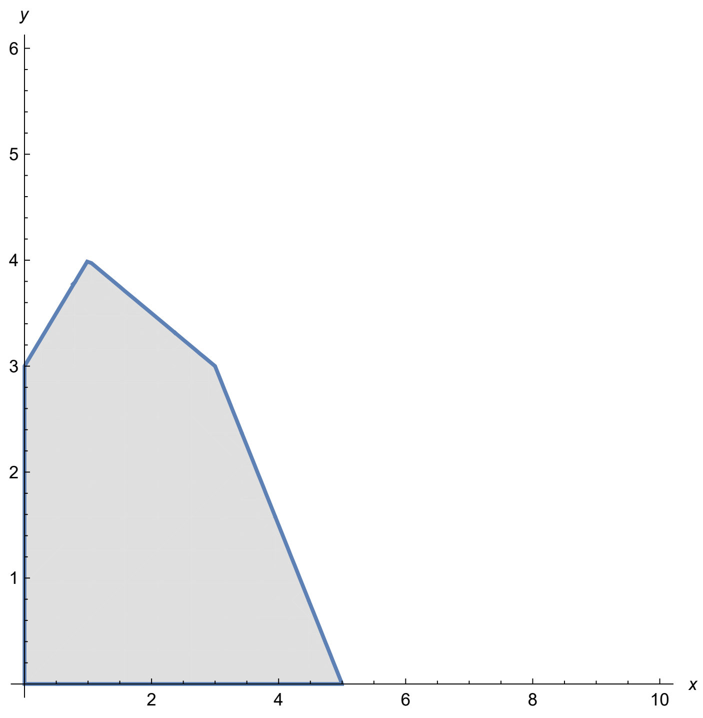
```
In[]:= (*
Задача 1. Решете задачата на линейното програмиране
  F(x,y)=3y-x→max(min)
    Ω е областта съставена от
  |-x+y-3≤0
  |-x-2y+9≥0
  |3x+2y-15≤0
  |x≥0
  |y≥0

*)

(* 1. Скицирайте областта Ω. *)

p = RegionPlot[
  -x + y - 3 ≤ 0 &&
   -x - 2 * y + 9 ≥ 0 &&
   3 * x + 2 * y - 15 ≤ 0 &&
   x ≥ 0 &&
   y ≥ 0,
  {x, 0, 10},
  {y, 0, 6},
  Axes → True,
  Frame → False,
  AxesLabel → {x, y},
  PlotStyle → {Directive[Gray, Opacity[0.25]]}
 ]
```

Out[]=



In[ ]:=
```
(* Множеството на всички производствени планове е петоъгълник *)
```

In[ ]:=
```
(* 2. Намерете върховете на Ω и ги скицирайте. *)

(*
    Трябва да намерим екстремалните точки.
    За целта трябва да изчислим координатите на върховете.
*)

(* Определяме първия връх *)
Reduce[-x + y - 3 == 0 && x == 0, {x, y}]
```

Out[ ]=
```
x == 0 && y == 3
```

In[ ]:=

In[ ]:=
```
a1 = {0, 3}
```

Out[ ]=
```
{0, 3}
```

In[ ]:=
```
(* Определяме втория връх *)
```

In[ ]:=
```
Reduce[-x + y - 3 == 0 && -x - 2 * y + 9 == 0, {x, y}]
```

Out[ ]=
```
x == 1 && y == 4
```

In[ ]:=
```
a2 = {1, 4}
```

Out[ ]=
```
{1, 4}
```

In[ ]:=
```
(* Определяме третия връх *)
```

In[ ]:=
```
Reduce[-x - 2 * y + 9 == 0 && 3 * x + 2 * y - 15 == 0, {x, y}]
```

Out[ ]=
```
x == 3 && y == 3
```

In[ ]:=
```
a3 = {3, 3}
```

Out[ ]=
```
{3, 3}
```

In[ ]:=
```
(* Определяме четвъртия връх *)
```

In[ ]:=
```
Reduce[y == 0 && 3 * x + 2 * y - 15 == 0, {x, y}]
```

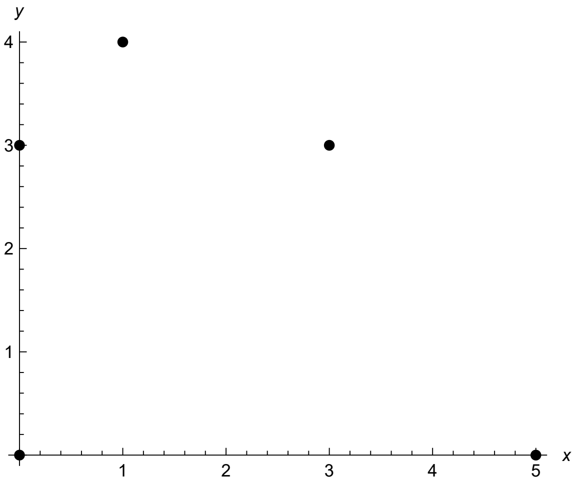
Out[ ]=
```
x == 5 && y == 0
```

In[ ]:=
```
a4 = {5, 0}
```

Out[ ]=
```
{5, 0}
```

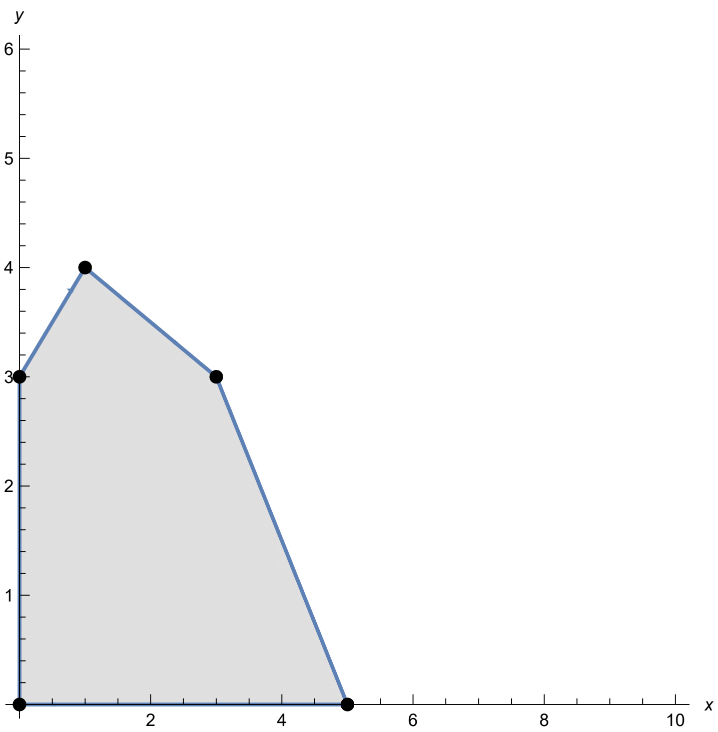
In[ ]:=
```
(* Скицираме върховете на Ω *)
t = Graphics[
  {Black, PointSize[.02], Point[{{0, 0}, a1, a2, a3, a4}]},
  AxesOrigin → {0, 0}, Axes → True, AxesLabel → {x, y}
 ]
```

Out[ ]=



In[ ]:=
```
r1 = Show[p, t]
```

Out[ ]=



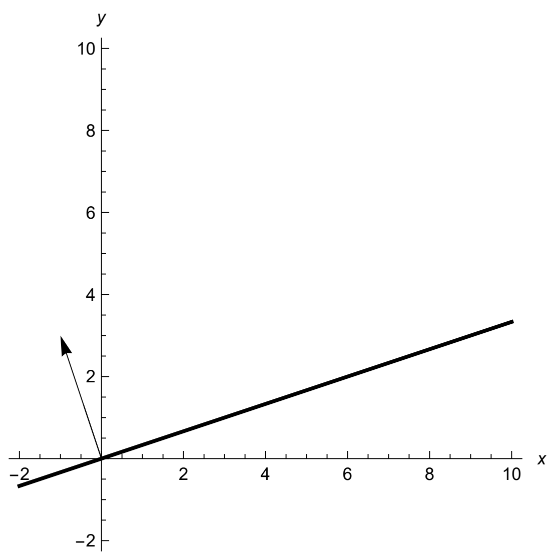
In[ ]:=
```
(* Функцията от условието: *)
F[x_, y_] = -x + 3 y
```

Out[ ]=

$-x + 3\,y$

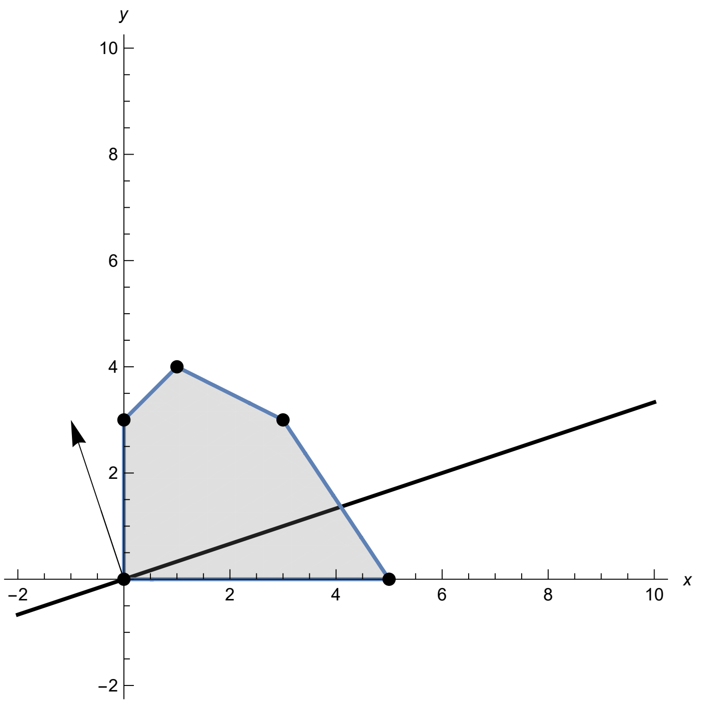
In[ ]:=
```
(*
3. За функцията F(x1,x2) скицирайте производствените планове {x1,x2},
при които F(x1,x2)=0.
  На същия чертеж скицирайте векторът,
показващ направлението в което нараства F(x1,x2).
*)

g0 = ContourPlot[
  F[x, y] == 0,
  {x, -2, 10},
  {y, -2, 10},
  Axes → True,
  Frame → False,
  AxesLabel → {x, y},
  ContourStyle → {Directive[Black]},
  AspectRatio → Automatic,
  Epilog → {Arrow[{{0, 0}, {-1, 3}}]}
 ]
```

Out[ ]=



In[∘]:=
```
Show[g0, r1]
```

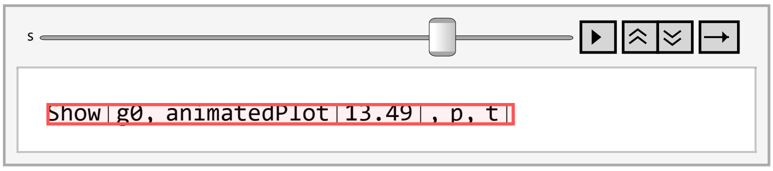
Out[∘]=



In[∘]:=
```
(* 4.Направете анимация за откриване на оптималните решения. *)
animatedPlot[s_] := ContourPlot[
  F[x, y] == s,
  {x, -2, 10},
  {y, -2, 10},
  Axes → True,
  Frame → False,
  ContourStyle → {Directive[Black]},
  AxesLabel → {x, y},
  AspectRatio → Automatic
 ]
```

In[∘]:=
```
Animate[Show[g0, animatedPlot[s], p, t], {s, -10, 20}, AnimationRunning → False]
```

Out[∘]=



In[∘]:=
```
(* Тоест минимумът и максимумът се получават върху поне един от върховете *)
```

In[∘]:=
```
{a1, a2, a3, a4}
```

Out[∘]=
```
{{0, 3}, {1, 4}, {3, 3}, {5, 0}}
```

In[∘]:=
```
allPoints = {{0, 0}, a1, a2, a3, a4}
```

Out[∘]=
```
{{0, 0}, {0, 3}, {1, 4}, {3, 3}, {5, 0}}
```

```
In[•]:= M = {
          F[allPoints[[1, 1]], allPoints[[1, 2]]],
          F[allPoints[[2, 1]], allPoints[[2, 2]]],
          F[allPoints[[3, 1]], allPoints[[3, 2]]],
          F[allPoints[[4, 1]], allPoints[[4, 2]]],
          F[allPoints[[5, 1]], allPoints[[5, 2]]]
         }
Out[•]=
        {0, 9, 11, 6, -5}

In[•]:= ourMax = Max[M]
        (* Максът е в точката {1,4}*)

Out[•]=
        11

In[•]:= ourMin = Min[M]
        (* Минимумът е в точката {5,0}*)
Out[•]=
        -5

In[•]:= (* 6. Решете задачата с функцията LinearProgramming[...]. *)

In[•]:= F[x_, y_]
Out[•]=
        -x_ + 3 y_

In[•]:= (* Макс и мин *)

In[•]:= c = {-1, 3}
Out[•]=
        {-1, 3}
```

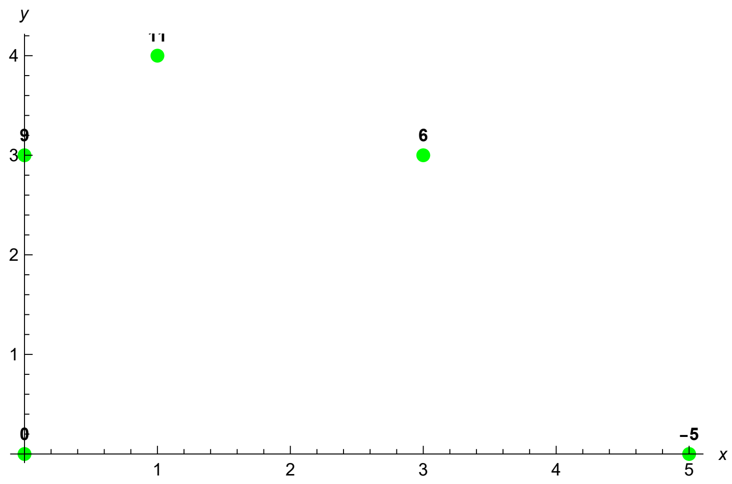
In[•]:= m = $\begin{pmatrix} -1 & 1 \\ 1 & 2 \\ 3 & 2 \end{pmatrix}$

```
Out[•]=
        {{-1, 1}, {1, 2}, {3, 2}}
```

In[•]:= b = $\begin{pmatrix} 3 & -1 \\ 9 & -1 \\ 15 & -1 \end{pmatrix}$

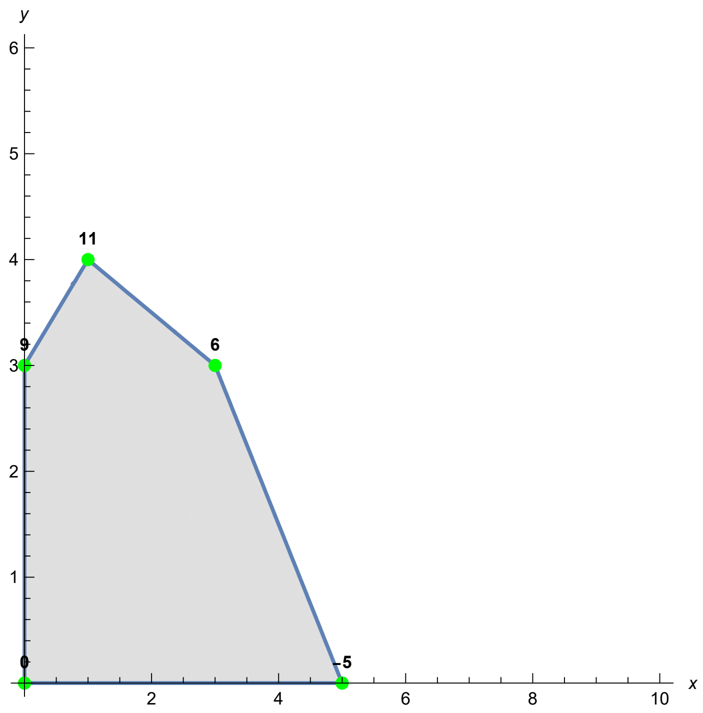
```
Out[•]=
        {{3, -1}, {9, -1}, {15, -1}}

In[•]:= v = LinearProgramming[c, m, b]
Out[•]=
        {5, 0}

In[•]:= (* Планът с x=5 и y=0 е минимум и минималната печалба на F[x,y] е *)

In[•]:= F[5, 0]
Out[•]=
        -5
```

```
In[∘]:= (* Понеже LinearProgramming е само за минимизация,
        то трябва да обърнем целевата функция *)

In[∘]:= cReversed = {1, -3}
Out[∘]=
        {1, -3}

In[∘]:= vv = LinearProgramming[cReversed, m, b]
Out[∘]=
        {1, 4}

In[∘]:= (* => максималната стойност на функцията е: *)

In[∘]:= maxValue = -cReversed.vv
Out[∘]=
        11

In[∘]:= (* доказателство *)

In[∘]:= F[1, 4]
Out[∘]=
        11

In[∘]:= (* 7. Намерете всички решения и ги скицирайте. *)
        (* Решенията на целевата функция ги имаме във вектора M *)
        M
Out[∘]=
        {0, 9, 11, 6, -5}

In[∘]:= lp = ListPlot[
          allPoints,
          PlotStyle → Directive[PointSize[.02], Green],
          AxesLabel → {x, y},
          Epilog → {Table[Text[Style[M[[i]], Black, Bold], allPoints[[i]] + {0.0, 0.2}],
             {i, 1, Length[allPoints]}]}]
Out[∘]=
```



In[∘]:=
```
Show[p, lp, Epilog → {Table[Text[Style[M⟦i⟧, Black, Bold], allPoints⟦i⟧ + {0.0, 0.2}],
    {i, 1, Length[allPoints]}]}]
```

Out[∘]=



In[∘]:=
```
(* 8. Решение на задачата с функциите Reduce[...] и Maximize[...]. *)
F[x_, y_] = -x + 3 * y
```

Out[∘]=

$-x + 3y$

In[∘]:=
```
Reduce[3 y - x == 11 && -x + y - 3 ≤ 0 && -x - 2 y + 9 ≥ 0 && 3 x + 2 y - 15 ≤ 0 && x ≥ 0 && y ≥ 0, {x, y}]
```

Out[∘]=

x == 1 && y == 4

In[∘]:=
```
x == 1 && y == 4
(* оптимален план *)
```

Out[∘]=

x == 1 && y == 4

In[∘]:=
```
Maximize[{F[x, y], -x + y - 3 ≤ 0 && -x - 2 y + 9 ≥ 0 && 3 x + 2 y - 15 ≤ 0 && x ≥ 0 && y ≥ 0}, {x, y}]
```

Out[∘]=

{11, {x → 1, y → 4}}