Przytaczając definicję liczb zespolonych, wyznacznika, wypowiedź twierdzenia Cramera oraz Kroneckera-Capelliego, korzystałem z I tomu wykładów Prof. Andrzeja Staruszkiewicza dla fizyków: ALGEBRA I GEOMETRIA, Kraków, 1993.

Liczby zespolone

Liczby zespolone to (uporządkowane) pary liczb rzeczywistych, dla których dodawanie i mnożenie jest określone wzorami:
(a,b) + (c,d) = (a+c,b+d)
(a,b) * (c,d) = (a*c-b*d,a*d+b*c),
gdzie a, b, c i d to liczby rzeczywiste. Liczby zespolone z powyższymi działaniami stanowią ciało.

Liczbę (a,b) można napisać w postaci:
(a,b) = (a,0)*(1,0) + (b,0)*(0,1).
Liczby postaci (a,0) wykazują identyczne własności jak liczby rzeczywiste, a więc możemy je utożsamić z nimi i pisać krótko a. Liczbę zespoloną (0,1) oznaczamy literą *i.*
Ostatecznie, zamiast (a,b) piszemy często a + b*i, a taką postać liczby zespolonej nazywamy algebraiczną lub równoważnie kanoniczną. Liczbę a nazywamy częścią rzeczywistą liczby z (a = Re(z)), a liczbę b nazywamy częścią urojoną liczby z (b=Im(z)).
Taki zapis umożliwia w szczególności szybkie znalezienie odwrotności liczby z, z^{-1}, która powinna spełniać z definicji
z* z^{-1}= z^{-1}* z=1. Mianowicie z^{-1}= 1/(a+b*i)= (a-b*i)/((a-b*i)*(a+b*i)) = $\frac{a}{a*a+b*b}$ - $\frac{b}{a*a+b*b}$ *i, przy czym musi zachodzić z ≠0 (tzn. przynajmniej jedna z liczb a,b musi być różna od zera).
Wielkość $|\ z\ | = \sqrt{a^2+b^2}$ nazywamy wartością bezwzględną (modułem) liczby zespolonej.
Każdą liczbę ϕ o tej własności, że
Re(z) = | z | cos ϕ ,
Im(z) = | z | sin ϕ ,

nazywamy argumentem liczby zespolonej z, pisząc ϕ=arg(z).
Jeśli ϕ=arg(z) → $\phi'=\phi+2\pi$=arg(z).
Dlatego umawiamy się co do tzw. argumentu głównego liczby zespolonej, że będzie leżał w przedziale $[0, 2\pi)$.

Każda liczba zespolona z może być więc zapisana w postaci trygonometrycznej:
$z = |z| (\cos\phi + i^*\sin\phi)$, gdzie ϕ jest dowolnym argumentem z.

Często używany jest też zapis wykładniczy
$z = |z| e^{i\phi}$, równoważny postaci trygonometrycznej.
Każda liczba zespolona różna od zera ma n różnych pierwiastków n-tego stopnia. Są to liczby o tej własności, że podniesione do potęgi n dają liczbę z. Jeśli $z = |z| e^{i\phi}$, to pierwiastkami są liczby
$z_k = \sqrt[n]{|z|}\, e^{\frac{i(\phi+2k\pi)}{n}}$, gdzie k=0, 1, ..., n-1.
W innym zapisie:
$z_k = \sqrt[n]{|z|} \left(\cos\left(\frac{\phi+2k\pi}{n}\right) + i*\sin\left(\frac{\phi+2k\pi}{n}\right)\right)$, k=0, 1, ..., n-1.

Proste przykłady

In[1]:=
```
Solve[z^2 == -1, z]
```
rozwiąż równanie

Out[1]= $\{\{z\to -i\},\{z\to i\}\}$

In[2]:=
```
Solve[z^4 == 1, z]
```
rozwiąż równanie

Out[2]= $\{\{z\to -1\},\{z\to -i\},\{z\to i\},\{z\to 1\}\}$

In[3]:=
```
Solve[z^4 == -1, z]
```
rozwiąż równanie

Out[3]= $\{\{z\to -(-1)^{1/4}\},\{z\to (-1)^{1/4}\},\{z\to -(-1)^{3/4}\},\{z\to (-1)^{3/4}\}\}$

In[4]:=
```
Solve[z^4 == Sqrt[2] / 2 + I * Sqrt[2] / 2, z]
```
rozwiąż równa··· pierwiastek k··· ··· pierwiastek kwadrat

Out[4]= $\left\{\left\{z\to -\frac{(1+i)^{1/4}}{2^{1/8}}\right\},\left\{z\to -\frac{i\,(1+i)^{1/4}}{2^{1/8}}\right\},\left\{z\to \frac{i\,(1+i)^{1/4}}{2^{1/8}}\right\},\left\{z\to \frac{(1+i)^{1/4}}{2^{1/8}}\right\}\right\}$

In[5]:=
```
Simplify[Solve[z^4 == Sqrt[2] / 2 + I * Sqrt[2] / 2, z]]
```
uprość rozwiąż równa··· pierwiastek k··· ··· pierwiastek kwadratow

Out[5]= $\left\{\left\{z\to -\frac{(1+i)^{1/4}}{2^{1/8}}\right\},\left\{z\to -\frac{i\,(1+i)^{1/4}}{2^{1/8}}\right\},\left\{z\to \frac{i\,(1+i)^{1/4}}{2^{1/8}}\right\},\left\{z\to \frac{(1+i)^{1/4}}{2^{1/8}}\right\}\right\}$

In[6]:=
```
FullSimplify[Solve[z^4 == Sqrt[2] / 2 + I * Sqrt[2] / 2, z]]
```
uprość pełniej | rozwiąż równa··· | pierwiastek k··· | ··· | pierwiastek kwadratow

Out[6]= $\{\{z \to -(-1)^{1/16}\}, \{z \to -(-1)^{9/16}\}, \{z \to (-1)^{9/16}\}, \{z \to (-1)^{1/16}\}\}$

Powyższe wyniki nie są dane ani w postaci algebraicznej, ani w postaci trygonometrycznej, ani w postaci wykładniczej. Dlatego spróbuję bezpośrednio wykorzystać definicję.

Najpierw sprawdzam jednak konwencje używane w programie Mathematica !

In[7]:=
```
p1 = ListPlot[{{1, 1}}, AspectRatio → 1, AxesLabel → {"Re(z)", "Im(z)"},
    PlotStyle → {Red, PointSize[Large]}, PlotRange → {{-2, 2}, {-2, 2}}];
```
wykres danych z listy | format obrazu | oznaczenia osi | część rze··· | część urojona
styl grafiki | cz··· | rozmiar kro··· | duży | zakres wykresu

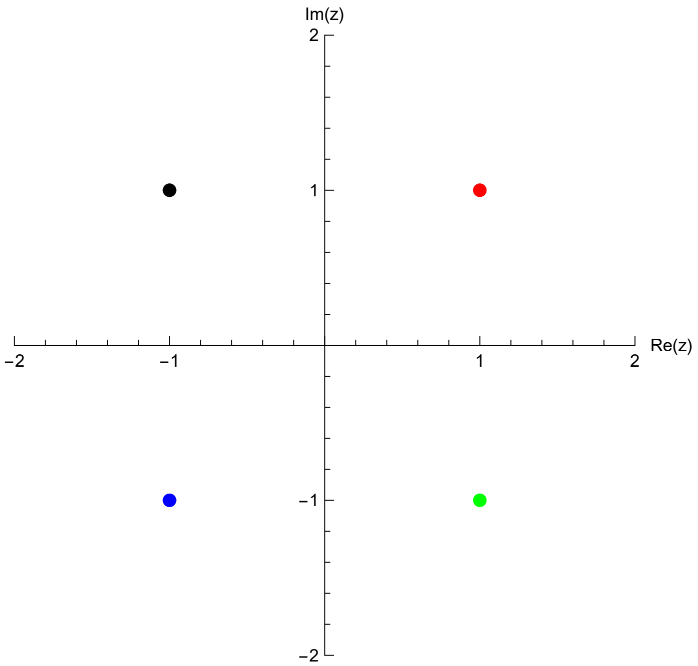
In[8]:=
```
p2 = ListPlot[{{-1, 1}}, AspectRatio → 1, AxesLabel → {"Re(z)", "Im(z)"},
    PlotStyle → {Black, PointSize[Large]}, PlotRange → {{-2, 2}, {-2, 2}}];
```
wykres danych z listy | format obrazu | oznaczenia osi | część rze··· | część urojona
styl grafiki | czarny | rozmiar kro··· | duży | zakres wykresu

In[9]:=
```
p3 = ListPlot[{{-1, -1}}, AspectRatio → 1, AxesLabel → {"Re(z)", "Im(z)"},
    PlotStyle → {Blue, PointSize[Large]}, PlotRange → {{-2, 2}, {-2, 2}}];
```
wykres danych z listy | format obrazu | oznaczenia osi | część rze··· | część urojon
styl grafiki | niebi··· | rozmiar kro··· | duży | zakres wykresu

In[10]:=
```
p4 = ListPlot[{{1, -1}}, AspectRatio → 1, AxesLabel → {"Re(z)", "Im(z)"},
    PlotStyle → {Green, PointSize[Large]}, PlotRange → {{-2, 2}, {-2, 2}}];
```
wykres danych z listy | format obrazu | oznaczenia osi | część rze··· | część urojona
styl grafiki | zielony | rozmiar kro··· | duży | zakres wykresu

In[11]:=
```
Show[p1, p2, p3, p4]
```
pokaż

Out[11]=



In[12]:=
```
Arg[1 + I]
```
argum··· jedność uroj

Out[12]= $\frac{\pi}{4}$

In[13]:=
```
Arg[-1 + I]
```
argument·· jedność ur

Out[13]= $\frac{3\pi}{4}$

In[14]:=
```
Arg[-1 - I]
```
argument·· jedność ur

Out[14]= $-\frac{3\pi}{4}$

In[15]:=
```
Arg[1 - I]
```
argum··· jedność uroj

Out[15]= $-\frac{\pi}{4}$

In[16]:=
```
Arg[-1]
```
argument liczby zespolonej

Out[16]= π

Nowa definicja argumentu liczby zespolonej

Widać, że dostaję wartości z przedziału $(-\pi, \pi]$. Chciałbym używać wartości argumentu głównego z przedziału $[0, 2\pi)$. Dlatego tworzę własną definicję argumentu:

In[17]:=
```
MyArg[z_] := If[Arg[z] ≥ 0, Arg[z], Arg[z] + 2 * Pi]
```
··· argument lic··· argumen·· argument lic··· pi

In[18]:=
```
MyArg[1 + I]
```
jedność uroj

Out[18]= $\frac{\pi}{4}$

In[19]:=
```
MyArg[-1 + I]
```
jedność ur

Out[19]= $\frac{3\pi}{4}$

In[20]:=
```
MyArg[-1 - I]
```
jedność ur

Out[20]= $\frac{5\pi}{4}$

In[21]:=
```
MyArg[1 - I]
```
jedność uroj

Out[21]=
$$\frac{7\pi}{4}$$

In[22]:=
```
Simplify[Abs[1 - I] * (Cos[MyArg[1 - I]] + I * Sin[MyArg[1 - I]])]
```
uprość wartoś··· jed··· cosinus jed··· ⌊··· sinus jedność

Out[22]=
$1 - i$

In[23]:=
```
MyArg[-1]
```

Out[23]=
π

Dlatego wprowadzam własne definicje pierwiastka n-tego stopnia z liczby zespolonej.
W postaci wykładniczej:

In[24]:=
```
MyRoot[n_, z_] :=
 Table[(Abs[z])^(1/n) * Exp[I * (MyArg[z] + 2 * k * Pi) / n], {k, 0, n - 1, 1}]
```
tabela wartość bezwzględna fu··· jedność urojona pi

oraz w równoważnej postaci trygonometrycznej:

In[25]:=
```
MyRootv2[n_, z_] := Table[(Abs[z])^(1/n) *
   (Cos[(MyArg[z] + 2 * k * Pi) / n] + I * Sin[(MyArg[z] + 2 * k * Pi) / n]), {k, 0, n - 1, 1}]
```
tabela wartość bezwzględna
cosinus pi ⌊··· sinus pi

In[26]:=
```
(* Przykład pierwiastka kwadratowego *)
```

In[27]:=
```
p22 = MyRoot[2, Sqrt[2] / 2 + I * Sqrt[2] / 2]
```
pierwiastek k··· ⌊··· pierwiastek kwa

Out[27]=
$$\left\{e^{\frac{i\pi}{8}}, e^{-\frac{7 i\pi}{8}}\right\}$$

In[28]:=
```
p22v2 = MyRootv2[2, Sqrt[2] / 2 + I * Sqrt[2] / 2]
```
pierwiastek k··· ⌊··· pierwiastek kwa

Out[28]=
$$\left\{\mathrm{Cos}\left[\frac{\pi}{8}\right] + i\,\mathrm{Sin}\left[\frac{\pi}{8}\right], -\mathrm{Cos}\left[\frac{\pi}{8}\right] - i\,\mathrm{Sin}\left[\frac{\pi}{8}\right]\right\}$$

In[29]:=
```
p22^2
```

Out[29]=
$$\left\{e^{\frac{i\pi}{4}}, e^{\frac{i\pi}{4}}\right\}$$

In[30]:=
$$\mathrm{DeleteDuplicates}\left[\left\{e^{\frac{i\pi}{4}}, e^{\frac{i\pi}{4}}\right\}\right]$$
usuń kopie

Out[30]=
$$\left\{e^{\frac{i\pi}{4}}\right\}$$

In[31]:=
```
(* Przykład pierwiastka szóstego stopnia *)
```

```
In[32]:= p61 = MyRoot[6, 1]
```

Out[32]=

$$\left\{1,\ e^{\frac{i\pi}{3}},\ e^{\frac{2i\pi}{3}},\ -1,\ e^{-\frac{2i\pi}{3}},\ e^{-\frac{i\pi}{3}}\right\}$$

```
In[33]:= p61^6
Out[33]= {1, 1, 1, 1, 1, 1}

In[34]:= DeleteDuplicates[p61^6]
         usuń kopie
Out[34]= {1}

In[35]:= DeleteDuplicates[p61^6][[1]]
         usuń kopie
Out[35]= 1
```

Teraz spróbujemy bardziej ogólnego zapisu, gdzie występują zmienne

```
In[36]:= ClearAll[x, y, z];
         wyczyść wszystko

In[37]:= z = x + I * y
                jedno
Out[37]= x + i y

In[38]:= Re[z]
         część rzeczywista
Out[38]= -Im[y] + Re[x]

In[39]:= (* Mathematica nie wie, że x i y mają być liczbami
          rzeczywistymi i daje ogólną odpowiedź. Możemy poinformować program,
         że x i y są rzeczywiste, używając ComplexExpand *)
                                       rozwiń na część rzeczywistą i urojoną

In[40]:= (* Ten zapis oznacza,
         że x i y (wszystkie wielkości w wyrażeniu) są liczbami rzeczywistymi *)

In[41]:= ComplexExpand[Re[x + I * y]]
         rozwiń na część ⋯ część⋯ jedność u
Out[41]= x

In[42]:= (* Ten zapis oznacza, że x i y są liczbami rzeczywistymi,
         ale w jest dowolną liczbą zespoloną ! *)

In[43]:= ComplexExpand[Re[w^2 * (x + I * y)], w]
         rozwiń na część ⋯ część rzeczywi⋯ jedność urojon
```

Out[43]=

$$-x\,\mathrm{Im}[w]^2 - 2\,y\,\mathrm{Im}[w]\,\mathrm{Re}[w] + x\,\mathrm{Re}[w]^2$$

Definiowanie obszarów i linii w płaszczyźnie XY za pomocą liczby zespolonych

Niech z1= x1 + I*y1, z2=x2 + I*y2 oznaczają dwie liczby na płaszczyźnie zespolonej.

Utożsamiamy | z1 - z2 | z odległością punktów P1(x1,y1) i P2(x2,y2) !

In[44]:=
```
(* W ten sposób określimy koło o środku w punkcie (0,0) i promieniu Sqrt[2] *)
```
pierwiastek kwa

In[45]:=
```
plot1 = ComplexExpand[Abs[z] ≤ Sqrt[2] ^ 2]
```
rozwiń na część ⋯ wartość ⋯ pierwiastek kwa

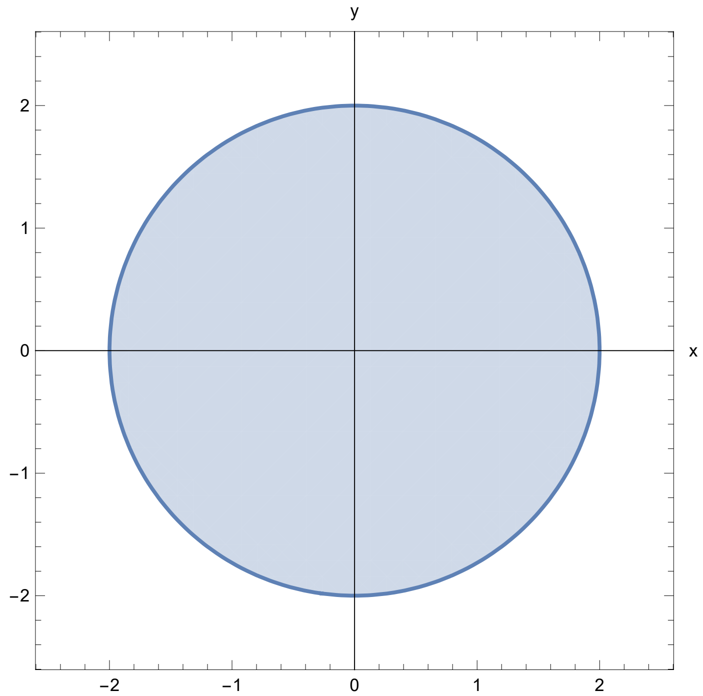
Out[45]=

$$\sqrt{x^2 + y^2} \le 2$$

In[46]:=
```
RegionPlot[plot1, {x, -5 / 2, 5 / 2}, {y, -5 / 2, 5 / 2},
 Axes → True, AxesLabel → {"x", "y"}, AspectRatio → 1]
```
wykres regionu na płaszczyźnie

osie pra⋯ oznaczenia osi format obrazu

Out[46]=



In[47]:=
```
(* W ten sposób określimy równanie okręgu
 o środku w punkcie (1,1) i promieniu Sqrt[2] *)
```
pierwiastek kwa

In[48]:=
```
plot2 = ComplexExpand[Abs[z - 1 - I] == 2]
```
rozwiń na część ⋯ wartość b⋯ jedność ui

Out[48]=

$$\sqrt{(-1 + x)^2 + (-1 + y)^2} == 2$$

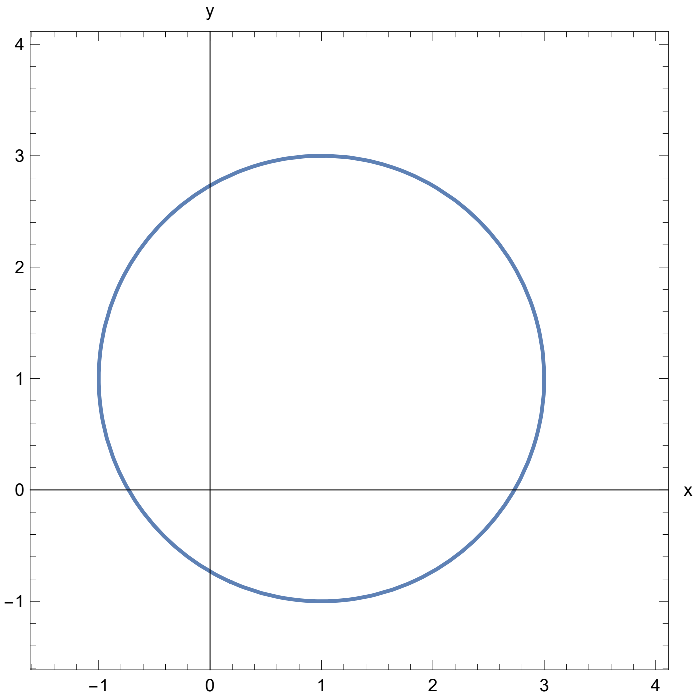
In[49]:=
```
(* Tak nie działa: ContourPlot[plot2,{x,-3/2,4},
   {y,-3/2,4},Axes→True,AxesLabel→{"x","y"},AspectRatio→1];
Konieczne jest w tym przypadku użycie "Evaluate"   *)
```
wykres konturowy

osie pra⋯ oznaczenia osi format obrazu

oblicz

In[50]:=
```
ContourPlot[Evaluate[plot2], {x, -3/2, 4}, {y, -3/2, 4},
 Axes → True, AxesLabel → {"x", "y"}, AspectRatio → 1]
```
wykres kontu··· oblicz
osie pra··· oznaczenia osi format obrazu

Out[50]=



In[51]:=
```
(* Poniżej znajdziemy równanie elipsy z ogniskami w punktach F1(0,1) i F2(0,-1). *)
```

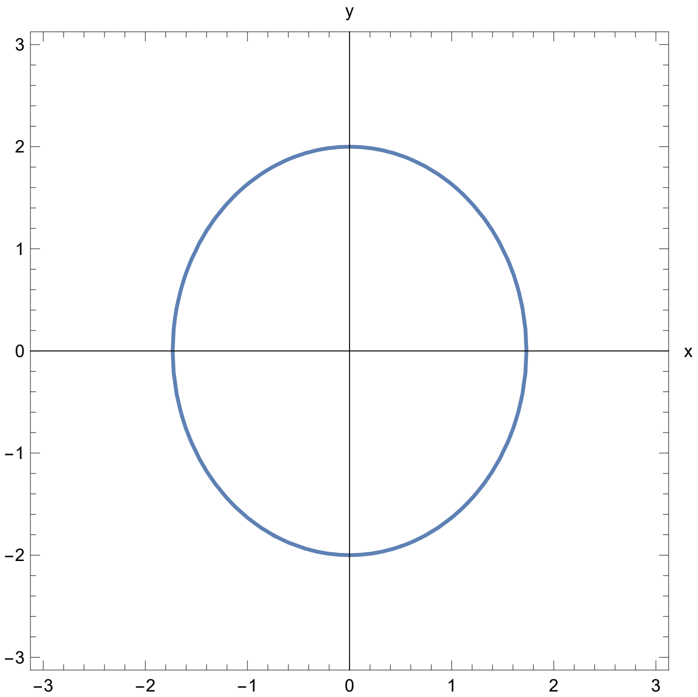
In[52]:=
```
plot3 = ComplexExpand[Abs[z + I] + Abs[z - I] == 4]
```
rozwiń na część ··· wartoś··· je··· wartoś··· jednosć u

Out[52]=

$$\sqrt{x^2 + (-1 + y)^2} + \sqrt{x^2 + (1 + y)^2} == 4$$

In[53]:=
```
ContourPlot[Evaluate[plot3], {x, -3, 3}, {y, -3, 3},
 Axes → True, AxesLabel → {"x", "y"}, AspectRatio → 1]
```
wykres kontu⋯ oblicz osie pra⋯ oznaczenia osi format obrazu

Out[53]=



In[54]:=
```
(* A w ten sposób zadajemy hiperbolę o ogniskach w punktach F1(0,1) i F2(0,-1) *)
```

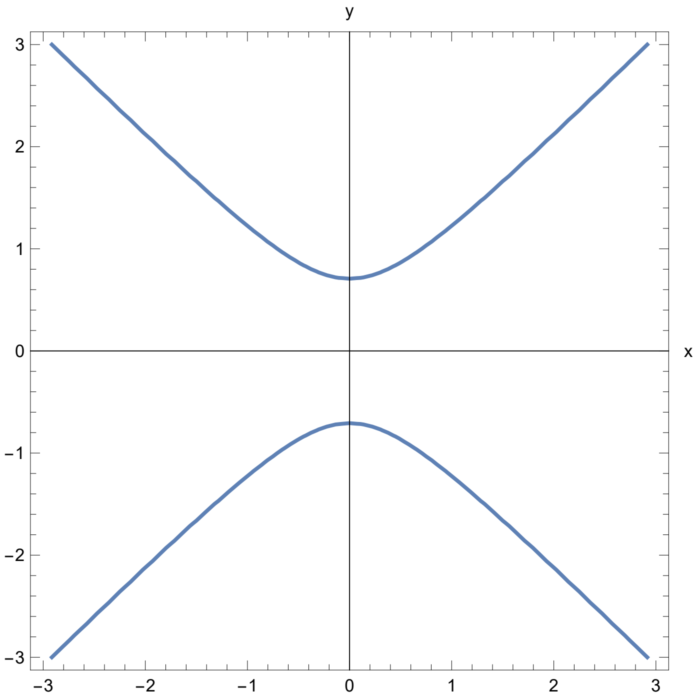
In[55]:=
```
plot4 = ComplexExpand[Abs[Abs[z + I] - Abs[z - I]] == Sqrt[2]]
```
rozwiń na część ⋯ wa⋯ wartoś⋯ je⋯ wartoś⋯ jedn⋯ pierwiastek k

Out[55]=

$$\sqrt{\left(-\sqrt{x^2+(-1+y)^2}+\sqrt{x^2+(1+y)^2}\right)^2} == \sqrt{2}$$

In[56]:=
```
ContourPlot[Evaluate[plot4], {x, -3, 3}, {y, -3, 3},
wykres kontu··· oblicz
 Axes → True, AxesLabel → {"x", "y"}, AspectRatio → 1]
 osie  pra···  oznaczenia osi             format obrazu
```

Out[56]=



In[57]:=
```
Clear[x, y, z];
wyczyść
```

Poniższe komendy pokazują, jak usprawnić znajdowanie i rysowanie pierwiastków równania czwartego stopnia

In[58]:=
```
plot5 = Solve[z^4 == 1]
        rozwiąż równanie
```

Out[58]=
```
{{z → -1}, {z → -i}, {z → i}, {z → 1}}
```

In[59]:=
```
(* Wyciąganie rozwiązań: *)
```

In[60]:=
```
z = Table[z /. plot5[[i]], {i, 1, Length[plot5]}]
    tabela                       długość
```

Out[60]=
```
{-1, -i, i, 1}
```

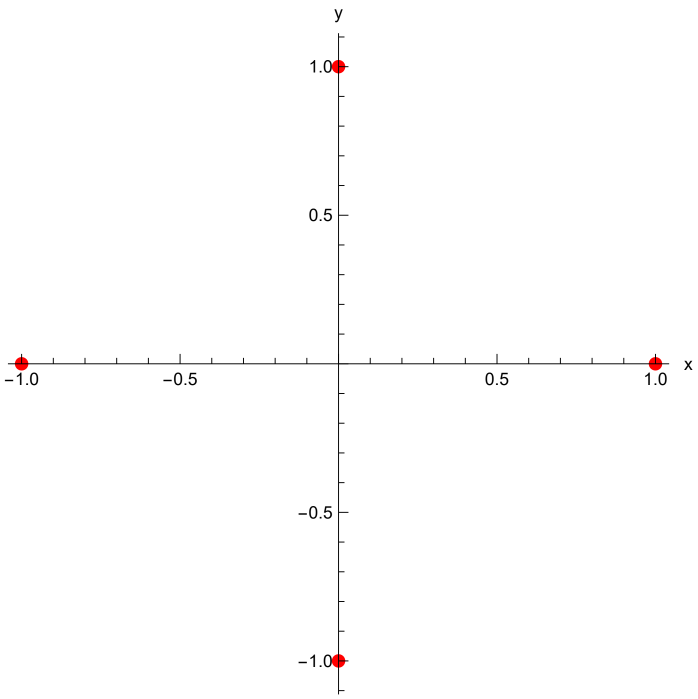
In[61]:=
```
(* Albo równoważnie tak: *)
```

In[62]:=
```
zz = z /. plot5
```

Out[62]=
```
{-1, -i, i, 1}
```

In[63]:=
```
(* Przygotowuję współrzędne punktów, by użyć ListPlot ... *)
                                          wykres danych z listy
```

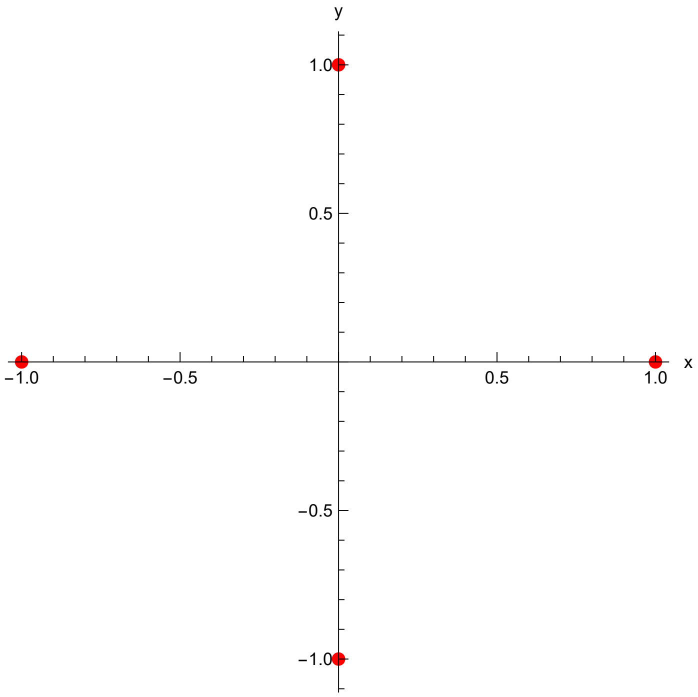
```
In[64]:= pkty = Table[{Re[z[[i]]], Im[z[[i]]]}, {i, 1, Length[z]}]
                tabela   część rzec⋯  część urojona      długość
Out[64]=
         {{-1, 0}, {0, -1}, {0, 1}, {1, 0}}

In[65]:= pkty[[1]]
Out[65]=
         {-1, 0}

In[66]:= (* Do rysowania odrębnych punktów używamy ListPlot *)
            rób                                     wykres danych z

In[67]:= ListPlot[pkty, AspectRatio → 1,
         wykres danych z li⋯ format obrazu
          AxesLabel → {"x", "y"}, PlotStyle → {Red, PointSize[Large]}]
          oznaczenia osi              styl grafiki     cz⋯  rozmiar kro⋯ duży
Out[67]=
```




```
In[68]:= (* Ale istnieje droga na
          skróty: można bezośrednio narysować listę liczb zespolonych,
         czyli zz w naszym przypadku ! *)
```

In[69]:=
```
ComplexListPlot[zz, AspectRatio → 1,
AxesLabel → {"x", "y"}, PlotStyle → {Red, PointSize[Large]}]
```
wykres listy wartości zesp··· format obrazu
oznaczenia osi styl grafiki cz··· rozmiar kro··· duży

Out[69]=



Teraz rozwiążemy równanie kwadratowe z zespolonymi współczynnikami

In[70]:=
```
ClearAll[z];
```
wyczyść wszystko

In[71]:=
```
sol = Solve[(5 - 5 * I) * z^2 - (3 - 2 * I) * z + 1 == 0, z]
```
rozwiąż równanie jedność urojona jedność urojona

Out[71]=

$$\left\{\left\{z \to \frac{1}{10} + \frac{3\,i}{10}\right\}, \left\{z \to \frac{2}{5} - \frac{i}{5}\right\}\right\}$$

In[72]:=
```
z1 = z /. sol[[1]]
```

Out[72]=

$$\frac{1}{10} + \frac{3\,i}{10}$$

In[73]:=
```
z2 = z /. sol[[2]]
```

Out[73]=

$$\frac{2}{5} - \frac{i}{5}$$

In[74]:=
```
(5 - 5 * I) * z1^2 - (3 - 2 * I) * z1 + 1 == 0
```
jedność urojona jedność urojona

Out[74]=

True

In[75]:=
```
(5 - 5 * I) * z2^2 - (3 - 2 * I) * z2 + 1 == 0
```
jedność urojona jedność urojona

Out[75]=
```
True
```

Ciekawostka: definicje funkcji sinus i cosinus dla zmiennej zespolonej bazujące na funkcji $e^z \equiv$ exp(z) !

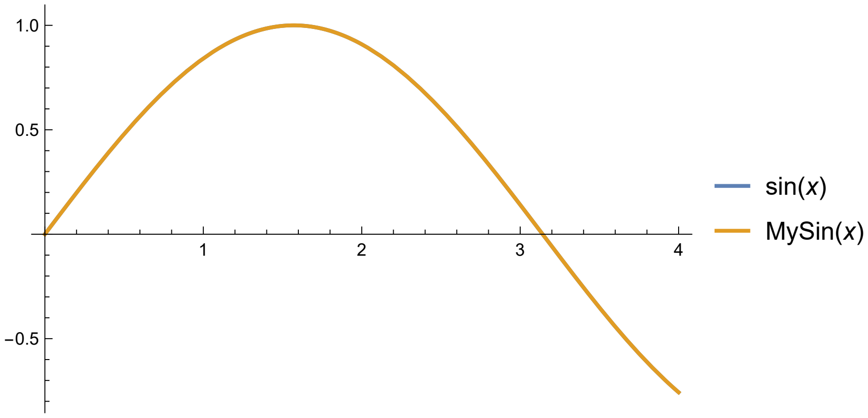
In[76]:=
```
Clear[z, x, y];
```
wyczyść

In[77]:=
```
MyCos[z_] := (E^(I * z) + E^(-I * z)) / 2
```
li··· jednoś·· licz··· jedność urojc

In[78]:=
```
MySin[z_] := (E^(I * z) - E^(-I * z)) / (2 * I)
```
li··· jednoś·· licz··· jedność uroj··· jedı

In[79]:=
```
Plot[{Sin[x], MySin[x]}, {x, 0, 4}, PlotLegends → "Expressions"]
```
wykres sinus legenda dla grafik

Out[79]=



In[80]:=
```
Sin[x] == MySin[x]
```
sinus

Out[80]=

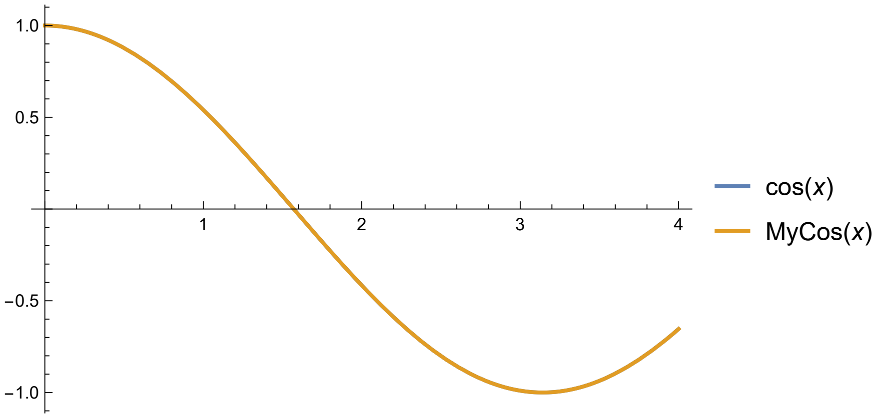
$$\mathrm{Sin}[x] == -\frac{1}{2}\, i\, \left(-e^{-i x} + e^{i x}\right)$$

In[81]:=
```
Simplify[Sin[x] == MySin[x]]
```
uprość sinus

Out[81]=
```
True
```

In[82]:=
```
Plot[{Cos[x], MyCos[x]}, {x, 0, 4}, PlotLegends → "Expressions"]
```
wykres cosinus legenda dla grafik

Out[82]=



In[83]:=
```
Cos[x] == MyCos[x]
```
cosinus

Out[83]=

$$\mathrm{Cos}[x] == \frac{1}{2}\left(e^{-i x} + e^{i x}\right)$$

In[84]:=
```
Simplify[Cos[x] == MyCos[x]]
```
uprość cosinus

Out[84]=

True

Dlatego prawdziwy jest wzór de Moivre'a (zwany też wzorem Eulera):
E^(I*z)== Cos[z] + I*Sin[z]

In[85]:=
```
Simplify[ E^(I*z) == Cos[z] + I*Sin[z] ]
```
uprość lic··· jedność ur··· cosinus j··· sinus

Out[85]=

True

In[86]:=
```
Sin[I]
```
si··· jedność urojona

Out[86]=

i Sinh[1]

In[87]:=
```
Cos[I]
```
co··· jedność urojona

Out[87]=

Cosh[1]

In[88]:=
```
(* "Jedynka trygonometryczna" jest prawdziwa dla argumentów zespolonych ... *)
```

In[89]:=
```
Simplify[Cos[z]^2 + Sin[z]^2]
```
uprość cosinus sinus

Out[89]=

1

In[90]:=
```
(* ... ale wartości absolutne sinusa i
 cosinusa nie muszą leżeć w przedziale [-1,1] ! *)
```

In[91]:=
```
N[Sin[5 * I]]
```
Out[91]=
```
0. + 74.2032 i
```

In[92]:=
```
N[Abs[Sin[5 * I]]]
```
Out[92]=
```
74.2032
```

In[93]:=
```
N[Cos[3 + 4 * I]]
```
Out[93]=
```
-27.0349 - 3.85115 i
```

In[94]:=
```
N[Abs[Cos[3 + 4 * I]]]
```
Out[94]=
```
27.3079
```

Wektory (trójwymiarowe)

W wielu wypadkach wystarcza utożsamienie trzyelementowej listy i wektora, gdzie kolejne elementy listy odpowiadają składowej x, y i z wektora

In[95]:=
```
v = {v1, v2, v3}
```
Out[95]=
```
{v1, v2, v3}
```

In[96]:=
```
MatrixForm[v]
```
Out[96]//MatrixForm=

$$\begin{pmatrix} v1 \\ v2 \\ v3 \end{pmatrix}$$

In[97]:=
```
y = {y1, y2, y3}
```
Out[97]=
```
{y1, y2, y3}
```

In[98]:=
```
MatrixForm[y]
```
Out[98]//MatrixForm=

$$\begin{pmatrix} y1 \\ y2 \\ y3 \end{pmatrix}$$

Dodawanie wektorów

In[99]:=
```
v + y
```
Out[99]=
```
{v1 + y1, v2 + y2, v3 + y3}
```

Mnożenie wektora przez liczbę

In[100]:=
```
5 * v
```
Out[100]=
```
{5 v1, 5 v2, 5 v3}
```

Iloczyn skalarny

In[101]:=
```
vdoty = v.y
```
Out[101]=
```
v1 y1 + v2 y2 + v3 y3
```
In[102]:=
```
ydotv = y.v
```
Out[102]=
```
v1 y1 + v2 y2 + v3 y3
```
In[103]:=
```
vdoty === ydotv
```
Out[103]=
```
True
```
In[104]:=
```
(* Inaczej iloczym skalarny *)
```
In[105]:=
```
Dot[v, y]
```
iloczyn wektorów, macierzy i tensorów

Out[105]=
```
v1 y1 + v2 y2 + v3 y3
```
In[106]:=
```
Dot[v, y] === v.y
```
iloczyn wektorów, macie

Out[106]=
```
True
```
In[107]:=
```
(* Norma wektora *)
```
In[108]:=
```
Norm[v]
```
norma

Out[108]=

$$\sqrt{\text{Abs}[v1]^2 + \text{Abs}[v2]^2 + \text{Abs}[v3]^2}$$

In[109]:=
```
(* wektor jednostkowy w tym samym kierunku co v *)
```
In[110]:=
```
Normalize[v]
```
znormalizuj

Out[110]=

$$\left\{\frac{v1}{\sqrt{\text{Abs}[v1]^2 + \text{Abs}[v2]^2 + \text{Abs}[v3]^2}}, \frac{v2}{\sqrt{\text{Abs}[v1]^2 + \text{Abs}[v2]^2 + \text{Abs}[v3]^2}}, \frac{v3}{\sqrt{\text{Abs}[v1]^2 + \text{Abs}[v2]^2 + \text{Abs}[v3]^2}}\right\}$$

In[111]:=

```
Normalize[{1, 2, 5}]
```
znormalizuj

Out[111]=

$$\left\{\frac{1}{\sqrt{30}}, \sqrt{\frac{2}{15}}, \sqrt{\frac{5}{6}}\right\}$$

In[112]:=

```
(* Sprawdzenie, czy mamy do czynienia z wektorem jednostkowym *)
```

In[113]:=

```
Norm[Normalize[{1, 2, 5}]]
```
norma znormalizuj

Out[113]=

1

In[114]:=

```
(* To jest częstym tematem zadań: kąt między wektorami *)
```

In[115]:=

```
VectorAngle[{1, 0, 0}, {0, 1, 0}]
```
kąt pomiędzy wektorami

Out[115]=

$$\frac{\pi}{2}$$

In[116]:=

```
VectorAngle[{1, 0, 0}, {0, 0, 0}]
```
kąt pomiędzy wektorami

Out[116]=

Indeterminate

In[117]:=

```
MyVectorAngle[z1_, z2_] :=
 If[Norm[z1] > 0 && Norm[z2] > 0, ArcCos[z1.z2 / (Norm[z1] * Norm[z2])], 0]
```
… norma norma arcus cosinus norma norma

In[118]:=

```
(* Jak to działa ? Przepis jest bardzo prosty, zakładając,
że mamy do czynienia z niezerowymi wektorami. Jeśli przynajmniej jeden z wektorów
 jest wektorem zerowym, przyjmuję, że kąt między wektorami wynosi zero.  *)
```

In[119]:=

```
MyVectorAngle[{1, 0, 0}, {0, 1, 0}]
```

Out[119]=

$$\frac{\pi}{2}$$

In[120]:=

```
MyVectorAngle[{1, 0, 0}, {0, 0, 0}]
```

Out[120]=

0

Iloczyn wektorowy

In[121]:=

```
z1 = Cross[v, y]
```
iloczyn wektorov

Out[121]=

$\{-v3\ y2 + v2\ y3,\ v3\ y1 - v1\ y3,\ -v2\ y1 + v1\ y2\}$

In[122]:=

```
(* Inaczej *)
```

In[123]:=

```
z2 = v × y  (* to nie jest znak "iks",
  tylko znak działania wzięty z palety "Writing Assistant"  ! *)
```

Out[123]=

$\{-v3\, y2 + v2\, y3,\ v3\, y1 - v1\, y3,\ -v2\, y1 + v1\, y2\}$

In[124]:=

```
Cross[v, y] === v × y
iloczyn wektorowy
```

Out[124]=

True

In[125]:=

```
(* Z wektorów niezależnych liniowo można stworzyć zbiór wektorów
 wzajemnie prostopadłych i unormowanych, czyli bazę ortonormalną *)
```

In[126]:=

```
baza = Orthogonalize[{{1, 2, 3}, {0, 3, 4}, {1, 5, -2}}]
       znajdź bazę ortogonalną
```

Out[126]=

$$\left\{\left\{\frac{1}{\sqrt{14}}, \sqrt{\frac{2}{7}}, \frac{3}{\sqrt{14}}\right\}, \left\{-\frac{9}{\sqrt{91}}, \frac{3}{\sqrt{91}}, \frac{1}{\sqrt{91}}\right\}, \left\{\frac{1}{\sqrt{26}}, 2\sqrt{\frac{2}{13}}, -\frac{3}{\sqrt{26}}\right\}\right\}$$

In[127]:=

```
Do[{Print["  baza[[", i, "]].baza[[", j, "]]= ", baza[[i]].baza[[j]]]},
rób  drukuj
 {i, 1, 3}, {j, 1, 3}]
```

```
  baza[[1]].baza[[1]]= 1
  baza[[1]].baza[[2]]= 0
  baza[[1]].baza[[3]]= 0
  baza[[2]].baza[[1]]= 0
  baza[[2]].baza[[2]]= 1
  baza[[2]].baza[[3]]= 0
  baza[[3]].baza[[1]]= 0
  baza[[3]].baza[[2]]= 0
  baza[[3]].baza[[3]]= 1
```

In[128]:=

(* Teraz dowolny wektor r może być zapisany w postaci kombinacji liniowej r= $\sum_{i=1}^{3} c_i$*baza[[i]], gdzie c_i=r.baza[[i]] *)

In[129]:=

```
ClearAll[r1, r2, r3];
wyczyść wszystko
```

In[130]:=

```
r = {r1, r2, r3}
```

Out[130]=

{r1, r2, r3}

In[131]:=

```
Do[c[i] = r.baza[[i]], {i, 1, 3}];
rób
```

In[132]:=

```
c[1]
```

Out[132]=

$$\frac{r1}{\sqrt{14}} + \sqrt{\frac{2}{7}}\, r2 + \frac{3\, r3}{\sqrt{14}}$$

In[133]:=

```
c[2]
```

Out[133]=

$$-\frac{9\, r1}{\sqrt{91}} + \frac{3\, r2}{\sqrt{91}} + \frac{r3}{\sqrt{91}}$$

In[134]:=

```
Simplify[r - Sum[c[i] * baza[[i]], {i, 1, 3}]]
uprość          sumowanie
```

Out[134]=

```
{0, 0, 0}
```

Wektory mogą mieć zespolone składowe

In[135]:=

```
f = {I, 1, 0}
     jedność urc
```

Out[135]=

$\{i, 1, 0\}$

In[136]:=

```
h = {1 - I, 2 * I + 4, 3 * I}
       jedn··· jedność ··· jedı
```

Out[136]=

$\{1 - i, 4 + 2 i, 3 i\}$

In[137]:=

```
(* Norma wektora powinna byc pierwiastkiem kwadratowym
 z iloczynu skalarnego wektora przez ten sam wektor,
ale taki iloczyn skalarny może mieć wartość zero lub być liczbą zespoloną   *)
```

In[138]:=

```
f.f
```

Out[138]=

```
0
```

In[139]:=

```
h.h
```

Out[139]=

$3 + 14 i$

In[140]:=

```
(* Trzeba zmienić definicję iloczynu skalarnego, bo, jak widać powyżej,
kwadrat niezerowego wektora może dać zero lub liczbę zespoloną ! *)
```

In[141]:=

```
Dotcmplx[z1_, z2_] := Conjugate[z1].z2 (* nie jest przemienny ! *)
                      sprzężenie zespolone
```

In[142]:=

```
Normcmplx[z_] := Sqrt[Dotcmplx[z, z]]
                 pierwiastek kwadratowy
```

In[143]:=

```
Normcmplx[f]
```

Out[143]=

$\sqrt{2}$

In[144]:=

```
Normcmplx[h]
```

Out[144]=

$\sqrt{31}$

In[145]:=

```
(* Funkcja "Norm" wie o tym i liczy poprawnie normę zespolonego wektora *)
```

norma

In[146]:=

```
Norm[f]
```

norma

Out[146]=

$\sqrt{2}$

In[147]:=

```
Norm[h]
```

norma

Out[147]=

$\sqrt{31}$

PODSTAWOWE TOŻSAMOŚCI WEKTOROWE

Udowodnić następujące tożsamości wektorowe spełnione dla dowolnych wektorów $\vec{A}$, $\vec{B}$, $\vec{C}$ i $\vec{D}$:

1. $(\vec{A} \times \vec{B})^2 + (\vec{A} \cdot \vec{B})^2 = (\vec{A})^2 (\vec{B})^2$
2. $\vec{A} \cdot (\vec{B} \times \vec{C}) = \vec{C} \cdot (\vec{A} \times \vec{B}) = \vec{B} \cdot (\vec{C} \times \vec{A}) =$
   $-\vec{A} \cdot (\vec{C} \times \vec{B}) = -\vec{C} \cdot (\vec{B} \times \vec{A}) = -\vec{B} \cdot (\vec{A} \times \vec{C})$
3. $\vec{A} \times (\vec{B} \times \vec{C}) = (\vec{A} \cdot \vec{C})\vec{B} - (\vec{A} \cdot \vec{B})\vec{C}$
4. $\vec{A} \times (\vec{B} \times \vec{C}) + \vec{B} \times (\vec{C} \times \vec{A}) + \vec{C} \times (\vec{A} \times \vec{B}) = 0$
5. $(\vec{A} \times \vec{B}) \cdot (\vec{C} \times \vec{D}) = (\vec{A} \cdot \vec{C})(\vec{B} \cdot D) - (\vec{A} \cdot \vec{D})(\vec{B} \cdot C)$
6. $(\vec{A} \times \vec{B}) \times (\vec{C} \times \vec{D}) = [\vec{A} \cdot (\vec{B} \times \vec{D})]\, \vec{C} - [\vec{A} \cdot (\vec{B} \times \vec{C})]\, \vec{D}$
7. $\vec{A} \times (\vec{B} \times (\vec{C} \times \vec{D})) = (\vec{B} \cdot \vec{D})(\vec{A} \times \vec{C}) - (\vec{B} \cdot \vec{C})(\vec{A} \times \vec{D})$
8. $(\vec{A} \times \vec{B}) \cdot (\vec{C} \times \vec{D}) + (\vec{B} \times \vec{C}) \cdot (\vec{A} \times \vec{D}) + (\vec{C} \times \vec{A}) \cdot (\vec{B} \times \vec{D}) = 0$

Dowody powyższych, bardzo potrzebnych twierdzeń, są bardzo łatwe do uzyskania przy pomocy programu *Mathematica*. W tych dowodach wystarczy używać trzyelementowych list

In[148]:=

```
Clear[a, b, c, d];
```

wyczyść

In[149]:=

```
a = {a1, a2, a3};
```

In[150]:=
```
b = {b1, b2, b3};
```

In[151]:=
```
c = {c1, c2, c3};
```

In[152]:=
```
d = {d1, d2, d3};
```

In[153]:=
```
(* Tw. 1 *)
```

In[154]:=
```
left1 = Cross[a, b].Cross[a, b] + (a.b)^2
```
iloczyn wekto··· iloczyn wektorowy

Out[154]=
$$(-a2\,b1 + a1\,b2)^2 + (a3\,b1 - a1\,b3)^2 + (-a3\,b2 + a2\,b3)^2 + (a1\,b1 + a2\,b2 + a3\,b3)^2$$

In[155]:=
```
right1 = (a.a) * (b.b)
```

Out[155]=
$$(a1^2 + a2^2 + a3^2)\,(b1^2 + b2^2 + b3^2)$$

In[156]:=
```
Simplify[left1 == right1]
```
uprość

Out[156]=
True

In[157]:=
```
(* Tw. 2 *)
```

In[158]:=
```
(* Uwaga: teraz dla odmiany używam palety Writing Assistant,
gdzie w grupie Typesetting jest zdefiniowany operator "cross" *)
```

In[159]:=
```
Simplify[a.(b×c) == c.(a×b) == b.(c×a) == -a.(c×b) == -c.(b×a) == -b.(a×c)]
```
uprość

Out[159]=
True

In[160]:=
```
(* Tw. 3 *)
```

In[161]:=
```
left3 = Cross[a, Cross[b, c]]
```
iloczyn w··· iloczyn wektorowy

Out[161]=
{-a2 b2 c1 - a3 b3 c1 + a2 b1 c2 + a3 b1 c3,
a1 b2 c1 - a1 b1 c2 - a3 b3 c2 + a3 b2 c3, a1 b3 c1 + a2 b3 c2 - a1 b1 c3 - a2 b2 c3}

In[162]:=
```
right3 = (a.c) * b - (a.b) * c
```

Out[162]=
{-((a1 b1 + a2 b2 + a3 b3) c1) + b1 (a1 c1 + a2 c2 + a3 c3),
-((a1 b1 + a2 b2 + a3 b3) c2) + b2 (a1 c1 + a2 c2 + a3 c3),
-((a1 b1 + a2 b2 + a3 b3) c3) + b3 (a1 c1 + a2 c2 + a3 c3)}

In[163]:=
```
Simplify[left3 == right3]
```
uprość

Out[163]=
```
True
```

In[164]:=
```
(* Tw. 4 *)
```

In[165]:=
```
left4 = a × (b × c) + b × (c × a) + c × (a × b)
```

Out[165]=
```
{0, 0, 0}
```

In[166]:=
```
Simplify[left4 == {0, 0, 0}]
```
uprość

Out[166]=
```
True
```

In[167]:=
```
(* Tw. 5 *)
```

In[168]:=
```
left5 = (a × b) . (c × d)
```

Out[168]=
```
(-a2 b1 + a1 b2) (-c2 d1 + c1 d2) +
  (a3 b1 - a1 b3) (c3 d1 - c1 d3) + (-a3 b2 + a2 b3) (-c3 d2 + c2 d3)
```

In[169]:=
```
right5 = (a.c) * (b.d) - (a.d) * (b.c)
```

Out[169]=
```
-((b1 c1 + b2 c2 + b3 c3) (a1 d1 + a2 d2 + a3 d3)) + (a1 c1 + a2 c2 + a3 c3) (b1 d1 + b2 d2 + b3 d3)
```

In[170]:=
```
Simplify[left5 == right5]
```
uprość

Out[170]=
```
True
```

In[171]:=
```
(* Tw. 6 *)
```

In[172]:=
```
left6 = (a × b) × (c × d)
```

Out[172]=
```
{-a3 b1 c2 d1 + a1 b3 c2 d1 + a2 b1 c3 d1 - a1 b2 c3 d1 + a3 b1 c1 d2 - a1 b3 c1 d2 -
   a2 b1 c1 d3 + a1 b2 c1 d3, -a3 b2 c2 d1 + a2 b3 c2 d1 + a3 b2 c1 d2 - a2 b3 c1 d2 +
   a2 b1 c3 d2 - a1 b2 c3 d2 - a2 b1 c2 d3 + a1 b2 c2 d3, -a3 b2 c3 d1 + a2 b3 c3 d1 +
   a3 b1 c3 d2 - a1 b3 c3 d2 + a3 b2 c1 d3 - a2 b3 c1 d3 - a3 b1 c2 d3 + a1 b3 c2 d3}
```

In[173]:=
```
right6 = (a.(b×d)) * c - (a.(b×c)) * d
```
Out[173]=
```
{-((a3 (-b2 c1 + b1 c2) + a2 (b3 c1 - b1 c3) + a1 (-b3 c2 + b2 c3)) d1) +
   c1 (a3 (-b2 d1 + b1 d2) + a2 (b3 d1 - b1 d3) + a1 (-b3 d2 + b2 d3)),
  -((a3 (-b2 c1 + b1 c2) + a2 (b3 c1 - b1 c3) + a1 (-b3 c2 + b2 c3)) d2) +
   c2 (a3 (-b2 d1 + b1 d2) + a2 (b3 d1 - b1 d3) + a1 (-b3 d2 + b2 d3)),
  -((a3 (-b2 c1 + b1 c2) + a2 (b3 c1 - b1 c3) + a1 (-b3 c2 + b2 c3)) d3) +
   c3 (a3 (-b2 d1 + b1 d2) + a2 (b3 d1 - b1 d3) + a1 (-b3 d2 + b2 d3))}
```
In[174]:=
```
Simplify[left6 == right6]
uprość
```
Out[174]=
```
True
```
In[175]:=
```
(* Tw. 7 *)
```
In[176]:=
```
left7 = a × (b × (c × d))
```
Out[176]=
```
{-a3 b1 c2 d1 + a2 b1 c3 d1 + a3 b1 c1 d2 + a2 b2 c3 d2 + a3 b3 c3 d2 - a2 b1 c1 d3 -
   a2 b2 c2 d3 - a3 b3 c2 d3, -a3 b2 c2 d1 - a1 b1 c3 d1 - a3 b3 c3 d1 + a3 b2 c1 d2 -
   a1 b2 c3 d2 + a1 b1 c1 d3 + a3 b3 c1 d3 + a1 b2 c2 d3, a1 b1 c2 d1 + a2 b2 c2 d1 +
   a2 b3 c3 d1 - a1 b1 c1 d2 - a2 b2 c1 d2 - a1 b3 c3 d2 - a2 b3 c1 d3 + a1 b3 c2 d3}
```
In[177]:=
```
right7 = (b.d) * (a × c) - (b.c) * (a × d)
```
Out[177]=
```
{-((b1 c1 + b2 c2 + b3 c3) (-a3 d2 + a2 d3)) + (-a3 c2 + a2 c3) (b1 d1 + b2 d2 + b3 d3),
  -((b1 c1 + b2 c2 + b3 c3) (a3 d1 - a1 d3)) + (a3 c1 - a1 c3) (b1 d1 + b2 d2 + b3 d3),
  -((b1 c1 + b2 c2 + b3 c3) (-a2 d1 + a1 d2)) + (-a2 c1 + a1 c2) (b1 d1 + b2 d2 + b3 d3)}
```
In[178]:=
```
Simplify[left7 == right7]
uprość
```
Out[178]=
```
True
```
In[179]:=
```
(* Tw. 8 *)
```
In[180]:=
```
left8 = (a × b).(c × d) + (b × c).(a × d) + (c × a).(b × d)
```
Out[180]=
```
(-b2 c1 + b1 c2) (-a2 d1 + a1 d2) +
 (a2 c1 - a1 c2) (-b2 d1 + b1 d2) + (-a2 b1 + a1 b2) (-c2 d1 + c1 d2) +
 (b3 c1 - b1 c3) (a3 d1 - a1 d3) + (-b3 c2 + b2 c3) (-a3 d2 + a2 d3) +
 (-a3 c1 + a1 c3) (b3 d1 - b1 d3) + (a3 c2 - a2 c3) (-b3 d2 + b2 d3) +
 (a3 b1 - a1 b3) (c3 d1 - c1 d3) + (-a3 b2 + a2 b3) (-c3 d2 + c2 d3)
```
In[181]:=
```
Simplify[left8 == 0]
uprość
```
Out[181]=
```
True
```

Macierze

In[182]:=

```
ClearAll[A, B, H, X, Y, Z];
```

wyczyść wszystko

“Macierz – układ liczb, symboli lub wyrażeń zapisanych w postaci prostokątnej tablicy.” (Wikipedia)

Nas chwilowo interesują tablice liczb.

A= $\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{bmatrix}$. Ta macierz ma n wierszy i m kolumn.

Do tej pory używaliśmy list trzyelementowych, mówiąc o wektorach i nie zastanawialiśmy się, czy mamy do czynienia z wierszem, czy z kolumną !

Czy można mieszać trzyelementowe listy i “column vectors” ? NIE MOŻNA, choć ich postać macierzowa “MatrixForm” jest identyczna.

In[183]:=

```
v = {v1, v2, v3}  (* lista *)
```

Out[183]=

```
{v1, v2, v3}
```

In[184]:=

```
MatrixForm[v]
```

postać macierzy

Out[184]//MatrixForm=

$\begin{pmatrix} v1 \\ v2 \\ v3 \end{pmatrix}$

In[185]:=

```
k = {{k1}, {k2}, {k3}}  (* kolumna *)
```

Out[185]=

```
{{k1}, {k2}, {k3}}
```

In[186]:=

```
MatrixForm[k]
```

postać macierzy

Out[186]//MatrixForm=

$\begin{pmatrix} k1 \\ k2 \\ k3 \end{pmatrix}$

In[187]:=
```
k.v
```

Dot: Tensors {{k1}, {k2}, {k3}} and {v1, v2, v3} have incompatible shapes.

Out[187]=
```
{{k1}, {k2}, {k3}}.{v1, v2, v3}
```

WNIOSEK: Wymagana jest ostrożność, jeśli chcemy konsekwentnie stosować zapis używany w podręcznikach algebry liniowej !

Jak wprowadzać macierz w notatniku ? Reguła jest bardzo prosta: macierz to lista wierszy !

In[188]:=
```
(* Poniżej macierz, która jest pojedynczym wierszem *)
```

In[189]:=
```
w = {{w1, w2, w3}} (* wiersz *)
```

Out[189]=
```
{{w1, w2, w3}}
```

In[190]:=
```
MatrixForm[w]
```
postać macierzy

Out[190]//MatrixForm=

$$\begin{pmatrix} w1 & w2 & w3 \end{pmatrix}$$

In[191]:=
```
(* A to przypomnienie: macierz k jest pojedynczą kolumną,
więc jej wiersze zawierają tylko po jednej liczbie  *)
```

In[192]:=
```
k = {{k1}, {k2}, {k3}} (* kolumna *)
```

Out[192]=
```
{{k1}, {k2}, {k3}}
```

In[193]:=
```
MatrixForm[k]
```
postać macierzy

Out[193]//MatrixForm=

$$\begin{pmatrix} k1 \\ k2 \\ k3 \end{pmatrix}$$

In[194]:=
```
A = {{1, 0, 3, 4}, {2, 4, -2, 8}, {1, 0, -4, 6}, {3, -7, 9, 1}}
```

Out[194]=
```
{{1, 0, 3, 4}, {2, 4, -2, 8}, {1, 0, -4, 6}, {3, -7, 9, 1}}
```

In[195]:=
```
MatrixForm[A]
```
postać macierzy

Out[195]//MatrixForm=

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

In[196]:=

```
B = {{-2, 1, 8, 0}, {7, 0, -2, 2}, {0, 1, 2, 3}, {2, 1, -9, 3}}
```

Out[196]=

```
{{-2, 1, 8, 0}, {7, 0, -2, 2}, {0, 1, 2, 3}, {2, 1, -9, 3}}
```

In[197]:=

```
MatrixForm[B]
```

postać macierzy

Out[197]//MatrixForm=

$$\begin{pmatrix} -2 & 1 & 8 & 0 \\ 7 & 0 & -2 & 2 \\ 0 & 1 & 2 & 3 \\ 2 & 1 & -9 & 3 \end{pmatrix}$$

In[198]:=

```
X = {{1, 2}, {-2, 3}, {4, 5}, {6, -5}}
```

Out[198]=

```
{{1, 2}, {-2, 3}, {4, 5}, {6, -5}}
```

In[199]:=

```
MatrixForm[X]
```

postać macierzy

Out[199]//MatrixForm=

$$\begin{pmatrix} 1 & 2 \\ -2 & 3 \\ 4 & 5 \\ 6 & -5 \end{pmatrix}$$

In[200]:=

```
Y = {{-1}, {2}, {0}, {9}}
```

Out[200]=

```
{{-1}, {2}, {0}, {9}}
```

In[201]:=

```
(* Teraz Y jest z zamierzenia "wektorem kolumnowym" ! *)
```

In[202]:=

```
MatrixForm[Y]
```

postać macierzy

Out[202]//MatrixForm=

$$\begin{pmatrix} -1 \\ 2 \\ 0 \\ 9 \end{pmatrix}$$

In[203]:=

```
Z = {{-3, 4, 5, 2}}
```

Out[203]=

```
{{-3, 4, 5, 2}}
```

In[204]:=

```
(* Teraz Z jest z zamierzenia "wektorem wierszowym" ! *)
```

In[205]:=

```
MatrixForm[Z]
```

postać macierzy

Out[205]//MatrixForm=

$$\begin{pmatrix} -3 & 4 & 5 & 2 \end{pmatrix}$$

Możemy wykonywać różne działania

In[206]:=
```
(* Mnożymy macierz przez liczbę *)
```

In[207]:=
```
2 * A
```

Out[207]=
```
{{2, 0, 6, 8}, {4, 8, -4, 16}, {2, 0, -8, 12}, {6, -14, 18, 2}}
```

In[208]:=
```
MatrixForm[2 * A]
```
postać macierzy

Out[208]//MatrixForm=

$$\begin{pmatrix} 2 & 0 & 6 & 8 \\ 4 & 8 & -4 & 16 \\ 2 & 0 & -8 & 12 \\ 6 & -14 & 18 & 2 \end{pmatrix}$$

In[209]:=
```
(* Kombinacja liniowa macierzy może być utworzona,
jeśli macierze A i B mają te same rozmiary ! *)
```

In[210]:=
```
2 * A + 3 * B
```

Out[210]=
```
{{-4, 3, 30, 8}, {25, 8, -10, 22}, {2, 3, -2, 21}, {12, -11, -9, 11}}
```

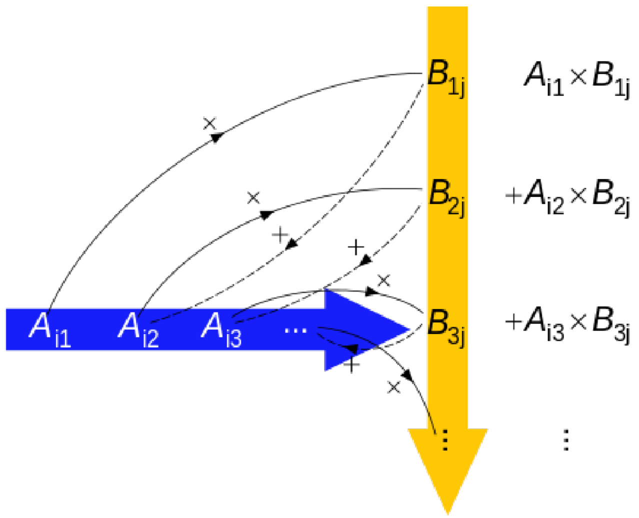
In[211]:=
```
MatrixForm[2 * A + 3 * B]
```
postać macierzy

Out[211]//MatrixForm=

$$\begin{pmatrix} -4 & 3 & 30 & 8 \\ 25 & 8 & -10 & 22 \\ 2 & 3 & -2 & 21 \\ 12 & -11 & -9 & 11 \end{pmatrix}$$

Mnożenie macierzowe, “wiersz razy kolumna”

$(A.B)_{ij} = \sum_{k=1}^{n} A_{ik} {}^* B_{kj}$, gdzie n jest liczbą kolumn macierzy A i liczbą wierszy macierzy B



(obrazek z Wikipedii)

Mnożenie macierzowe A.B, tzw. mnożenie wiersz przez kolumnę. Znakiem tego mnożenia też jest kropka wprowadzana wprost z klawiatury, której wcześniej używaliśmy dla oznaczenia iloczynu skalarnego, opisując wektory przy pomocy list ! Nie każde mnożenie macierzowe jest wykonalne: wymagane jest aby liczba kolumn macierzy A była równa liczbie wierszy macierzy B. Mnożenie macierzowe na ogół NIE JEST PRZEMIENNE.

In[212]:=
```
(* Najprostszy przykład: mnożenie kolumny przez wiersz ... *)
```

In[213]:=
```
k.w // MatrixForm
```
postać macierzy

Out[213]//MatrixForm=

$$\begin{pmatrix} k1\,w1 & k1\,w2 & k1\,w3 \\ k2\,w1 & k2\,w2 & k2\,w3 \\ k3\,w1 & k3\,w2 & k3\,w3 \end{pmatrix}$$

In[214]:=
```
(* ...jest czymś zupełnie innym niż mnożenie wiersza przez kolumnę, *)
```

In[215]:=
```
w.k // MatrixForm
```
postać macierzy

Out[215]//MatrixForm=

$$\begin{pmatrix} k1\,w1 + k2\,w2 + k3\,w3 \end{pmatrix}$$

In[216]:=
```
(* chociaż oba działania (k.w oraz w.k) są wykonalne. *)
```

In[217]:=
```
A.B
```

Out[217]=
```
{{6, 8, -22, 21}, {40, 8, -68, 26}, {10, 3, -54, 6}, {-53, 13, 47, 16}}
```

In[218]:=
```
MatrixForm[A.B]
```
postać macierzy

Out[218]//MatrixForm=

$$\begin{pmatrix} 6 & 8 & -22 & 21 \\ 40 & 8 & -68 & 26 \\ 10 & 3 & -54 & 6 \\ -53 & 13 & 47 & 16 \end{pmatrix}$$

In[219]:=
```
A.X
```

Out[219]=
```
{{37, -3}, {34, -34}, {21, -48}, {59, 25}}
```

In[220]:=
```
MatrixForm[A.X]
```
postać macierzy

Out[220]//MatrixForm=

$$\begin{pmatrix} 37 & -3 \\ 34 & -34 \\ 21 & -48 \\ 59 & 25 \end{pmatrix}$$

WAŻNE: Instrukcja

F=MatrixForm[A.X]

lub

F=A.X //MatrixForm

jest **błędna, jeśli F ma być macierzą !** W programie *Mathematica* “macierz” i “postać macierzowa” to dwie różne rzeczy.

In[221]:=
```
A.Y
```
Out[221]=
```
{{35}, {78}, {53}, {-8}}
```
In[222]:=
```
MatrixForm[A.Y]
```
postać macierzy

Out[222]//MatrixForm=

$$\begin{pmatrix} 35 \\ 78 \\ 53 \\ -8 \end{pmatrix}$$

In[223]:=
```
Z.A
```
Out[223]=
```
{{16, 2, -19, 52}}
```
In[224]:=
```
MatrixForm[Z.A]
```
postać macierzy

Out[224]//MatrixForm=

$$\begin{pmatrix} 16 & 2 & -19 & 52 \end{pmatrix}$$

In[225]:=
```
Z.A.Y
```
Out[225]=
```
{{456}}
```
Uwagi:
(1) Dla Programu Mathematica {{456}} to nie jest po prostu liczba 456, ale macierz, która ma jeden wiersz i jedną kolumnę !
(2) Mnożenie macierzy jest łączne, więc nie ma potrzeby wskazywania na kolejność działań

In[226]:=
```
(* Przypominam macierz A *)
```
In[227]:=
```
MatrixForm[A]
```
postać macierzy

Out[227]//MatrixForm=

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

In[228]:=
```
(* Macierz transponowana do A: zamieniamy wiersze z kolumnami *)
```
In[229]:=
```
Transpose[A]
```
transpozycja

Out[229]=
```
{{1, 2, 1, 3}, {0, 4, 0, -7}, {3, -2, -4, 9}, {4, 8, 6, 1}}
```

In[230]:=

```
MatrixForm[Transpose[A]]
```
postać macie·· transpozycja

Out[230]//MatrixForm=

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

In[231]:=

```
(* Na ćwiczeniach sprawdzimy na kilku przykładach twierdzenie,
że Transpose[A.B]=Transpose[B].Transpose[A] *)
```
transpozycja transpozycja transpozycja

In[232]:=

```
A.A.A
```

Out[232]=

```
{{65, -294, 230, 28}, {138, -412, 188, 312},
 {-38, -70, -289, 148}, {-40, -217, 188, -289}}
```

Uwaga: A.A.A===MatrixPower[A,3] to coś zupełnie innego niż A^3 !!!

In[233]:=

```
MatrixPower[A, 3]
```
potęga macierzy

Out[233]=

```
{{65, -294, 230, 28}, {138, -412, 188, 312},
 {-38, -70, -289, 148}, {-40, -217, 188, -289}}
```

In[234]:=

```
A^3 (* to macierz, której elementy są sześcianami elementów macierzy A *)
```

Out[234]=

```
{{1, 0, 27, 64}, {8, 64, -8, 512}, {1, 0, -64, 216}, {27, -343, 729, 1}}
```

Można więc zadziałać funkcją na całą macierz:

In[235]:=

```
Sin[A] (* to macierz, której elementy są sinusami elementów macierzy A *)
```
sinus

Out[235]=

```
{{Sin[1], 0, Sin[3], Sin[4]}, {Sin[2], Sin[4], -Sin[2], Sin[8]},
 {Sin[1], 0, -Sin[4], Sin[6]}, {Sin[3], -Sin[7], Sin[9], Sin[1]}}
```

Duża ostrożność jest tutaj wymagana,
bo zwykle funkcję macierzy rozumiemy zupełnie inaczej,
przez rozwinięcie w szereg. Wowczas sin (A) byłby równy

$$A - \frac{A.A.A}{3\,!} + \frac{A.A.A.A.A}{5\,!} - \frac{A.A.A.A.A.A.A}{7\,!} + \ldots$$

Dla macierzy zespolonych mamy operację, która łączy w sobie transponowanie i sprzężenie zespolone.
Jest to tzw. sprzężenie hermitowskie. Odpowiada jej komenda ConjugateTranspose.

In[236]:=

```
H = A + I * B (* to jest macierz o wyrazach zespolonych ! *)
```

jedność urojona

Out[236]=

```
{{1 - 2 i, i, 3 + 8 i, 4}, {2 + 7 i, 4, -2 - 2 i, 8 + 2 i},
 {1, i, -4 + 2 i, 6 + 3 i}, {3 + 2 i, -7 + i, 9 - 9 i, 1 + 3 i}}
```

In[237]:=

```
MatrixForm[H]
```

postać macierzy

Out[237]//MatrixForm=

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

In[238]:=

```
G = ConjugateTranspose[H]
```

sprzężenie hermitowskie macie

Out[238]=

```
{{1 + 2 i, 2 - 7 i, 1, 3 - 2 i}, {-i, 4, -i, -7 - i},
 {3 - 8 i, -2 + 2 i, -4 - 2 i, 9 + 9 i}, {4, 8 - 2 i, 6 - 3 i, 1 - 3 i}}
```

In[239]:=

```
MatrixForm[G]
```

postać macierzy

Out[239]//MatrixForm=

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

Prawdziwe jest twierdzenie: ConjugateTranspose[A.B]=ConjugateTranspose[B].ConjugateTranspose[A]