min}.
Table[*expr*, {*i*, i_{min}, i_{max}, *di*}] uses steps *di*.
Table[*expr*, {*i*, {i_1, i_2, ...}}] uses the successive values i_1, i_2, ….
Table[*expr*, {*i*, i_{min}, i_{max}}, {*j*, j_{min}, j_{max}}, ...] gives a nested list. The list associated with *i* is outermost. ≫

In[20]:=
```
m4 =.
m4 = Table[Table[n ^k, {n, 1, 6}], {k, 1, 6}]
% // MatrixForm
```

Out[21]=

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 4 & 9 & 16 & 25 & 36 \\ 1 & 8 & 27 & 64 & 125 & 216 \\ 1 & 16 & 81 & 256 & 625 & 1296 \\ 1 & 32 & 243 & 1024 & 3125 & 7776 \\ 1 & 64 & 729 & 4096 & 15\,625 & 46\,656 \end{pmatrix}$$

Out[22]//MatrixForm=

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 4 & 9 & 16 & 25 & 36 \\ 1 & 8 & 27 & 64 & 125 & 216 \\ 1 & 16 & 81 & 256 & 625 & 1296 \\ 1 & 32 & 243 & 1024 & 3125 & 7776 \\ 1 & 64 & 729 & 4096 & 15\,625 & 46\,656 \end{pmatrix}$$

In[23]:=
```
Table[i^3, {i, Array[Prime, 10]}]
```

Out[23]=
```
{8, 27, 125, 343, 1331, 2197, 4913, 6859, 12 167, 24 389}
```

In[24]:=
```
m4 // TableForm
```

Out[24]//TableForm=

1	2	3	4	5	6
1	4	9	16	25	36
1	8	27	64	125	216
1	16	81	256	625	1296
1	32	243	1024	3125	7776
1	64	729	4096	15 625	46 656

In[25]:=
```
TableForm[m4, TableAlignments → Center]
```

Out[25]//TableForm=

1	2	3	4	5	6
1	4	9	16	25	36
1	8	27	64	125	216
1	16	81	256	625	1296
1	32	243	1024	3125	7776
1	64	729	4096	15 625	46 656

In[26]:=
```
TableForm[m4, TableAlignments → Center, TableHeadings →
  {{"w1", "w2", "w3", "w4", "w5", "w6"}, {"n^1", "n^2", "n^3", "n^4", "n^5", "n^6"}}]
```

Out[26]//TableForm=

	n^1	n^2	n^3	n^4	n^5	n^6
w1	1	2	3	4	5	6
w2	1	4	9	16	25	36
w3	1	8	27	64	125	216
w4	1	16	81	256	625	1296
w5	1	32	243	1024	3125	7776
w6	1	64	729	4096	15 625	46 656

In[27]:=
```
TeXForm[%]
```

Out[27]//TeXForm=
```
\left(
\begin{array}{cccccc}
 1 & 2 & 3 & 4 & 5 & 6 \\
 1 & 4 & 9 & 16 & 25 & 36 \\
 1 & 8 & 27 & 64 & 125 & 216 \\
 1 & 16 & 81 & 256 & 625 & 1296 \\
 1 & 32 & 243 & 1024 & 3125 & 7776 \\
 1 & 64 & 729 & 4096 & 15625 & 46656 \\
\end{array}
\right)
```

In[28]:=
```
Table[f[k], {k, 1, 10, 2}]
```

Out[28]= $\{f(1), f(3), f(5), f(7), f(9)\}$

In[29]:=
```
f[k_] := k^2
```

In[30]:=
```
Table[f[k], {k, 1, 10, 2}]
```

Out[30]= {1, 9, 25, 49, 81}

Array

In[31]:=
```
? Array
```

Array[f, n] generates a list of length n, with elements $f[i]$.
Array[f, n, r] generates a list using the index origin r.
Array[f, n, {a, b}] generates a list using n values from a to b.
Array[f, {n_1, n_2, ...}] generates an $n_1 \times n_2 \times \ldots$ array of nested lists, with elements $f[i_1, i_2, \ldots]$.
Array[f, {n_1, n_2, ...}, {r_1, r_2, ...}] generates a list using the index origins r_i (default 1).
Array[f, {n_1, n_2, ...}, {{a_1, b_1}, {a_2, b_2}, ...}] generates a list using n_i values from a_i to b_i.
Array[f, *dims*, *origin*, h] uses head h, rather than List, for each level of the array. ≫

In[32]:=
```
Array[ff, 10]
```

Out[32]=
{ff(1), ff(2), ff(3), ff(4), ff(5), ff(6), ff(7), ff(8), ff(9), ff(10)}

In[33]:=
```
Array[f, 10]
```

Out[33]=
{1, 4, 9, 16, 25, 36, 49, 64, 81, 100}

In[34]:=
```
Array[#^3 &, 10]
```

Out[34]=
{1, 8, 27, 64, 125, 216, 343, 512, 729, 1000}

In[35]:=
```
Array[Sin, 10]
% // FunctionExpand
Array[Sin, 10] // N
```

Out[35]=
{sin(1), sin(2), sin(3), sin(4), sin(5), sin(6), sin(7), sin(8), sin(9), sin(10)}

Out[36]=
{sin(1), sin(2), sin(3), sin(4), sin(5), sin(6), sin(7), sin(8), sin(9), sin(10)}

Out[37]=
{0.841471, 0.909297, 0.14112, −0.756802, −0.958924, −0.279415, 0.656987, 0.989358, 0.412118, −0.544021}

In[38]:=
```
Array[f, {3, 2}]
% // MatrixForm
```

Out[38]=
$$\begin{pmatrix} f(1,1) & f(1,2) \\ f(2,1) & f(2,2) \\ f(3,1) & f(3,2) \end{pmatrix}$$

Out[39]//MatrixForm=
$$\begin{pmatrix} f(1,1) & f(1,2) \\ f(2,1) & f(2,2) \\ f(3,1) & f(3,2) \end{pmatrix}$$

In[40]:=
```
Array[a_{#1,#2} &, {3, 2}]
```

Out[40]=
$$\begin{pmatrix} a_{\{1,1\}} & a_{\{1,2\}} \\ a_{\{2,1\}} & a_{\{2,2\}} \\ a_{\{3,1\}} & a_{\{3,2\}} \end{pmatrix}$$

In[41]:=
```
Array[f, 10, 0]
```

Out[41]=
{0, 1, 4, 9, 16, 25, 36, 49, 64, 81}

Range

In[42]:=
```
Range[10]
```

Out[42]=
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

In[43]:=
```
Select[#, # > 5 &] &@ (Prime /@ (Range@10))
```

Out[43]=
{7, 11, 13, 17, 19, 23, 29}

In[44]:=
```
Range[3, 10]
```

Out[44]=
{3, 4, 5, 6, 7, 8, 9, 10}

In[45]:=
```
Range[3, 10, 2]
```

Out[45]=
{3, 5, 7, 9}

In[46]:=
```
First[%]
```

Out[46]=
3

In[47]:=
```
Last[%%]
```

Out[47]=
9

Operacje na listach

In[48]:=
```
Clear[f]
```

In[49]:=
```
Lista = {a, b, c, d, e, f, g, h}
```

Out[49]=
{*a*, *b*, *c*, *d*, *e*, *f*, *g*, *h*}

In[50]:=
```
Take[Lista, 3]
```
Out[50]= $\{a, b, c\}$

In[51]:=
```
Take[Lista, {3}]
```
Out[51]= $\{c\}$

In[52]:=
```
Take[Lista, {3, 5}]
```
Out[52]= $\{c, d, e\}$

In[53]:=
```
Take[Lista, {3, 5, 2}]
```
Out[53]= $\{c, e\}$

In[54]:=
```
? Take
```

Take[*list*, *n*] gives the first *n* elements of *list*.
Take[*list*, –*n*] gives the last *n* elements of *list*.
Take[*list*, {*m*, *n*}] gives elements *m* through *n* of *list*.
Take[*list*, seq_1, seq_2, ...] gives a nested list in which elements specified by seq_i are taken at level *i* in *list*. ≫

In[55]:=
```
? Part
```

expr[[*i*]] or Part[*expr*, *i*] gives the i^{th} part of *expr*.
expr[[–*i*]] counts from the end.
expr[[*i*, *j*, ...]] or Part[*expr*, *i*, *j*, ...] is equivalent to *expr*[[*i*]][[*j*]] ....
expr[[{i_1, i_2, ...}]] gives a list of the parts i_1, i_2, ... of *expr*.
expr[[*m* ;; *n*]] gives parts *m* through *n*.
expr[[*m* ;; *n* ;; *s*]] gives parts *m* through *n* in steps of *s*.
expr[["*key*"]] gives the value associated with the key "*key*" in an association *expr*.
expr[[Key[*k*]]] gives the value associated with an arbitrary key *k* in the association *expr*. ≫

In[56]:=
```
Part[Lista, 2]
```
Out[56]= b

In[57]:=
```
Part[Lista, -2]
```
Out[57]= g

In[58]:=
```
? Lista
```

```
Global`Lista
```

```
Lista = {a, b, c, d, e, f, g, h}
```

In[59]:=
```
? Drop
```

Drop[*list*, *n*] gives *list* with its first *n* elements dropped.
Drop[*list*, –*n*] gives *list* with its last *n* elements dropped.
Drop[*list*, {*n*}] gives *list* with its n^{th} element dropped.
Drop[*list*, {*m*, *n*}] gives *list* with elements *m* through *n* dropped.
Drop[*list*, {*m*, *n*, *s*}] gives *list* with elements *m* through *n* in steps of *s* dropped.
Drop[*list*, seq_1, seq_2, ...] gives a nested list
in which elements specified by seq_i have been dropped at level *i* in *list*. ≫

In[60]:=
```
Drop[Lista, 2]
```
Out[60]=
$\{c, d, e, f, g, h\}$

In[61]:=
```
Drop[Lista, -2]
```
Out[61]=
$\{a, b, c, d, e, f\}$

In[62]:=
```
Delete[Lista, 2]
```
Out[62]=
$\{a, c, d, e, f, g, h\}$

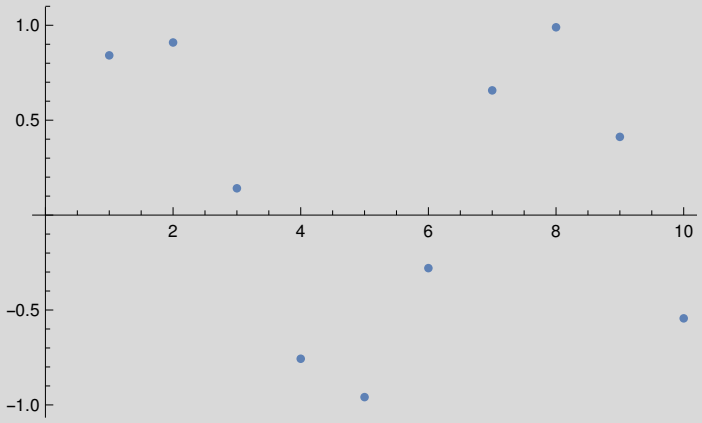
In[63]:=
```
Delete[Lista, -2]
```
Out[63]=
$\{a, b, c, d, e, f, h\}$

In[64]:=
```
Clear[Lista]
```

In[65]:=
```
Lista = Array[Sin, 10] // N
```
Out[65]=
{0.841471, 0.909297, 0.14112, –0.756802, –0.958924, –0.279415, 0.656987, 0.989358, 0.412118, –0.544021}

In[66]:=
```
ListPlot[Lista]
```

Out[66]=

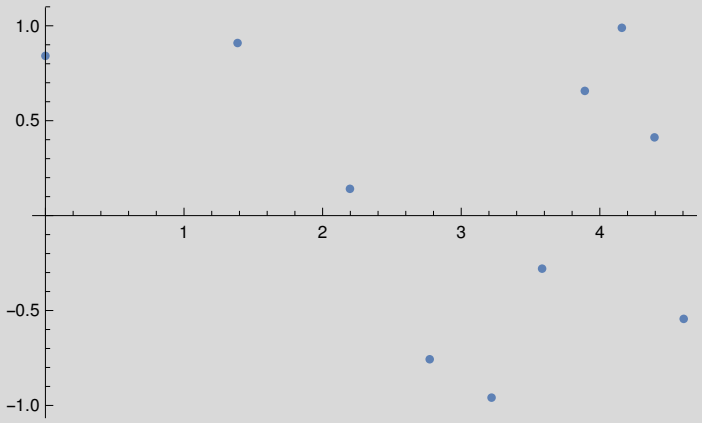


In[67]:=
```
Array[N@{Log[#^2], Sin[#]} &, 10]
ListPlot[%]
```

Out[67]=

0.	0.841471
1.38629	0.909297
2.19722	0.14112
2.77259	−0.756802
3.21888	−0.958924
3.58352	−0.279415
3.89182	0.656987
4.15888	0.989358
4.39445	0.412118
4.60517	−0.544021

Out[68]=



In[69]:=
```
? Extract
```

Extract[*expr*, *list*] extracts the part of *expr* at the position specified by *list*.
Extract[*expr*, {$list_1$, $list_2$, ...}] extracts a list of parts of *expr*.
Extract[*expr*, *list*, *h*] extracts parts of *expr*, wrapping each of them with head *h* before evaluation.
Extract[*list*] represents an operator form of Extract that can be applied to an expression. ≫

In[70]:=
```
? Lista
```

```
Global`Lista
```

```
Lista = {0.841471, 0.909297, 0.14112, -0.756802,
  -0.958924, -0.279415, 0.656987, 0.989358, 0.412118, -0.544021}
```

In[71]:=
```
Extract[Lista, 3]
```

Out[71]=
```
0.14112
```

In[72]:=
```
ConstantArray[10, {5, 2}]
```

Out[72]=
$$\begin{pmatrix} 10 & 10 \\ 10 & 10 \\ 10 & 10 \\ 10 & 10 \\ 10 & 10 \end{pmatrix}$$

In[73]:=
```
Join[{A1, A2, A3}, {A1, B1, B2}]
```

Out[73]=
```
{A1, A2, A3, A1, B1, B2}
```

In[74]:=
```
Union[{A1, A2, A3}, {A1, B1, B2}]
```

Out[74]=
```
{A1, A2, A3, B1, B2}
```

In[75]:=
```
Join[{A1, A2, A3}, {{B1, B2}, C1}]
```

Out[75]=
```
{A1, A2, A3, {B1, B2}, C1}
```

Filtrowanie list

In[76]:=
```
Lista =.
```

In[77]:=
```
Lista = {1.0, 2, 3 / 2, A, b , 1 + c}
```

Out[77]=
$$\left\{1., 2, \frac{3}{2}, A, b, c+1\right\}$$

In[78]:=
```
? DeleteCases
```

DeleteCases[*expr*, *pattern*] removes all elements of *expr* that match *pattern*.
DeleteCases[*expr*, *pattern*, *levelspec*]
removes all parts of *expr* on levels specified by *levelspec* that match *pattern*.
DeleteCases[*expr*, *pattern*, *levelspec*, *n*] removes the first *n* parts of *expr* that match *pattern*.
DeleteCases[*pattern*] represents an operator form of DeleteCases that can be applied to an expression. >>

In[79]:=
```
DeleteCases[Lista, _Integer]
```

Out[79]=
$$\left\{1., \frac{3}{2}, A, b, c+1\right\}$$

In[80]:=
```
DeleteCases[1.0 x + 2 y + 3, _Integer]
```

Out[80]=
$$1. x + 2 y$$

In[81]:=
```
Take[1.0 x + 2 y + 3, 2]
```

Out[81]=
$$1. x + 3$$

In[82]:=
```
? Lista
```

```
Global`Lista
```

```
Lista = {1., 2, 3/2, A, b, 1 + c}
```

In[83]:=
```
Select[Lista, IntegerQ]
```

Out[83]=
$$\{2\}$$

In[84]:=
```
DeleteCases[Lista, _Rational]
```

Out[84]=
$$\{1., 2, A, b, c+1\}$$

In[85]:=
```
DeleteCases[Lista, _Real]
```

Out[85]=
$$\left\{2, \frac{3}{2}, A, b, c+1\right\}$$

Append, Prepend

In[86]:=
```
Lista =.
Lista = {1, 2, 3}
Prepend[Lista, 10]
Lista
```

Out[87]= {1, 2, 3}

Out[88]= {10, 1, 2, 3}

Out[89]= {1, 2, 3}

In[90]:=
```
PrependTo[Lista, 10]
Lista
```

Out[90]= {10, 1, 2, 3}

Out[91]= {10, 1, 2, 3}

In[92]:=
```
? Append
```

Append[*expr*, *elem*] gives *expr* with *elem* appended.
Append[*elem*] represents an operator form of Append that can be applied to an expression. ≫

In[93]:=
```
? AppendTo
```

AppendTo[*s*, *elem*] appends *elem* to the value of *s*, and resets *s* to the result. ≫

In[94]:=
```
? Insert
```

Insert[*list*, *elem*, *n*] inserts *elem* at position *n* in *list*. If *n* is negative, the position is counted from the end.
Insert[*expr*, *elem*, {*i*, *j*, ...}] inserts *elem* at position {*i*, *j*, ...} in *expr*.
Insert[*expr*, *elem*, {{i_1, j_1, ...}, {i_2, j_2, ...}, ...}] inserts *elem* at several positions.
Insert[*elem*, *pos*] represents an operator form of Insert that can be applied to an expression. ≫

In[95]:=
```
Insert[{A, B, F}, GGGGG, 2]
```

Out[95]= {*A*, GGGGG, *B*, *F*}

Zmiana wartości listy

In[96]:=
```
Lista
```
Out[96]=
```
{10, 1, 2, 3}
```

In[97]:=
```
Lista[[2]] = 100
Lista
```
Out[97]=
```
100
```
Out[98]=
```
{10, 100, 2, 3}
```

In[99]:=
```
Part[Lista, 3] = 300
Lista
```
Out[99]=
```
300
```
Out[100]=
```
{10, 100, 300, 3}
```

In[101]:=
```
ReplacePart[Lista, "Graczyk", 1]
```
Out[101]=
```
{Graczyk, 100, 300, 3}
```

Zapisy skrótowe

In[102]:=
```
MojaLista = Array[A, 10]
```
Out[102]=
```
{A(1), A(2), A(3), A(4), A(5), A(6), A(7), A(8), A(9), A(10)}
```

In[103]:=
```
MojaLista[[2 ;; 6]]
```
Out[103]=
```
{A(2), A(3), A(4), A(5), A(6)}
```

In[104]:=
```
MojaLista[[1 ;; 9 ;; 3]]
```
Out[104]=
```
{A(1), A(4), A(7)}
```

In[105]:=
```
MojaLista[[3 ;;  ;; 2]]
```
Out[105]=
```
{A(3), A(5), A(7), A(9)}
```

In[106]:=
```
MojaLista[[ ;; 7 ;; 3]]
```

Out[106]=
$\{A(1), A(4), A(7)\}$

In[107]:=
```
MojaLista[[ ;;  ;; 3]]
```

Out[107]=
$\{A(1), A(4), A(7), A(10)\}$

In[108]:=
```
(Range@100)[[3 ;;  ;; 3]]
```

Out[108]=
{3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45,
 48, 51, 54, 57, 60, 63, 66, 69, 72, 75, 78, 81, 84, 87, 90, 93, 96, 99}

Kolejne funkcje na listach

In[109]:=
```
PadLeft[Lista, 20]
```

Out[109]=
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 10, 100, 300, 3}

In[110]:=
```
N@(Sin /@ Range@10)
```

Out[110]=
{0.841471, 0.909297, 0.14112, −0.756802, −0.958924, −0.279415, 0.656987, 0.989358, 0.412118, −0.544021}

In[111]:=
```
? RealDigits
```

RealDigits[*x*] gives a list of the digits in the approximate real number
x, together with the number of digits that are to the left of the decimal point.
RealDigits[*x*, *b*] gives a list of base-*b* digits in *x*.
RealDigits[*x*, *b*, *len*] gives a list of *len* digits.
RealDigits[*x*, *b*, *len*, *n*] gives *len* digits starting with the coefficient of b^n. ≫

In[112]:=
```
RealDigits@(N@(Sin /@ Range@10))
```

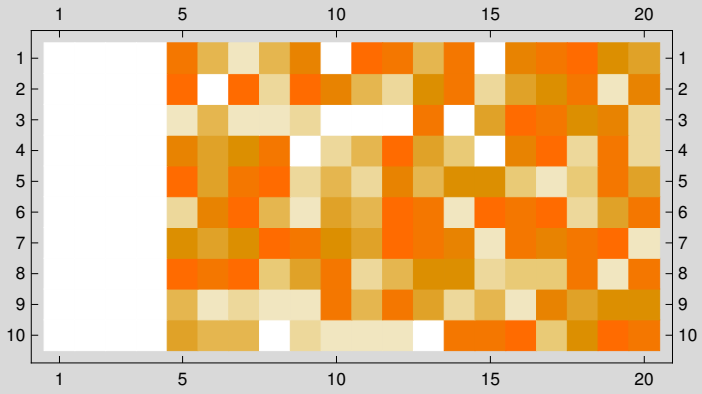
Out[112]=
$$\begin{pmatrix} \{8,4,1,4,7,0,9,8,4,8,0,7,8,9,6,5\} & 0 \\ \{9,0,9,2,9,7,4,2,6,8,2,5,6,8,1,7\} & 0 \\ \{1,4,1,1,2,0,0,0,8,0,5,9,8,6,7,2\} & 0 \\ \{7,5,6,8,0,2,4,9,5,3,0,7,9,2,8,2\} & 0 \\ \{9,5,8,9,2,4,2,7,4,6,6,3,1,3,8,5\} & 0 \\ \{2,7,9,4,1,5,4,9,8,1,9,8,9,2,5,8\} & 0 \\ \{6,5,6,9,8,6,5,9,8,7,1,8,7,8,9,1\} & 0 \\ \{9,8,9,3,5,8,2,4,6,6,2,3,3,8,1,8\} & 0 \\ \{4,1,2,1,1,8,4,8,5,2,4,1,7,5,6,6\} & 0 \\ \{5,4,4,0,2,1,1,1,0,8,8,9,3,6,9,8\} & 0 \end{pmatrix}$$

In[113]:=
```
PadLeft[#, 20] & /@ First /@ RealDigits@(N@(Sin /@ Range@10))
MatrixPlot[%]
```

Out[113]=

$$\begin{pmatrix}
0 & 0 & 0 & 0 & 8 & 4 & 1 & 4 & 7 & 0 & 9 & 8 & 4 & 8 & 0 & 7 & 8 & 9 & 6 & 5 \\
0 & 0 & 0 & 0 & 9 & 0 & 9 & 2 & 9 & 7 & 4 & 2 & 6 & 8 & 2 & 5 & 6 & 8 & 1 & 7 \\
0 & 0 & 0 & 0 & 1 & 4 & 1 & 1 & 2 & 0 & 0 & 0 & 8 & 0 & 5 & 9 & 8 & 6 & 7 & 2 \\
0 & 0 & 0 & 0 & 7 & 5 & 6 & 8 & 0 & 2 & 4 & 9 & 5 & 3 & 0 & 7 & 9 & 2 & 8 & 2 \\
0 & 0 & 0 & 0 & 9 & 5 & 8 & 9 & 2 & 4 & 2 & 7 & 4 & 6 & 6 & 3 & 1 & 3 & 8 & 5 \\
0 & 0 & 0 & 0 & 2 & 7 & 9 & 4 & 1 & 5 & 4 & 9 & 8 & 1 & 9 & 8 & 9 & 2 & 5 & 8 \\
0 & 0 & 0 & 0 & 6 & 5 & 6 & 9 & 8 & 6 & 5 & 9 & 8 & 7 & 1 & 8 & 7 & 8 & 9 & 1 \\
0 & 0 & 0 & 0 & 9 & 8 & 9 & 3 & 5 & 8 & 2 & 4 & 6 & 6 & 2 & 3 & 3 & 8 & 1 & 8 \\
0 & 0 & 0 & 0 & 4 & 1 & 2 & 1 & 1 & 8 & 4 & 8 & 5 & 2 & 4 & 1 & 7 & 5 & 6 & 6 \\
0 & 0 & 0 & 0 & 5 & 4 & 4 & 0 & 2 & 1 & 1 & 1 & 0 & 8 & 8 & 9 & 3 & 6 & 9 & 8
\end{pmatrix}$$

Out[114]=



In[115]:=
```
PadRight[Lista, 20]
```

Out[115]=

{10, 100, 300, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}

In[116]:=
```
Dimensions[%]
```

Out[116]=

{20}

In[117]:=
```
{{A, B}, {C, {D, {help}}}}
Dimensions[%]
```

Out[117]=

$$\begin{pmatrix} A & B \\ C & \{D, \{\text{help}\}\} \end{pmatrix}$$

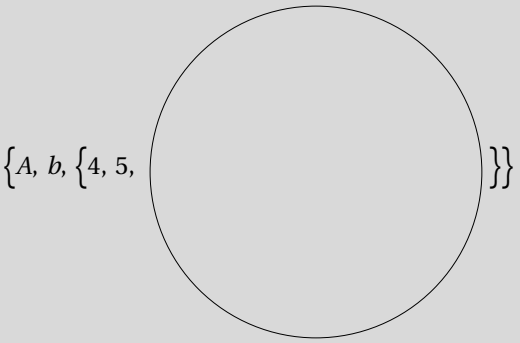
Out[118]=

{2, 2}

In[119]:=
```
Lista =.
```

In[120]:=
```
Lista = {A, b, {4, 5, Graphics[Circle[]]}}
```

Out[120]=



In[121]:=
```
Length[Lista]
```

Out[121]=
```
3
```

In[122]:=
```
ListaB = {{A, B}, {c, d}}
```

Out[122]=

$$\begin{pmatrix} A & B \\ c & d \end{pmatrix}$$

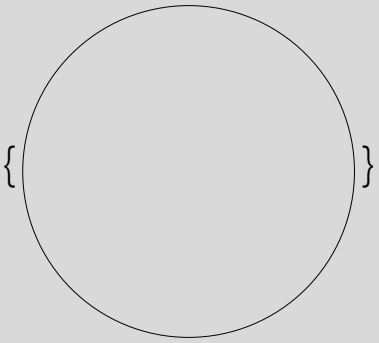
In[123]:=
```
Dimensions[ListaB]
```

Out[123]=
```
{2, 2}
```

In[124]:=
```
Position[Lista, Graphics[Circle[]]]
Extract[Lista, %]
```

Out[124]=
```
(3  3)
```

Out[125]=



In[126]:=
```
? Extract
```

Extract[*expr*, *list*] extracts the part of *expr* at the position specified by *list*.
Extract[*expr*, {$list_1$, $list_2$, ...}] extracts a list of parts of *expr*.
Extract[*expr*, *list*, *h*] extracts parts of *expr*, wrapping each of them with head *h* before evaluation.
Extract[*list*] represents an operator form of Extract that can be applied to an expression. >>

In[127]:=
```
ListaC = {{A, B}, {c, d, 3, 3, 3, 3}}
```

Out[127]=
$$\{\{A, B\}, \{c, d, 3, 3, 3, 3\}\}$$

In[128]:=
```
? Count
```

Count[*list*, *pattern*] gives the number of elements in *list* that match *pattern*.
Count[*expr*, *pattern*, *levelspec*] gives the total number of
    subexpressions matching *pattern* that appear at the levels in *expr* specified by *levelspec*.
Count[*pattern*] represents an operator form of Count that can be applied to an expression. >>

In[129]:=
```
Count[ListaC, 3]
```

Out[129]=
$$0$$

In[130]:=
```
Count[ListaC[[2]], 3]
```

Out[130]=
$$4$$

In[131]:=
```
Count[ListaC, 3, Infinity]
```

Out[131]=
$$4$$

In[132]:=
```
a3 x^3 + 3 x^2 + Log[x^3]
Count[%, 3, Infinity]
```

Out[132]=
$$a_3\, x^3 + \log\left(x^3\right) + 3\, x^2$$

Out[133]=
$$4$$

In[134]:=
```
? MemebrQ
```

Information: Symbol MemebrQ not found.

In[135]:=
```
? Lista
```

```
Global`Lista
```

```
Lista = {A, b, {4, 5, Graphics[Circle[{0, 0}]]}}
```

In[136]:=
```
MemberQ[Lista, A]
```
Out[136]=
```
True
```

In[137]:=
```
Reduce[# == 2] & /@ {1, 3, 4, 1, 2}
```
Out[137]=
```
{False, False, False, False, True}
```

In[138]:=
```
MemberQ[{1, 3, 4, 1, 2}, 2]
```
Out[138]=
```
True
```

In[139]:=
```
FreeQ[Lista, A]
```
Out[139]=
```
False
```

In[140]:=
```
? FreeQ
```

FreeQ[*expr*, *form*] yields True if no subexpression in *expr* matches *form*, and yields False otherwise.
FreeQ[*expr*, *form*, *levelspec*] tests only those parts of *expr* on levels specified by *levelspec*.
FreeQ[*form*] represents an operator form of FreeQ that can be applied to an expression. ≫

Sortowanie elementów list

In[141]:=
```
?? Sort
```

Sort[*list*] sorts the elements of *list* into canonical order.
Sort[*list*, *p*] sorts using the ordering function *p*. ≫

```
Attributes[Sort] = {Protected}
```

In[142]:=
```
Clear[Lista]
Lista = {P, r, A, c, G, F}
% // Sort
```
Out[143]=
$\{P, r, A, c, G, F\}$

Out[144]=
$\{A, c, F, G, P, r\}$

In[145]:=
```
Sort[{4, 1, 3, 2, 2}, Greater]
```

Out[145]= {4, 3, 2, 2, 1}

In[146]:=
```
Sort[{4, 1, 3, 2, 2}, Log[#1] > #2 &]
```

Out[146]= {2, 2, 3, 4, 1}

In[147]:=
```
Clear[Lista]
Lista = {5, 7, 1, 3, 100, 0}
% // Sort
```

Out[148]= {5, 7, 1, 3, 100, 0}

Out[149]= {0, 1, 3, 5, 7, 100}

In[150]:=
```
? Permutations
```

Permutations[*list*] generates a list of all possible permutations of the elements in *list*.
Permutations[*list*, *n*] gives all permutations containing at most *n* elements.
Permutations[*list*, {*n*}] gives all permutations containing exactly *n* elements. ≫

In[151]:=
```
Delete[%%, {{1}, {2}, {5}}]
Permutations[%]
```

Out[151]= {3, 5, 100}

Out[152]=
$$\begin{pmatrix} 3 & 5 & 100 \\ 3 & 100 & 5 \\ 5 & 3 & 100 \\ 5 & 100 & 3 \\ 100 & 3 & 5 \\ 100 & 5 & 3 \end{pmatrix}$$

In[153]:=
```
Clear[A, B, a, c]
A = {a, b, c};
B = {b, d, f};
Union[A, B]
Intersection[A, B]
```

Out[156]= {*a*, *b*, *c*, *d*, *f*}

Out[157]= {*b*}

In[158]:=
```
? Partition
```

Partition[*list*, *n*] partitions *list* into nonoverlapping sublists of length *n*.
Partition[*list*, *n*, *d*] generates sublists with offset *d*.
Partition[*list*, {n_1, n_2, ...}] partitions a nested list into blocks of size $n_1 \times n_2 \times \ldots$.
Partition[*list*, {n_1, n_2, ...}, {d_1, d_2, ...}] uses offset d_i at level *i* in *list*.
Partition[*list*, *n*, *d*, {k_L, k_R}] specifies that the first element of *list* should appear at position k_L in the first sublist, and the last element of *list* should appear at or after position k_R in the last sublist. If additional elements are needed, Partition fills them in by treating *list* as cyclic.
Partition[*list*, *n*, *d*, {k_L, k_R}, *x*] pads if necessary by repeating the element *x*.
Partition[*list*, *n*, *d*, {k_L, k_R}, {x_1, x_2, ...}] pads if necessary by cyclically repeating the elements x_i.
Partition[*list*, *n*, *d*, {k_L, k_R}, {}] uses no padding, and so can yield sublists of different lengths.
Partition[*list*, *nlist*, *dlist*, {$klist_L$, $klist_R$}, *padlist*] specifies alignments and padding in a nested list. ≫

In[159]:=
```
Clear[Lista]
Lista = {a, b, c, d, e, f, g, h, i}
Partition[Lista, 3, 2] // InputForm
```

Out[160]=
$\{a, b, c, d, e, f, g, h, i\}$

Out[161]//InputForm=
```
{{a, b, c}, {c, d, e}, {e, f, g}, {g, h, i}}
```

Flatten

In[162]:=
```
? Flatten
```

Flatten[*list*] flattens out nested lists.
Flatten[*list*, *n*] flattens to level *n*.
Flatten[*list*, *n*, *h*] flattens subexpressions with head *h*.
Flatten[*list*, {{s_{11}, s_{12}, ...}, {s_{21}, s_{22}, ...}, ...}]
flattens *list* by combining all levels s_{ij} to make each level *i* in the result. ≫

In[163]:=
```
Clear[L]
L = {{1, 1}, {1}, {{1, {1}, 1}, 1}}
% // Flatten
Flatten[%%, 1]
Flatten[%%%, 2]
```

Out[164]= {{1, 1}, {1}, {{1, {1}, 1}, 1}}

Out[165]= {1, 1, 1, 1, 1, 1, 1}

Out[166]= {1, 1, 1, {1, {1}, 1}, 1}

Out[167]= {1, 1, 1, 1, {1}, 1, 1}

In[168]:=
```
FlattenAt[L, {3, 1, 2}]
```

Out[168]= {{1, 1}, {1}, {{1, 1, 1}, 1}}

In[169]:=
```
Range[4]
```

Out[169]= {1, 2, 3, 4}

In[170]:=
```
{%, A, B, %, %, D}
```

Out[170]= {{1, 2, 3, 4}, {*a*, *b*, *c*}, {*b*, *d*, *f*}, {1, 2, 3, 4}, {1, 2, 3, 4}, *D*}

In[171]:=
```
Flatten[%]
```

Out[171]= {1, 2, 3, 4, *a*, *b*, *c*, *b*, *d*, *f*, 1, 2, 3, 4, 1, 2, 3, 4, *D*}

In[172]:=
```
sol = DSolve[{3 x''[t] - 2 x'[t] + x[t] == 0, x'[0] == 1, x[0] == 0}, x[t], t]
```

Out[172]= $\left\{\left\{x(t) \to \frac{3\, e^{t/3} \sin\left(\frac{\sqrt{2}\, t}{3}\right)}{\sqrt{2}}\right\}\right\}$

In[173]:=
```
x[t]^2 /. %
```

Out[173]= $\left\{\frac{9}{2}\, e^{2\, t/3} \sin^2\left(\frac{\sqrt{2}\, t}{3}\right)\right\}$

In[174]:=
```
x[t]^2 /. Flatten[%%]
```

Out[174]=
$$\frac{9}{2} e^{2\,t/3} \sin^2\left(\frac{\sqrt{2}\,t}{3}\right)$$

Flatten - zastosowanie fizyczne

Zadanie

Stwórz tablę liczb kwantowych w atomie wodoru

n=1, 2, 3, 4, ...

l=0, 1, 2,...,(n-1)

$m_l = -l, -l+1, \ldots, 0, 1, 2, \ldots, +l$

$s = -\frac{1}{2}, +\frac{1}{2}$

In[175]:=
```
LiczbyKwantowe =
 Table[{n, l, ml, s}, {n, 3}, {l, 0, n - 1}, {ml, -l, l}, {s, -1 / 2, 1 / 2}]
% // InputForm
```

Out[175]=
$$\left\{\left(\begin{pmatrix} 1 & 0 & 0 & -\frac{1}{2} \\ 1 & 0 & 0 & \frac{1}{2} \end{pmatrix}\right), \left\{\left(\{2,0,0,-\tfrac{1}{2}\}\ \{2,0,0,\tfrac{1}{2}\}\right), \begin{pmatrix} \{2,1,-1,-\frac{1}{2}\} & \{2,1,-1,\frac{1}{2}\} \\ \{2,1,0,-\frac{1}{2}\} & \{2,1,0,\frac{1}{2}\} \\ \{2,1,1,-\frac{1}{2}\} & \{2,1,1,\frac{1}{2}\} \end{pmatrix}\right\},$$
$$\left\{\left(\{3,0,0,-\tfrac{1}{2}\}\ \{3,0,0,\tfrac{1}{2}\}\right), \begin{pmatrix} \{3,1,-1,-\frac{1}{2}\} & \{3,1,-1,\frac{1}{2}\} \\ \{3,1,0,-\frac{1}{2}\} & \{3,1,0,\frac{1}{2}\} \\ \{3,1,1,-\frac{1}{2}\} & \{3,1,1,\frac{1}{2}\} \end{pmatrix}, \begin{pmatrix} \{3,2,-2,-\frac{1}{2}\} & \{3,2,-2,\frac{1}{2}\} \\ \{3,2,-1,-\frac{1}{2}\} & \{3,2,-1,\frac{1}{2}\} \\ \{3,2,0,-\frac{1}{2}\} & \{3,2,0,\frac{1}{2}\} \\ \{3,2,1,-\frac{1}{2}\} & \{3,2,1,\frac{1}{2}\} \\ \{3,2,2,-\frac{1}{2}\} & \{3,2,2,\frac{1}{2}\} \end{pmatrix}\right\}\right\}$$

Out[176]//InputForm=
```
{{{{{1, 0, 0, -1/2}, {1, 0, 0, 1/2}}}},
 {{{{2, 0, 0, -1/2}, {2, 0, 0, 1/2}}},
  {{{2, 1, -1, -1/2}, {2, 1, -1, 1/2}},
   {{2, 1, 0, -1/2}, {2, 1, 0, 1/2}}, {{2, 1, 1, -1/2},
    {2, 1, 1, 1/2}}}}, {{{{3, 0, 0, -1/2}, {3, 0, 0, 1/2}}},
  {{{3, 1, -1, -1/2}, {3, 1, -1, 1/2}},
   {{3, 1, 0, -1/2}, {3, 1, 0, 1/2}}, {{3, 1, 1, -1/2},
    {3, 1, 1, 1/2}}}, {{{3, 2, -2, -1/2}, {3, 2, -2, 1/2}},
   {{3, 2, -1, -1/2}, {3, 2, -1, 1/2}},
   {{3, 2, 0, -1/2}, {3, 2, 0, 1/2}}, {{3, 2, 1, -1/2},
    {3, 2, 1, 1/2}}, {{3, 2, 2, -1/2}, {3, 2, 2, 1/2}}}}}
```

In[177]:=
```
LiczbyKwantoweP = Flatten[LiczbyKwantowe, 3]
```

Out[177]=

$$\begin{pmatrix} 1 & 0 & 0 & -\frac{1}{2} \\ 1 & 0 & 0 & \frac{1}{2} \\ 2 & 0 & 0 & -\frac{1}{2} \\ 2 & 0 & 0 & \frac{1}{2} \\ 2 & 1 & -1 & -\frac{1}{2} \\ 2 & 1 & -1 & \frac{1}{2} \\ 2 & 1 & 0 & -\frac{1}{2} \\ 2 & 1 & 0 & \frac{1}{2} \\ 2 & 1 & 1 & -\frac{1}{2} \\ 2 & 1 & 1 & \frac{1}{2} \\ 3 & 0 & 0 & -\frac{1}{2} \\ 3 & 0 & 0 & \frac{1}{2} \\ 3 & 1 & -1 & -\frac{1}{2} \\ 3 & 1 & -1 & \frac{1}{2} \\ 3 & 1 & 0 & -\frac{1}{2} \\ 3 & 1 & 0 & \frac{1}{2} \\ 3 & 1 & 1 & -\frac{1}{2} \\ 3 & 1 & 1 & \frac{1}{2} \\ 3 & 2 & -2 & -\frac{1}{2} \\ 3 & 2 & -2 & \frac{1}{2} \\ 3 & 2 & -1 & -\frac{1}{2} \\ 3 & 2 & -1 & \frac{1}{2} \\ 3 & 2 & 0 & -\frac{1}{2} \\ 3 & 2 & 0 & \frac{1}{2} \\ 3 & 2 & 1 & -\frac{1}{2} \\ 3 & 2 & 1 & \frac{1}{2} \\ 3 & 2 & 2 & -\frac{1}{2} \\ 3 & 2 & 2 & \frac{1}{2} \end{pmatrix}$$

Wybieramy tylko wyniki z l =1

In[178]:=
```
MojTest[x_] := x[[2]] == 1
```

In[179]:=
```
? Select
```

Select[*list*, *crit*] picks out all elements e_i of *list* for which *crit*[e_i] is True.
Select[*list*, *crit*, *n*] picks out the first *n* elements for which *crit*[e_i] is True.
Select[*crit*] represents an operator form of Select that can be applied to an expression. ≫

In[180]:=
```
Select[LiczbyKwantoweP, MojTest]
```

Out[180]=

$$\begin{pmatrix} 2 & 1 & -1 & -\frac{1}{2} \\ 2 & 1 & -1 & \frac{1}{2} \\ 2 & 1 & 0 & -\frac{1}{2} \\ 2 & 1 & 0 & \frac{1}{2} \\ 2 & 1 & 1 & -\frac{1}{2} \\ 2 & 1 & 1 & \frac{1}{2} \\ 3 & 1 & -1 & -\frac{1}{2} \\ 3 & 1 & -1 & \frac{1}{2} \\ 3 & 1 & 0 & -\frac{1}{2} \\ 3 & 1 & 0 & \frac{1}{2} \\ 3 & 1 & 1 & -\frac{1}{2} \\ 3 & 1 & 1 & \frac{1}{2} \end{pmatrix}$$

In[181]:=
```
Select[LiczbyKwantoweP, #[[2]] == 1 &]
```

Out[181]=

$$\begin{pmatrix} 2 & 1 & -1 & -\frac{1}{2} \\ 2 & 1 & -1 & \frac{1}{2} \\ 2 & 1 & 0 & -\frac{1}{2} \\ 2 & 1 & 0 & \frac{1}{2} \\ 2 & 1 & 1 & -\frac{1}{2} \\ 2 & 1 & 1 & \frac{1}{2} \\ 3 & 1 & -1 & -\frac{1}{2} \\ 3 & 1 & -1 & \frac{1}{2} \\ 3 & 1 & 0 & -\frac{1}{2} \\ 3 & 1 & 0 & \frac{1}{2} \\ 3 & 1 & 1 & -\frac{1}{2} \\ 3 & 1 & 1 & \frac{1}{2} \end{pmatrix}$$

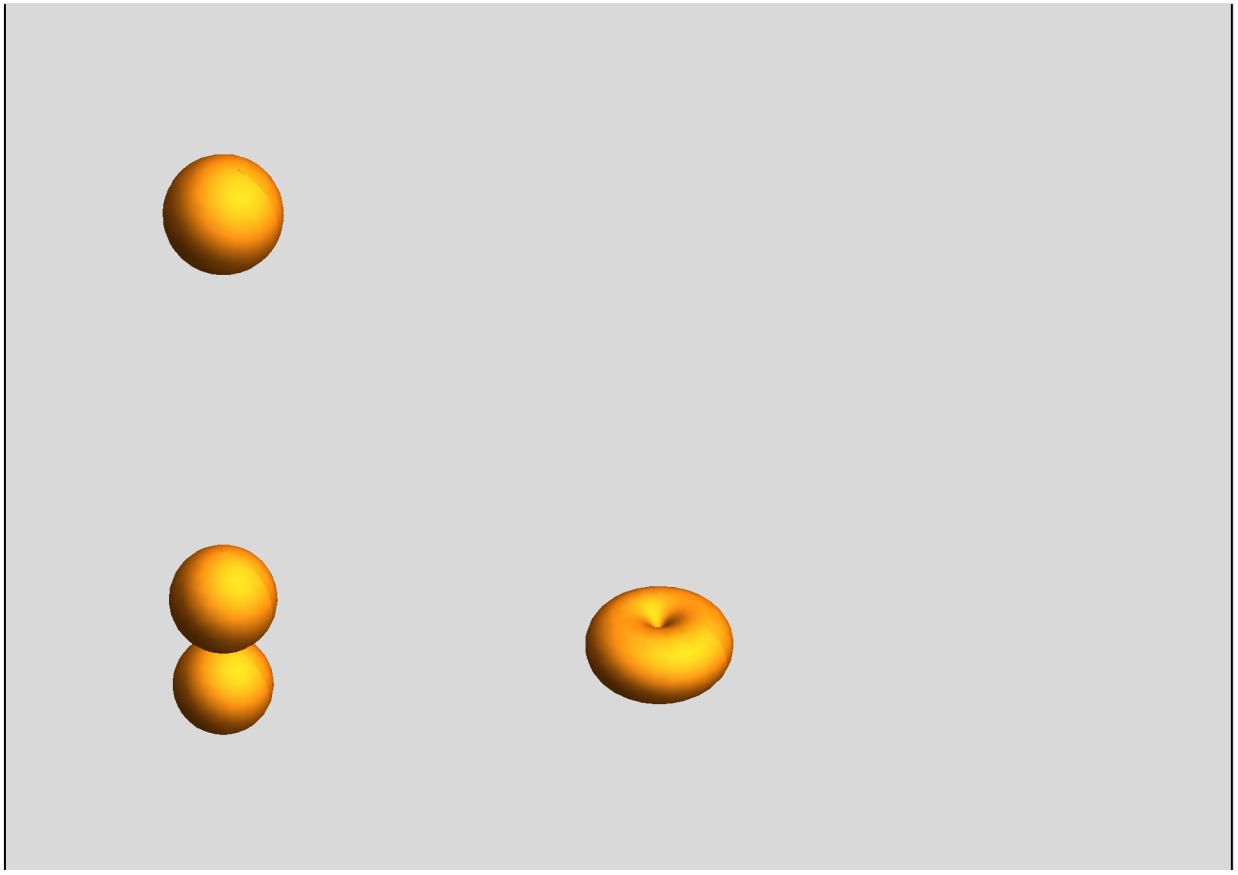
Narysuj orbity elektronu w atomie wodoru

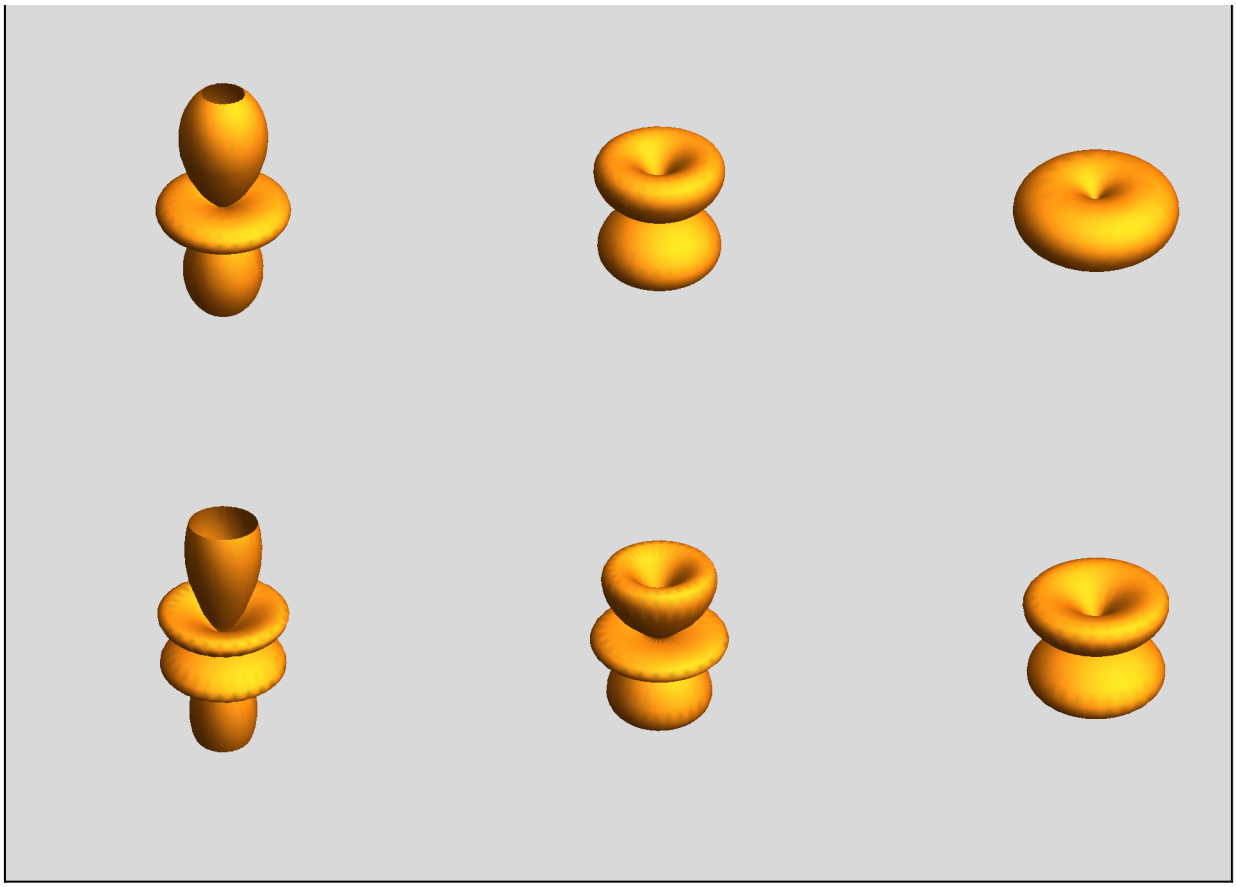
In[182]:=
```
? SphericalPlot3D
```

SphericalPlot3D[r, θ, ϕ] generates a 3D plot
with a spherical radius r as a function of spherical coordinates θ and ϕ.
SphericalPlot3D[r, {θ, θ_{min}, θ_{max}}, {ϕ, ϕ_{min}, ϕ_{max}}] generates a 3D spherical
plot over the specified ranges of spherical coordinates.
SphericalPlot3D[{r_1, r_2, ...}, {θ, θ_{min}, θ_{max}}, {ϕ, ϕ_{min}, ϕ_{max}}] generates a 3D spherical plot with multiple surfaces. ≫

In[183]:=
```
TableForm@Table[SphericalPlot3D[Evaluate@Abs@SphericalHarmonicY[l, m, θ, ϕ],
    {θ, 0, Pi}, {ϕ, 0, 2 Pi}, PlotRange → 0.6, Mesh → None,
    Boxed → False, Axes → None, ImageSize → 200], {l, 0, 3}, {m, 0, l}]
```

Out[183]//TableForm=





In[184]:=
```
Select[a3 x^3 + 3 x^2 + Log[x^2], MemberQ[#, 3, Infinity] &]
```

Out[184]=
$$a_3 x^3 + 3 x^2$$

In[185]:=
```
Select[a3 x^3 + 3 x^2 + Log[x^2], FreeQ[#, 3, Infinity] &]
```

Out[185]=
$$\log(x^2)$$

Wektory

W praktyce wektor to lista, ale mamy dość dużo dodatkowych funkcji

In[186]:=
```
Information /@ {PrimeQ, OddQ};
```

PrimeQ[*expr*] yields True if *expr* is a prime number, and yields False otherwise. ≫

```
Attributes[PrimeQ] = {Listable, Protected}
```

```
Options[PrimeQ] = {GaussianIntegers → False}
```

OddQ[*expr*] gives True if *expr* is an odd integer, and False otherwise. ≫

```
Attributes[OddQ] = {Listable, Protected}
```

In[187]:=
```
u = {x, y, z}
```
Out[187]= $\{x, y, z\}$

In[188]:=
```
u2 = {x1, x2, x3}
```
Out[188]= $\{x1, x2, x3\}$

In[189]:=
```
u.u2
```
Out[189]= $x\,x1 + x2\,y + x3\,z$

In[190]:=
```
Dot[u, u2]
```
Out[190]= $x\,x1 + x2\,y + x3\,z$

In[191]:=
```
MapThread[f, {u, u2}]
```
Out[191]= $\{f(x, x1), f(y, x2), f(z, x3)\}$

In[192]:=
```
Total@MapThread[Times, {u, u2}]
```
Out[192]= $x\,x1 + x2\,y + x3\,z$

In[193]:=
```
Dot[u, u2]
```
Out[193]= $x\,x1 + x2\,y + x3\,z$

In[194]:=
```
Cross[u, u2]
```
Out[194]= $\{x3\,y - x2\,z,\ x1\,z - x\,x3,\ x\,x2 - x1\,y\}$

In[195]:=
```
UnitVector[3, 3]
```
Out[195]= $\{0, 0, 1\}$

In[196]:=
```
? VectorQ
```

VectorQ[*expr*] gives True if *expr* is a list or a one-dimensional SparseArray object, none of whose elements are themselves lists, and gives False otherwise.
VectorQ[*expr*, *test*] gives True only if *test* yields True when applied to each of the elements in *expr*. ≫

In[197]:=
```
Norm[u2]
```

Out[197]=
$$\sqrt{\left|\text{x1}\right|^2+\left|\text{x2}\right|^2+\left|\text{x3}\right|^2}$$

In[198]:=
```
v1 = {1, 2, 3}; v2 = {4, 5, 6};
```

In[199]:=
```
VectorAngle[v1, v2]
% // N
```

Out[199]=
$$\cos^{-1}\left(\frac{16\sqrt{\frac{2}{11}}}{7}\right)$$

Out[200]=
$$0.225726$$

In[201]:=
```
Projection[v1, v2]
```

Out[201]=
$$\left\{\frac{128}{77},\frac{160}{77},\frac{192}{77}\right\}$$

In[202]:=
```
Normalize[v1]
```

Out[202]=
$$\left\{\frac{1}{\sqrt{14}},\sqrt{\frac{2}{7}},\frac{3}{\sqrt{14}}\right\}$$

In[203]:=
```
? Orthogonalize
```

Orthogonalize[{v_1, v_2, ...}] gives an orthonormal basis found by orthogonalizing the vectors v_i.
Orthogonalize[{e_1, e_2, ...}, f] gives a basis for the e_i orthonormal with respect to the inner product function f. ≫

Macierze

In[204]:=
```
Clear[m2]
m2 = {{a, b}, {c, d}}
```

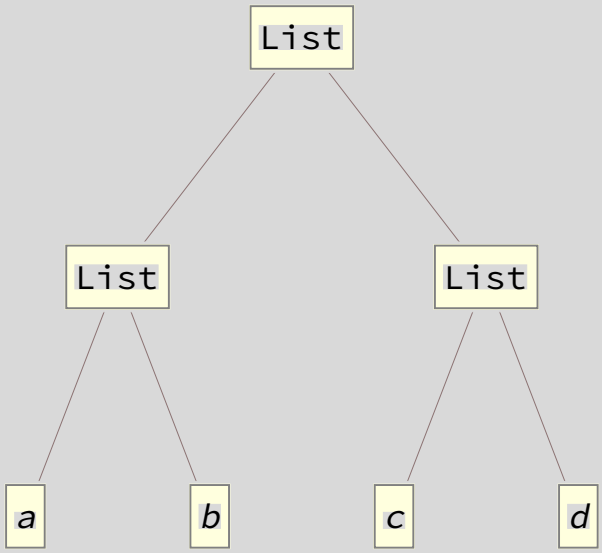
Out[205]=
$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

In[206]:=
```
m2 // StandardForm
```

Out[206]//StandardForm=
```
{{a, b}, {c, d}}
```

In[207]:=
```
m2 // TreeForm
```

Out[207]//TreeForm=



In[208]:=
```
m2[[2, 1]]
```

Out[208]=
$$c$$

In[209]:=
```
Det[m2]
```

Out[209]=
$$a\,d - b\,c$$

In[210]:=
```
Inverse[m2] // MatrixForm
```

Out[210]//MatrixForm=
$$\begin{pmatrix} \frac{d}{a\,d-b\,c} & -\frac{b}{a\,d-b\,c} \\ -\frac{c}{a\,d-b\,c} & \frac{a}{a\,d-b\,c} \end{pmatrix}$$

In[211]:=
```
m2.% // Simplify
% // MatrixForm
```

Out[211]=
$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Out[212]//MatrixForm=
$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

In[213]:=
```
Clear[M]
```

In[214]:=
```
M = {{A, B, C}, {1, 2, 3}}
```

Out[214]=
$$\begin{pmatrix} \{a, b, c\} & \{b, d, f\} & C \\ 1 & 2 & 3 \end{pmatrix}$$

In[215]:=
```
M[[2]]
```

Out[215]=
$$\{1, 2, 3\}$$

Ale

In[216]:=
```
M[[2, 3]]
```

Out[216]=
$$3$$

In[217]:=
```
Part[M, 1, 2]
```

Out[217]=
$$\{b, d, f\}$$

In[218]:=
```
Clear[M, a, b, c, d]
```

In[219]:=
```
M = {a, b, c, d}
```

Out[219]=
$$\{a, b, c, d\}$$

In[220]:=
```
M[[{1, 2}]]
```

Out[220]=
$$\{a, b\}$$

In[221]:=
```
Part[M, 1]
```

Out[221]=
$$a$$

In[222]:=
```
Part[M, -1]
```

Out[222]=
$$d$$

In[223]:=
```
DiagonalMatrix[{1, 2, 3}]
```

Out[223]=
$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$$

In[224]:=
```
DiagonalMatrix[{1, 2, 3, 4, 5, 6}, 3]
% // MatrixForm
```

Out[224]=
$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Out[225]//MatrixForm=
$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

In[226]:=
```
DiagonalMatrix[{1, 2, 3, 4, 5, 6}, -1]
```

Out[226]=
$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 6 & 0 \end{pmatrix}$$

In[227]:=
```
HankelMatrix[{1, 2, 3, 4}]
% // MatrixForm
```

Out[227]=
$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 0 \\ 3 & 4 & 0 & 0 \\ 4 & 0 & 0 & 0 \end{pmatrix}$$

Out[228]//MatrixForm=
$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 0 \\ 3 & 4 & 0 & 0 \\ 4 & 0 & 0 & 0 \end{pmatrix}$$

In[229]:=
```
Dimensions[%]
Dimensions[%%]
```

Out[229]= {4, 4}

Out[230]= {4, 4}

In[231]:=
```
? MatrixQ
```

MatrixQ[*expr*] gives True if *expr* is a list of lists or a two–dimensional
SparseArray object that can represent a matrix, and gives False otherwise.
MatrixQ[*expr*, *test*] gives True only if *test* yields True when applied to each of the matrix elements in *expr*. ≫

In[232]:=
```
? HermitianMatrixQ
```

HermitianMatrixQ[m] gives True if m is explicitly Hermitian, and False otherwise. ≫

In[233]:=
```
? PositiveDefiniteMatrixQ
```

PositiveDefiniteMatrixQ[m] gives True if m is explicitly positive definite, and False otherwise. ≫

Operacje na macierzach

In[234]:=
```
m2 = {{A, B}, {X, Δ}}
```

Out[234]=
$$\begin{pmatrix} \{a, b, c\} & \{b, d, f\} \\ X & \Delta \end{pmatrix}$$

In[235]:=
```
m3 = {{2, 2}, {1, 0}}
```

Out[235]=
$$\begin{pmatrix} 2 & 2 \\ 1 & 0 \end{pmatrix}$$

In[236]:=
```
m2 m3 // MatrixForm
```

Out[236]//MatrixForm=
$$\begin{pmatrix} \{2\,a,\,2\,b,\,2\,c\} & \{2\,b,\,2\,d,\,2\,f\} \\ \text{X} & 0 \end{pmatrix}$$

In[237]:=
```
m2.m3 // MatrixForm
```

... Dot: Nonrectangular tensor encountered.

Out[237]//MatrixForm=
$$\begin{pmatrix} \{a,\,b,\,c\} & \{b,\,d,\,f\} \\ \text{X} & \Delta \end{pmatrix}.\begin{pmatrix} 2 & 2 \\ 1 & 0 \end{pmatrix}$$

In[238]:=
```
Inverse[m3]
```

Out[238]=
$$\begin{pmatrix} 0 & 1 \\ \frac{1}{2} & -1 \end{pmatrix}$$

In[239]:=
```
m3 = {{2, 2}, {1, 0}}
```

Out[239]=
$$\begin{pmatrix} 2 & 2 \\ 1 & 0 \end{pmatrix}$$

In[240]:=
```
Inverse[m3] // MatrixForm
```

Out[240]//MatrixForm=
$$\begin{pmatrix} 0 & 1 \\ \frac{1}{2} & -1 \end{pmatrix}$$

In[241]:=
```
IdentityMatrix[25]
```

Out[241]=
$$\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

In[242]:=
```
v = {{0, 0, 0, 1}, {0, 0, 1, 0}}
% // MatrixForm
```

Out[242]=
$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Out[243]//MatrixForm=
$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

In[244]:=
```
Transpose[v]
% // MatrixForm
```

Out[244]=
$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Out[245]//MatrixForm=
$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}$$

In[246]:=
```
Transpose[{{0, 0, 0, 1}, {0, 0, 1, 0}}] // MatrixForm
```

Out[246]//MatrixForm=

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}$$

In[247]:=
```
Det[m4]
```

Out[247]= 24 883 200

In[248]:=
```
Inverse[m4]
```

Out[248]=

$$\begin{pmatrix} 6 & -\frac{87}{10} & \frac{29}{6} & -\frac{31}{24} & \frac{1}{6} & -\frac{1}{120} \\ -\frac{15}{2} & \frac{117}{8} & -\frac{461}{48} & \frac{137}{48} & -\frac{19}{48} & \frac{1}{48} \\ \frac{20}{3} & -\frac{127}{9} & \frac{31}{3} & -\frac{121}{36} & \frac{1}{2} & -\frac{1}{36} \\ -\frac{15}{4} & \frac{33}{4} & -\frac{307}{48} & \frac{107}{48} & -\frac{17}{48} & \frac{1}{48} \\ \frac{6}{5} & -\frac{27}{10} & \frac{13}{6} & -\frac{19}{24} & \frac{2}{15} & -\frac{1}{120} \\ -\frac{1}{6} & \frac{137}{360} & -\frac{5}{16} & \frac{17}{144} & -\frac{1}{48} & \frac{1}{720} \end{pmatrix}$$

In[249]:=
```
Eigenvalues[{{a, b}, {c, d}}]
```

Out[249]= $\left\{\frac{1}{2}\left(-\sqrt{a^2-2\,a\,d+4\,b\,c+d^2}+a+d\right),\ \frac{1}{2}\left(\sqrt{a^2-2\,a\,d+4\,b\,c+d^2}+a+d\right)\right\}$

In[250]:=
```
www = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}
% // MatrixForm
```

Out[250]=

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Out[251]//MatrixForm=

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

In[252]:=
```
% // MatrixForm
```

Out[252]//MatrixForm=

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

In[253]:=
```
Eigenvalues[www]
```

Out[253]=
$$\left\{\frac{3}{2}\left(5+\sqrt{33}\right), \frac{3}{2}\left(5-\sqrt{33}\right), 0\right\}$$

In[254]:=
```
Eigenvectors[www] // Simplify
```

Out[254]=
$$\begin{pmatrix} \frac{\perp}{22}\left(-11+3\sqrt{33}\right) & \frac{\perp}{44}\left(11+3\sqrt{33}\right) & 1 \\ \frac{\perp}{22}\left(-11-3\sqrt{33}\right) & \frac{\perp}{44}\left(11-3\sqrt{33}\right) & 1 \\ 1 & -2 & 1 \end{pmatrix}$$

In[255]:=
```
? Eigensystem
```

Eigensystem[m] gives a list {$values$, $vectors$} of the eigenvalues and eigenvectors of the square matrix m.
Eigensystem[{m, a}] gives the generalized eigenvalues and eigenvectors of m with respect to a.
Eigensystem[m, k] gives the eigenvalues and eigenvectors for the first k eigenvalues of m.
Eigensystem[{m, a}, k] gives the first k generalized eigenvalues and eigenvectors. ≫

In[256]:=
```
Eigensystem[www] // Simplify
```

Out[256]=
$$\begin{pmatrix} \frac{3}{2}\left(5+\sqrt{33}\right) & -\frac{3}{2}\left(-5+\sqrt{33}\right) & 0 \\ \left\{\frac{\perp}{22}\left(-11+3\sqrt{33}\right), \frac{\perp}{44}\left(11+3\sqrt{33}\right), 1\right\} & \left\{\frac{\perp}{22}\left(-11-3\sqrt{33}\right), \frac{\perp}{44}\left(11-3\sqrt{33}\right), 1\right\} & \{1, -2, 1\} \end{pmatrix}$$

In[257]:=
```
? Minors
```

Minors[m] gives the minors of a matrix m.
Minors[m, k] gives the k^{th} minors. ≫

In[258]:=
```
? ConjugateTranspose
```

ConjugateTranspose[m] or $m^{\dagger}$ gives the conjugate transpose of m. ≫

In[259]:=
```
? MatrixPower
```

MatrixPower[m, n] gives the n^{th} matrix power of the matrix m.
MatrixPower[m, n, v] gives the n^{th} matrix power of the matrix m applied to the vector v. ≫

In[260]:=
```
Tr[www]
```

Out[260]=
$$15$$

Wizualizacja Macierzy

In[261]:=
```
? PrimeQ
```

PrimeQ[*expr*] yields True if *expr* is a prime number, and yields False otherwise. ≫

In[262]:=
```
Definition[PrimeQ]
```

Out[262]=
```
Attributes[PrimeQ] = {Listable, Protected}

Options[PrimeQ] = {GaussianIntegers → False}
```

In[263]:=
```
?? PrimeQ
```

PrimeQ[*expr*] yields True if *expr* is a prime number, and yields False otherwise. ≫

```
Attributes[PrimeQ] = {Listable, Protected}

Options[PrimeQ] = {GaussianIntegers → False}
```

In[264]:=
```
Information[PrimeQ]
```

PrimeQ[*expr*] yields True if *expr* is a prime number, and yields False otherwise. ≫

```
Attributes[PrimeQ] = {Listable, Protected}

Options[PrimeQ] = {GaussianIntegers → False}
```

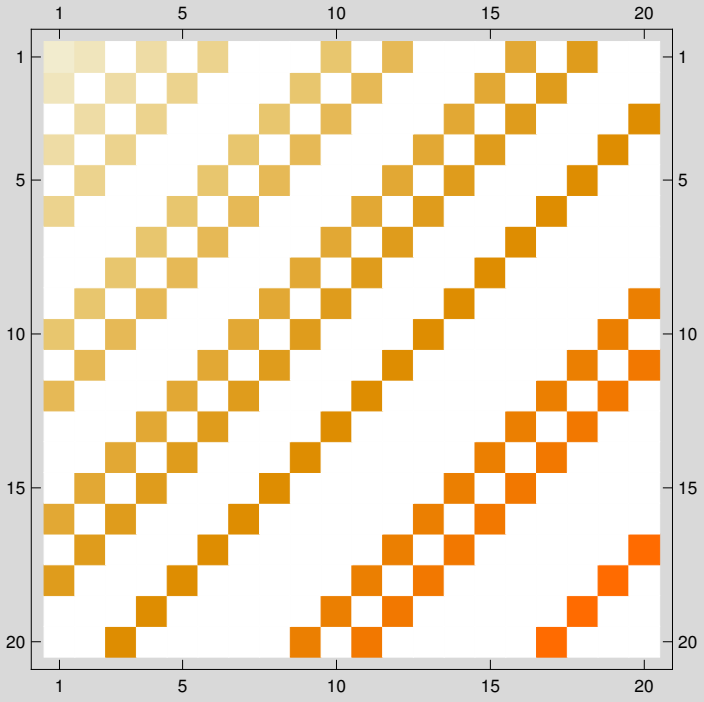
In[265]:=
```
Pierwsze = Table[If[PrimeQ[i + j], i + j, 0], {i, 1, 20}, {j, 1, 20}]
```

Out[265]=

2	3	0	5	0	7	0	0	0	11	0	13	0	0	0	17	0	19	0	0
3	0	5	0	7	0	0	0	11	0	13	0	0	0	17	0	19	0	0	0
0	5	0	7	0	0	0	11	0	13	0	0	0	17	0	19	0	0	0	23
5	0	7	0	0	0	11	0	13	0	0	0	17	0	19	0	0	0	23	0
0	7	0	0	0	11	0	13	0	0	0	17	0	19	0	0	0	23	0	0
7	0	0	0	11	0	13	0	0	0	17	0	19	0	0	0	23	0	0	0
0	0	0	11	0	13	0	0	0	17	0	19	0	0	0	23	0	0	0	0
0	0	11	0	13	0	0	0	17	0	19	0	0	0	23	0	0	0	0	0
0	11	0	13	0	0	0	17	0	19	0	0	0	23	0	0	0	0	0	29
11	0	13	0	0	0	17	0	19	0	0	0	23	0	0	0	0	0	29	0
0	13	0	0	0	17	0	19	0	0	0	23	0	0	0	0	0	29	0	31
13	0	0	0	17	0	19	0	0	0	23	0	0	0	0	0	29	0	31	0
0	0	0	17	0	19	0	0	0	23	0	0	0	0	0	29	0	31	0	0
0	0	17	0	19	0	0	0	23	0	0	0	0	0	29	0	31	0	0	0
0	17	0	19	0	0	0	23	0	0	0	0	0	29	0	31	0	0	0	0
17	0	19	0	0	0	23	0	0	0	0	0	29	0	31	0	0	0	0	0
0	19	0	0	0	23	0	0	0	0	0	29	0	31	0	0	0	0	0	37
19	0	0	0	23	0	0	0	0	0	29	0	31	0	0	0	0	0	37	0
0	0	0	23	0	0	0	0	0	29	0	31	0	0	0	0	0	37	0	0
0	0	23	0	0	0	0	0	29	0	31	0	0	0	0	0	37	0	0	0

In[266]:=
```
MatrixPlot[Pierwsze]
```

Out[266]=



In[267]:=
```
? MatrixPlot
```

MatrixPlot[m] generates a plot that gives a visual representation of the values of elements in a matrix. >>

In[268]:=
```
? RandomInteger
```

RandomInteger[{i_{min}, i_{max}}] gives a pseudorandom integer in the range {i_{min}, i_{max}}.
RandomInteger[i_{max}] gives a pseudorandom integer in the range {0, …, i_{max}}.
RandomInteger[] pseudorandomly gives 0 or 1.
RandomInteger[*range*, n] gives a list of n pseudorandom integers.
RandomInteger[*range*, {n_1, n_2, …}] gives an $n_1 \times n_2 \times \ldots$ array of pseudorandom integers. ≫

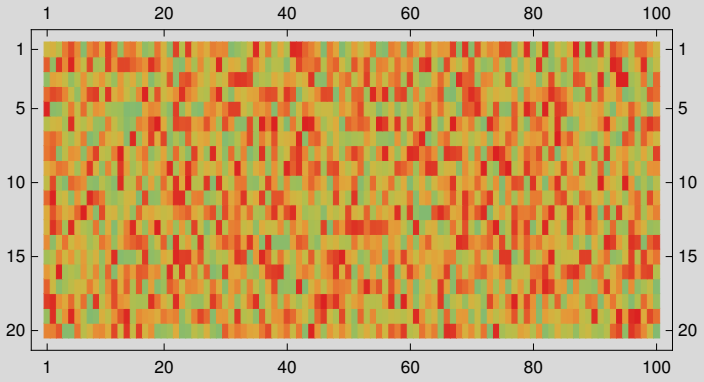
In[269]:=
```
a = RandomInteger[{0, 100}, {20, 100}]
```

Out[269]=

39	45	29	72	87	66	2	68	78	52	91	71	10	70	74	49	7	16	75	44	4	90	26	92	42	6
82	7	93	66	99	62	32	88	19	22	91	63	89	89	80	73	64	95	81	48	60	6	2	58	19	6
8	55	50	35	9	25	63	26	46	61	31	12	60	30	49	16	50	2	21	34	84	5	83	87	69	4
78	88	84	68	35	8	60	88	92	30	94	47	62	36	20	67	99	29	85	37	50	65	95	70	59	2
99	4	7	27	72	33	77	28	63	57	19	16	14	0	2	0	53	68	65	19	4	91	9	43	5	1
26	39	42	18	96	36	25	45	11	9	70	33	16	3	10	70	63	88	100	50	10	89	85	86	55	6
72	79	69	31	17	42	71	0	9	71	46	37	48	2	98	12	51	91	50	89	4	22	8	79	35	5
74	81	4	30	64	60	6	67	90	32	26	71	100	44	20	59	74	7	22	54	9	77	28	5	90	4
48	59	75	71	50	17	59	93	45	8	68	23	93	57	17	33	66	30	48	65	97	18	59	95	72	2
31	88	16	2	79	72	65	8	4	27	14	3	97	29	29	78	11	55	12	77	36	95	51	100	26	2
79	100	30	39	5	62	23	97	84	72	18	11	22	40	43	0	88	7	9	57	68	96	96	18	70	7
50	85	67	45	55	79	100	61	4	87	39	43	17	43	9	79	46	53	41	34	38	69	74	81	52	4
44	95	79	17	97	43	24	17	74	6	49	1	35	70	97	45	11	60	100	33	60	13	59	14	90	3
66	5	41	75	38	53	44	17	29	74	63	11	49	73	80	86	79	10	36	81	6	96	29	56	18	9
17	25	69	40	62	28	58	26	45	9	55	96	2	6	6	30	92	6	11	52	40	98	91	92	58	8
60	39	99	17	24	18	60	30	67	12	79	2	50	76	58	83	81	59	66	16	96	33	93	37	71	2
81	37	5	17	30	40	38	72	49	16	67	87	50	81	21	83	6	57	2	13	82	72	37	68	66	5
92	97	64	52	78	41	76	4	93	29	58	87	64	58	22	19	20	95	75	47	29	96	57	20	11	1
11	96	22	63	98	62	28	0	4	46	31	64	84	98	21	66	60	57	49	73	65	70	7	30	70	8
69	70	83	2	3	74	28	43	65	67	50	89	38	94	34	91	43	64	6	11	16	50	4	6	29	1

In[270]:=
```
MatrixPlot[a, ColorFunction → "Rainbow", AspectRatio → 1 / 2]
```

Out[270]=

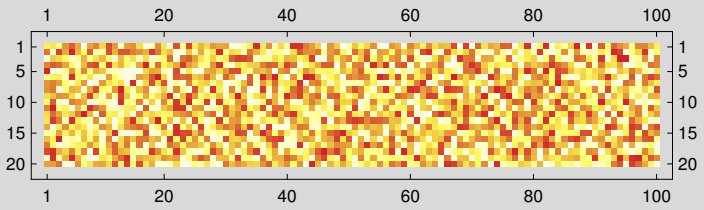


In[271]:=
```
Options[MatrixPlot]
```

Out[271]=
```
{AlignmentPoint → Center, AspectRatio → Automatic, Axes → False, AxesLabel → None,
  AxesOrigin → Automatic, AxesStyle → {}, Background → None, BaselinePosition → Automatic,
  BaseStyle → {}, ClippingStyle → Automatic, ColorFunction → Automatic, ColorFunctionScaling → True,
  ColorOutput → Automatic, ColorRules → Automatic, ContentSelectable → Automatic,
  CoordinatesToolOptions → Automatic, DataRange → All, DataReversed → False,
  DisplayFunction :→ $DisplayFunction, Epilog → {}, FormatType :→ TraditionalForm,
  Frame → True, FrameLabel → None, FrameStyle → {}, FrameTicks → All, FrameTicksStyle → {},
  GridLines → None, GridLinesStyle → {}, ImageMargins → 0., ImagePadding → All,
  ImageSize → Automatic, ImageSizeRaw → Automatic, LabelStyle → {}, MaxPlotPoints → Automatic,
  Mesh → False, MeshStyle → GrayLevel[ϕ − 1], Method → Automatic, MissingStyle → Automatic,
  PixelConstrained → False, PlotLabel → None, PlotLegends → None, PlotRange → All,
  PlotRangeClipping → False, PlotRangePadding → Automatic, PlotRegion → Automatic,
  PlotTheme :→ $PlotTheme, PreserveImageOptions → Automatic, Prolog → {},
  RotateLabel → True, TargetUnits → Automatic, Ticks → Automatic, TicksStyle → {}}
```

In[272]:=
```
MatrixPlot[a, ColorFunction → "TemperatureMap"]
```

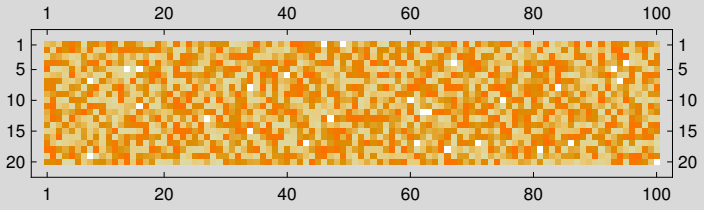
Out[272]=



In[273]:=
```
MatrixPlot[a, ColorFunction → "Hue"]
```

MatrixPlot: Value of option ColorFunction –> Hue is not a valid color function, or a gradient ColorData entity.

Out[273]=



Macierze Pauliego

In[274]:=
```
σx = {{0, 1}, {1, 0}}
```

Out[274]=

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

In[275]:=
```
Eigensystem[σx]
Print["Wartości własne:"]
%%[[1]]
Print["Wectory własne"]
%%%%[[2, 1]] // MatrixForm
%%%%%[[2, 2]] // MatrixForm
```

Out[275]=
$$\begin{pmatrix} -1 & 1 \\ \{-1,\ 1\} & \{1,\ 1\} \end{pmatrix}$$

Wartości własne:

Out[277]=
$$\{-1,\ 1\}$$

Wectory własne

Out[279]//MatrixForm=
$$\begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

Out[280]//MatrixForm=
$$\begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

In[281]:=
```
? *Pauli*
```

PauliMatrix[k] gives the k^{th} Pauli spin matrix σ_k. ≫

In[282]:=
```
PauliMatrix[1] // MatrixForm
```

Out[282]//MatrixForm=
$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

In[283]:=
```
PauliMatrix[2] // MatrixForm
```

Out[283]//MatrixForm=
$$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

In[284]:=
```
PauliMatrix[3] // MatrixForm
```

Out[284]//MatrixForm=
$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Wyrażenie jako lista -- lista to wyrażenie

In[285]:=
```
lista44 = (1 + x)^10 // Expand
```
Out[285]= $x^{10} + 10\,x^9 + 45\,x^8 + 120\,x^7 + 210\,x^6 + 252\,x^5 + 210\,x^4 + 120\,x^3 + 45\,x^2 + 10\,x + 1$

In[286]:=
```
lista44
% // Reverse
```
Out[286]= $x^{10} + 10\,x^9 + 45\,x^8 + 120\,x^7 + 210\,x^6 + 252\,x^5 + 210\,x^4 + 120\,x^3 + 45\,x^2 + 10\,x + 1$

Out[287]= $x^{10} + 10\,x^9 + 45\,x^8 + 120\,x^7 + 210\,x^6 + 252\,x^5 + 210\,x^4 + 120\,x^3 + 45\,x^2 + 10\,x + 1$

In[288]:=
```
lista44[[6]]
```
Out[288]= $252\,x^5$

In[289]:=
```
lista44bis = lista44 /. 252 x^5 → b
```
Out[289]= $b + x^{10} + 10\,x^9 + 45\,x^8 + 120\,x^7 + 210\,x^6 + 210\,x^4 + 120\,x^3 + 45\,x^2 + 10\,x + 1$

In[290]:=
```
Length[lista44]
```
Out[290]= 11

In[291]:=
```
Dimensions[lista44]
```
Out[291]= {11}

In[292]:=
```
Table[lista44[[k]], {k, 1, 11}]
```
Out[292]= $\{1,\ 10\,x,\ 45\,x^2,\ 120\,x^3,\ 210\,x^4,\ 252\,x^5,\ 210\,x^6,\ 120\,x^7,\ 45\,x^8,\ 10\,x^9,\ x^{10}\}$

In[293]:=
```
lista44[[1]]
lista44 = (lista44 /. lista44[[2]] → Graczyk)
```
Out[293]= 1

Out[294]= $\text{Graczyk} + x^{10} + 10\,x^9 + 45\,x^8 + 120\,x^7 + 210\,x^6 + 252\,x^5 + 210\,x^4 + 120\,x^3 + 45\,x^2 + 1$

In[295]:=
```
lista44[[1]]
```
Out[295]= 1

In[296]:=
```
lista44[[2]]
```

Out[296]=
Graczyk

In[297]:=
```
lista44[[2]]
```

Out[297]=
Graczyk

In[298]:=
```
lista55 = Log[x + y^2] + Sin[x^2 + y]
```

Out[298]=
$\sin(x^2+y)+\log(x+y^2)$

In[299]:=
```
lista55[[2]][[1]][[2]]
```

Out[299]=
y

In[300]:=
```
Array[LegendreP[#, x] &, 10] // Expand
Select[%, (3 x^2 / 2 - 1 / 2 == # &)]
```

Out[300]=
$$\left\{x,\ \frac{3x^2}{2}-\frac{1}{2},\ \frac{5x^3}{2}-\frac{3x}{2},\ \frac{35x^4}{8}-\frac{15x^2}{4}+\frac{3}{8},\ \frac{63x^5}{8}-\frac{35x^3}{4}+\frac{15x}{8},\right.$$
$$\frac{231x^6}{16}-\frac{315x^4}{16}+\frac{105x^2}{16}-\frac{5}{16},\ \frac{429x^7}{16}-\frac{693x^5}{16}+\frac{315x^3}{16}-\frac{35x}{16},$$
$$\frac{6435x^8}{128}-\frac{3003x^6}{32}+\frac{3465x^4}{64}-\frac{315x^2}{32}+\frac{35}{128},\ \frac{12\,155x^9}{128}-\frac{6435x^7}{32}+\frac{9009x^5}{64}-\frac{1155x^3}{32}+\frac{315x}{128},$$
$$\left.\frac{46\,189x^{10}}{256}-\frac{109\,395x^8}{256}+\frac{45\,045x^6}{128}-\frac{15\,015x^4}{128}+\frac{3465x^2}{256}-\frac{63}{256}\right\}$$

Out[301]=
$$\left\{\frac{3x^2}{2}-\frac{1}{2}\right\}$$