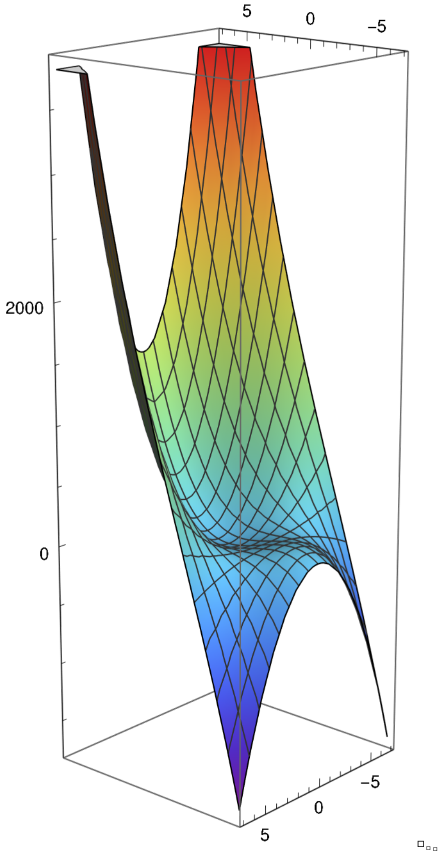
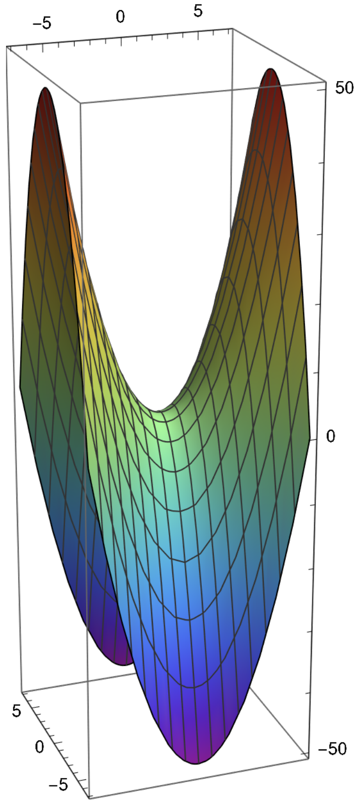
```
In[2]:= SetOptions[Plot3D(*Or whichever plot you desire*),
          ColorFunction → "Rainbow"(*One of many options*)];
        (*Ukazka*)

In[ ∘ ]:= Plot3D[2 x ^ 3 + 9 x * y ^ 2 + 15 x ^ 2 + 27 y ^ 2, {x, -7, 7},
          {y, -7, 7}, ColorFunction → "Rainbow", BoxRatios → {1, 1, 3}]
```



```
In[ ◦ ]:= Plot3D[x^2 - y^2, {x, -7, 7}, {y, -7, 7}, ColorFunction → "Rainbow", BoxRatios → {1, 1, 3}]
```

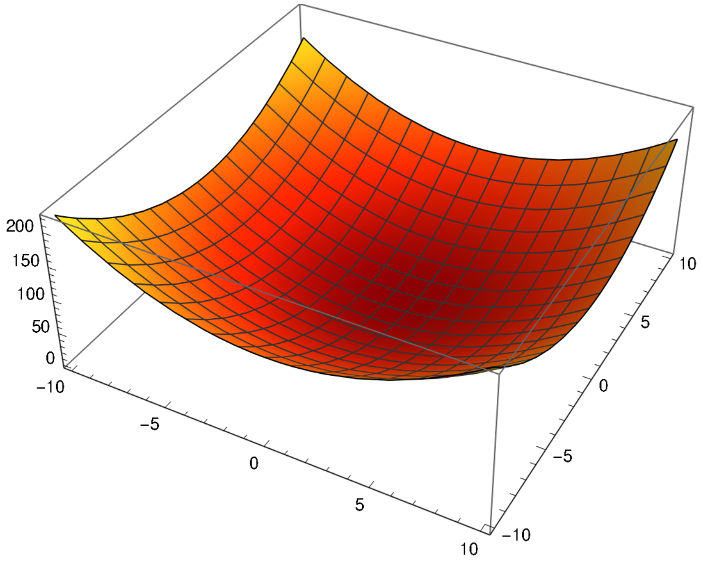
Out[◦]=



(1 a)

```
In[ ◦ ]:= Plot3D[x^2 + (y - 1)^2, {x, -10, 10}, {y, -10, 10}, ColorFunction → "SolarColors"]
```

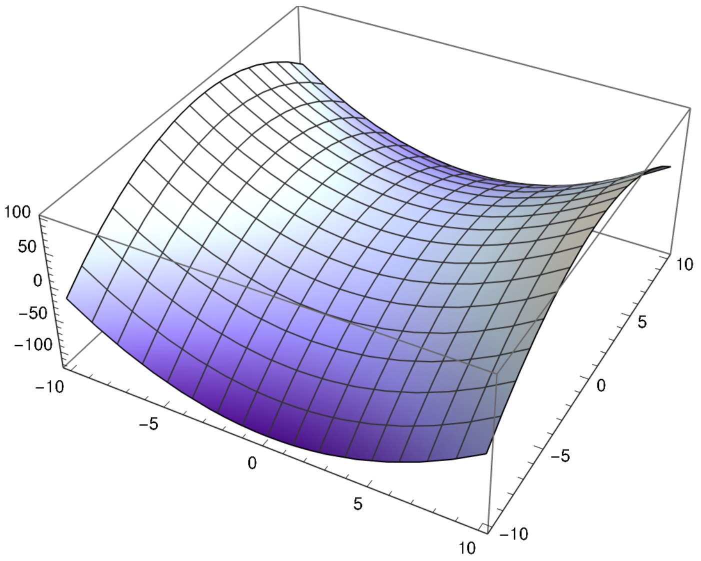
Out[◦]=



(1 b)

```
In[ ∘ ]:= Plot3D[x^2 - (y - 1)^2, {x, -10, 10}, {y, -10, 10}, ColorFunction → "LakeColors"]
```

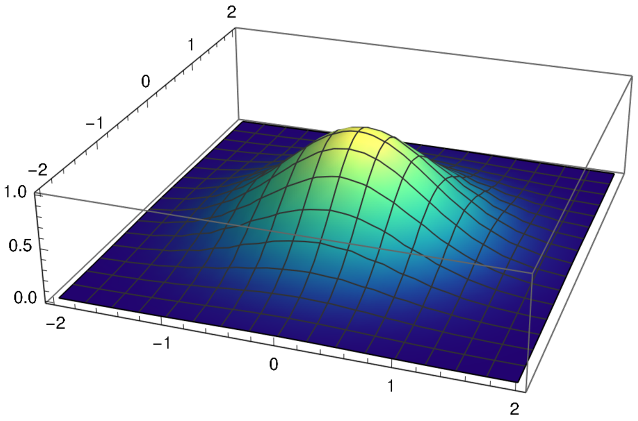
Out[∘]=



(1 c)

```
In[ ∘ ]:= Plot3D[Exp[-x^2 - y^2], {x, -2, 2}, {y, -2, 2},
           BoxRatios → Automatic, ColorFunction → "BlueGreenYellow"]
```

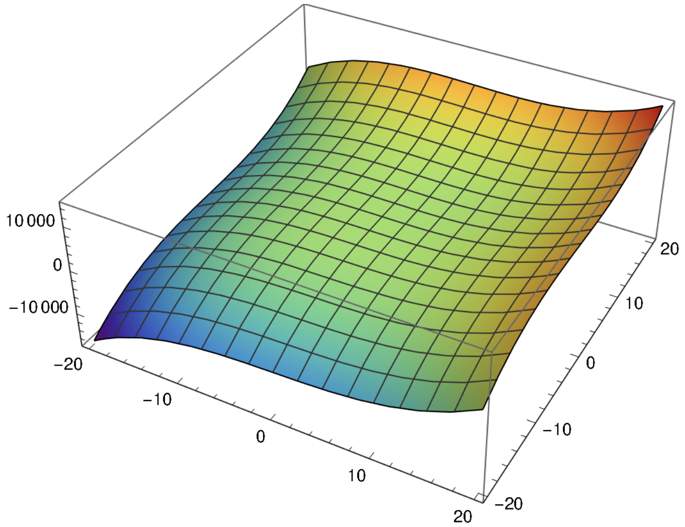
Out[∘]=



(1 d)

```
In[ ∘ ]:= Plot3D[x ^ 3 + y ^ 3 - 3 x * y, {x, -20, 20}, {y, -20, 20}, ColorFunction → "Rainbow"]
```

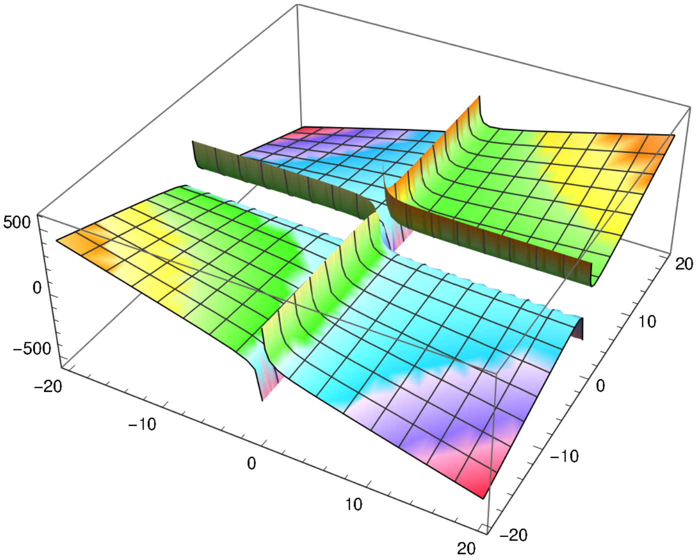
Out[∘]=



(e)

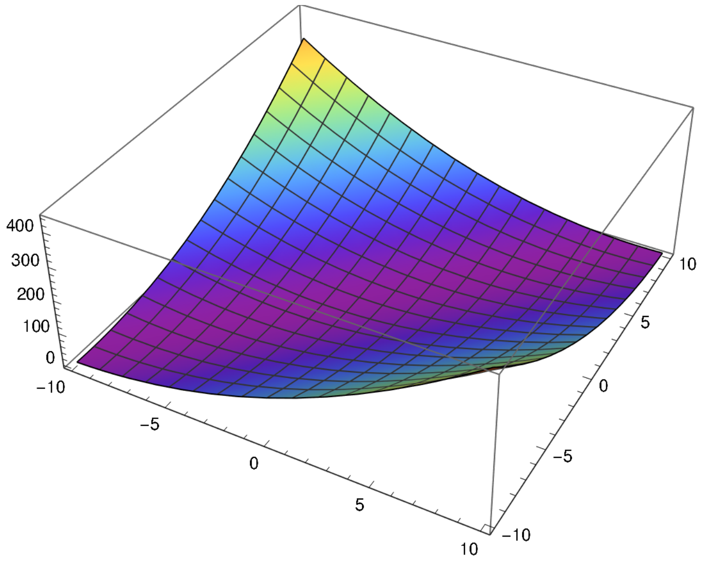
```
In[ ∘ ]:= Plot3D[x * y + 50 / x + 50 / y, {x, -20, 20}, {y, -20, 20}, ColorFunction → "BrightBands"]
```

Out[∘]=



(f)

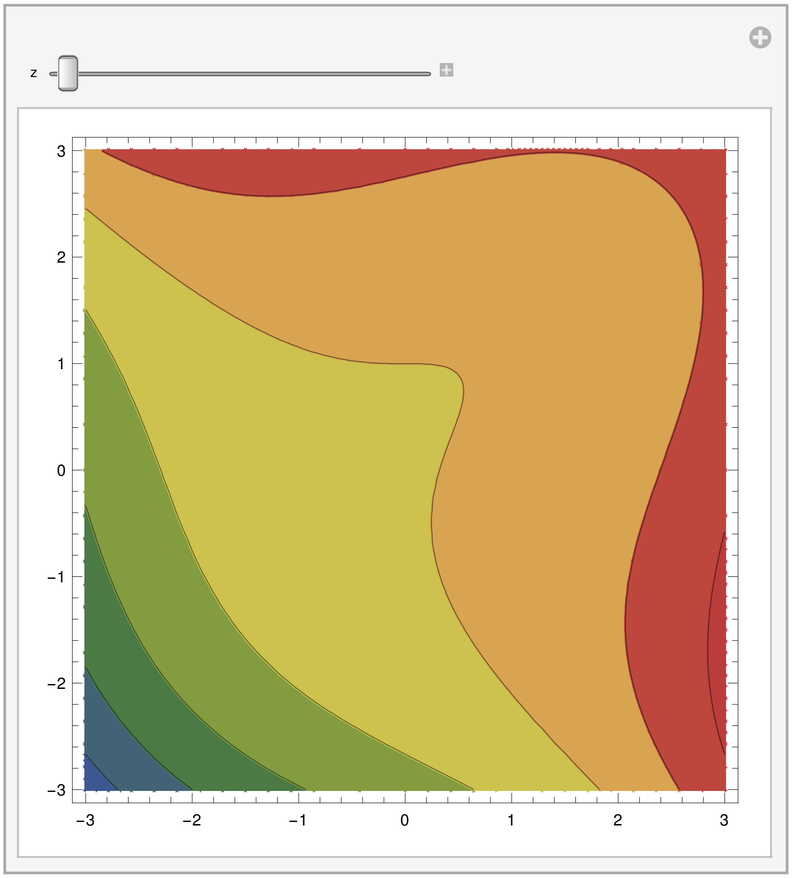
```
Manipulate[ContourPlot[x^2 + y^3 + z^4,
  {x, -1, 1}, {y, -1, 1}, ColorFunction → "DarkRainbow"], {z, -1, 1}]
```



(g)

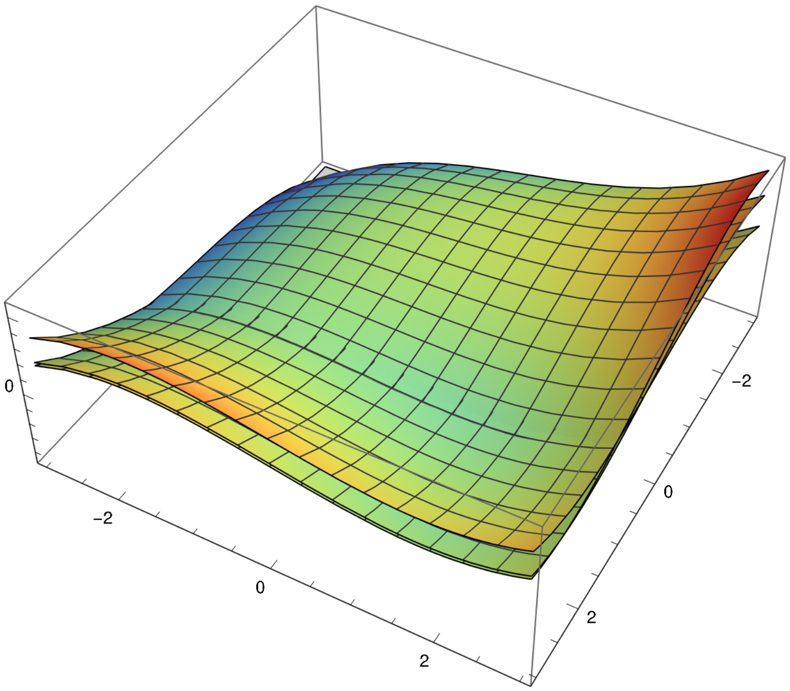
```
In[ ∘ ]:= Manipulate[ContourPlot[x^3 + y^3 + z^3 - 3 (x * y + x * z),
   {x, -3, 3}, {y, -3, 3}, ColorFunction → "DarkRainbow"], {z, -1, 1}]
```

Out[∘]=



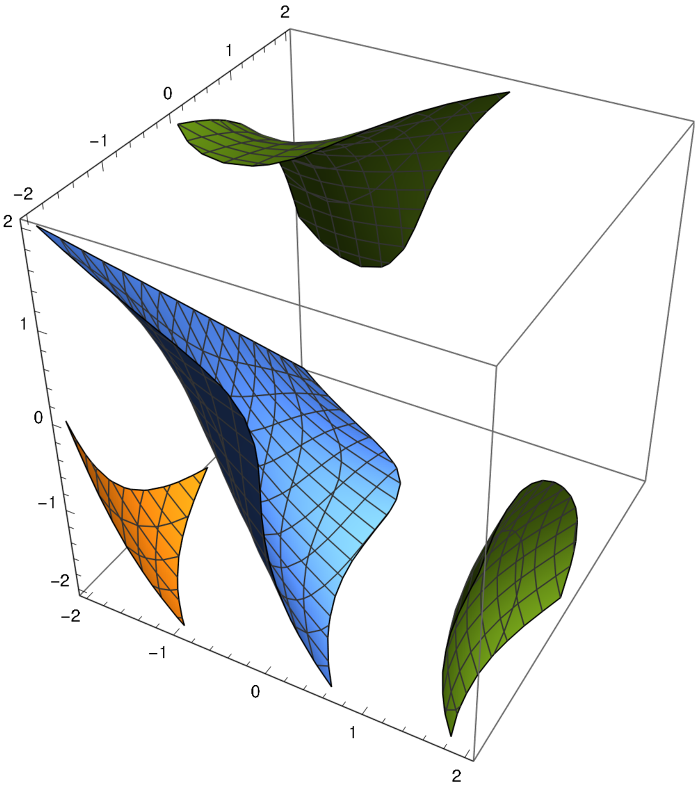
```
In[ • ]:= Plot3D[Evaluate @Table[x ^ 3 + y ^ 3 + z ^ 3 - 3 (x * y + x * z), {z, {-1, 1, 2}}],
           {x, -3, 3}, {y, -3, 3}, PlotStyle → {Red, Green, Blue}]
```

Out[•]=



```
In[ • ]:= ContourPlot3D [x ^ 3 + y ^ 3 + z ^ 3 - 3 (x * y + x * z), {x, -2, 2}, {y, -2, 2}, {z, -2, 2}]
```

Out[•]=



In[1]:=

```
Outer[Function[{z, y, x},
      If[x > 0 && y > 0 && z > 0, 0, Rescale[x ^ 3 + y ^ 3 + z ^ 3 - 3 (x * y + x * z), valueInterval ]]],
    Reverse @#, Reverse @#, #] &@Range[-1, 1, .02] //
 Image3D[#, ColorFunction → "RainbowOpacity ", Boxed → True, Axes → True,
   AxesLabel → (ToBoxes[Style[#, 20]] & /@ {x, y, z})] &
```

```
••• Rescale : The argument valueInterval at position 2 is expected to be a list of a lower bound and an upper bound .

••• Rescale : The argument valueInterval at position 2 is expected to be a list of a lower bound and an upper bound .

••• Rescale : The argument valueInterval at position 2 is expected to be a list of a lower bound and an upper bound .

••• General : Further output of Rescale ::rtwo will be suppressed during this calculation .

••• Image3D : The specified argument {{{Rescale [7., valueInterval ], Rescale [6.93881 , valueInterval ],
    Rescale [6.87526 , valueInterval ], Rescale [6.80942 , valueInterval ], Rescale [6.74131 , valueInterval ],
    Rescale [6.671 , valueInterval ], Rescale [6.59853 , valueInterval ], Rescale [<<2>>], <<89>>, 0, 0, 0, 0},
    <<100 >>}, <<100 >>} should be an array of rank 3 or 4 with machine -sized numbers or a list of images of
    consistent dimension and color space .
```

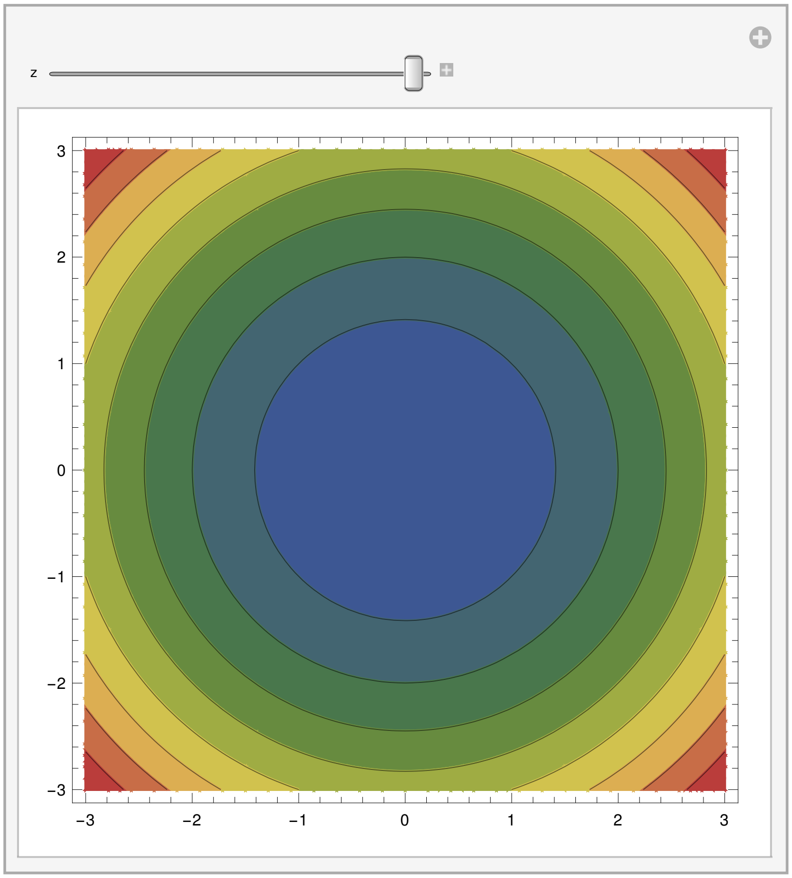
Out[1]=

```
Image3D[{{ ··· 1 ··· }, ··· 100 ··· }, ColorFunction → RainbowOpacity , Boxed → True,
 Axes → True, AxesLabel → {StyleBox[x, 20, StripOnInput → False],
   StyleBox[y, 20, StripOnInput → False], StyleBox[z, 20, StripOnInput → False]}]

large output | show less | show more | show all | set size limit...
```

```
(*Ukazka*)
```

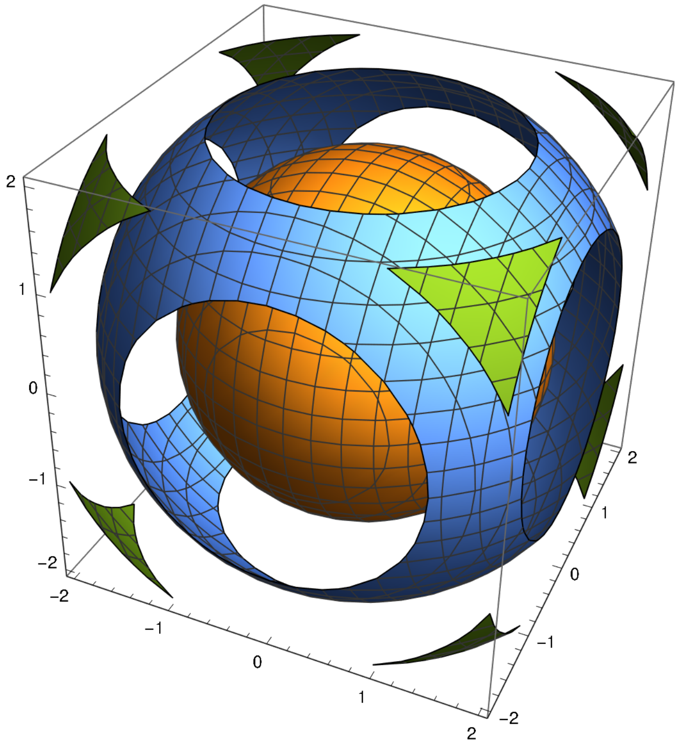
```
In[ ∘ ]:= Manipulate[ContourPlot[x^2 + y^2 + z^2,
  {x, -3, 3}, {y, -3, 3}, ColorFunction → "DarkRainbow"], {z, -2, 2}]
```

Out[∘]=



```
In[ ]:= ContourPlot3D[x^2 + y^2 + z^2, {x, -2, 2}, {y, -2, 2}, {z, -2, 2}]
```

Out[ ]=

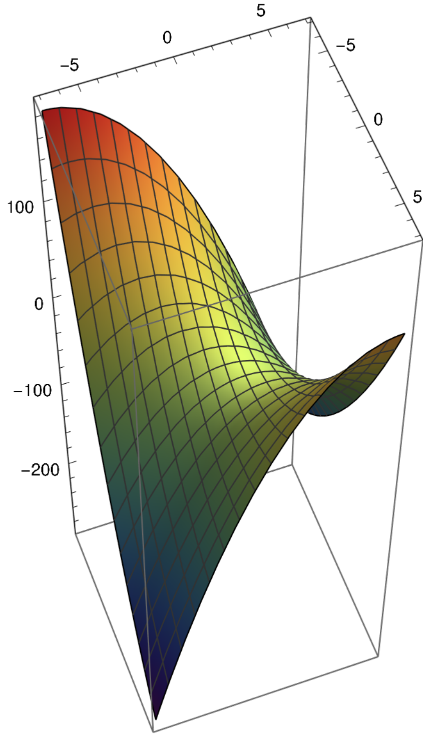


```
In[ ]:= Outer[Function[{z, y, x},
           If[x > 0 && y > 0 && z > 0, 0, Rescale[x^3 + y^3 + z^3 - 3 (x * y + x * z), valueInterval]]],
          Reverse@#, Reverse@#, #] &@Range[-1, 1, 0.2] //
       Image3D[#, ColorFunction → "RainbowOpacity", Boxed → True, Axes → True,
         AxesLabel → (ToBoxes[Style[#, 20]] & /@ {x, y, z})] &
```

```
••• Rescale : The argument valueInterval at position 2 is expected to be a list of a lower bound and an upper bound .

••• Rescale : The argument valueInterval at position 2 is expected to be a list of a lower bound and an upper bound .

••• Rescale : The argument valueInterval at position 2 is expected to be a list of a lower bound and an upper bound .

••• General : Further output of Rescale ::rtwo will be suppressed during this calculation .

••• Image3D : The specified argument
      {{{Rescale [7., valueInterval ], Rescale [6.288 , valueInterval ], Rescale [5.384 , valueInterval ], Rescale [4.336 ,
            valueInterval ], Rescale [3.192 , valueInterval ], Rescale [2., valueInterval ], 0, 0, 0, 0, 0}, ≪9≫, {Rescale [−1.,
            0.512 , valueInterval ], Rescale [1., valueInterval ]}}, ≪9≫, {≪1≫}} should be an
      array of rank 3 or 4 with machine −sized numbers or a list of images of consistent dimension and color
      space .
```

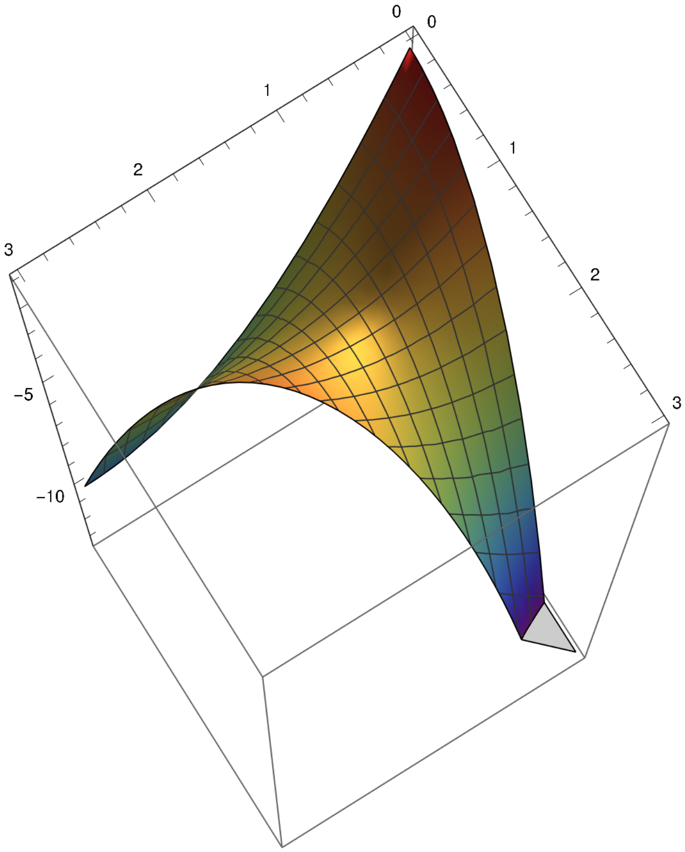
```
In[ ∘ ]:= Plot3D[x ^ 2 - 2 y ^ 2 + 4 x * y - 6 x - 1, {x, -7, 7},
  {y, -7, 7}, ColorFunction → "Rainbow", BoxRatios → {1, 1, 3}]
```

Out[∘]=



```
In[ ∘ ]:= Plot3D[{x ^ 2 - 2 y ^ 2 + 4 x * y - 6 x - 1}, {x, 0, 4},
            {y, 0, 4}, RegionFunction → Function[{x, y, z}, y < 3 - x],
            BoxRatios → Automatic, ColorFunction → "Rainbow"]
```

Out[∘]=


0
0
1
2
3
1
2
3
-5
-10