In [6]:
```
#primal
import numpy as np
from scipy.optimize import linprog
primal_c = [-3, -5]
primal_A = [[1, 1],
            [1, 3]]
primal_b = [6, 9]
bounds = [(0, None), (0, None)]
primal = linprog(primal_c,
                 A_ub=primal_A,
                 b_ub=primal_b,
                 bounds=bounds,
                 method="highs")
print(primal)
```

```
       message: Optimization terminated successfully. (HiGHS Status 7: Optimal)
       success: True
        status: 0
           fun: -21.0
             x: [ 4.500e+00  1.500e+00]
           nit: 2
         lower:  residual: [ 4.500e+00  1.500e+00]
                marginals: [ 0.000e+00  0.000e+00]
         upper:  residual: [       inf        inf]
                marginals: [ 0.000e+00  0.000e+00]
         eqlin:  residual: []
                marginals: []
       ineqlin:  residual: [ 0.000e+00  0.000e+00]
                marginals: [-2.000e+00 -1.000e+00]
mip_node_count: 0
mip_dual_bound: 0.0
       mip_gap: 0.0
```

In [4]:
```
print(primal.x)
```

```
[4.5 1.5]
```

In [5]:
```
print(-primal.fun)
```

```
21.0
```

In [9]:
```
print(primal.ineqlin.marginals)
```

```
[-2. -1.]
```

In [8]:
```
#dual
import numpy as np
from scipy.optimize import linprog
dual_c = [6, 9]
dual_A = [[-1, -1],
          [-1, -3]]
dual_b = [-3, -5]
bounds = [(0, None), (0, None)]
dual = linprog(dual_c,
                 A_ub=dual_A,
                 b_ub=dual_b,
                 bounds=bounds,
                 method="highs")
print(dual)
```

```
           message: Optimization terminated successfully. (HiGHS Status 7: Optimal)
           success: True
            status: 0
               fun: 21.0
                 x: [ 2.000e+00  1.000e+00]
               nit: 2
             lower:  residual: [ 2.000e+00  1.000e+00]
                    marginals: [ 0.000e+00  0.000e+00]
             upper:  residual: [       inf        inf]
                    marginals: [ 0.000e+00  0.000e+00]
             eqlin:  residual: []
                    marginals: []
           ineqlin:  residual: [ 0.000e+00  0.000e+00]
                    marginals: [-4.500e+00 -1.500e+00]
    mip_node_count: 0
    mip_dual_bound: 0.0
           mip_gap: 0.0
```

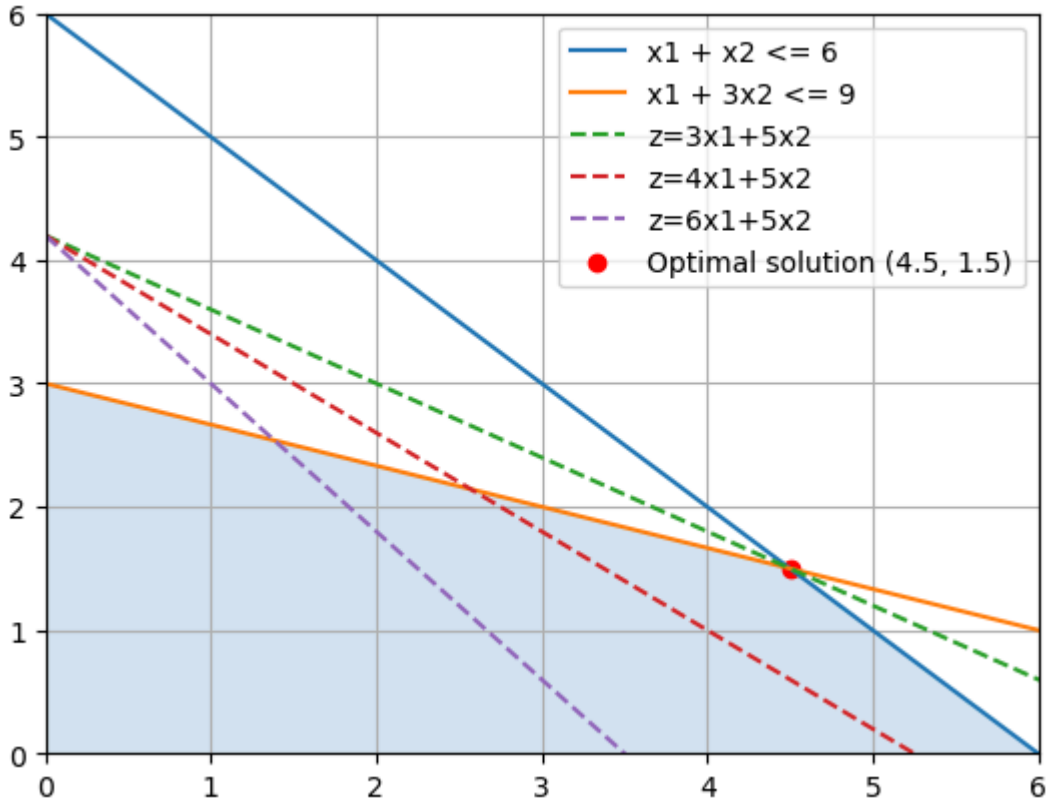
In [10]:
```
print(dual.ineqlin.marginals)
```

```
[-4.5 -1.5]
```

In [17]:
```
# shadow primal
import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(0, 7, 400)
constraint_1 = 6 - x
constraint_2 = (9 - x) / 3
z_original=(21-3*x)/5
z_small=(21-4*x)/5
z_large=(21-6*x)/5
plt.figure()
plt.plot(x, constraint_1, label="x1 + x2 <= 6")
plt.plot(x, constraint_2, label="x1 + 3x2 <= 9")
plt.plot(x,z_original,"--",label="z=3x1+5x2")
plt.plot(x,z_small,"--",label="z=4x1+5x2")
plt.plot(x,z_large,"--",label="z=6x1+5x2")
plt.fill_between(x, 0, np.minimum(constraint_1, constraint_2), alpha=0.2)
plt.scatter(primal.x[0], primal.x[1], color='red', label="Optimal solution (4.5,

plt.xlim(0, 6)
plt.ylim(0, 6)

plt.legend()
plt.grid()
plt.show()
```



In [5]:
```
import numpy as np
from scipy.optimize import linprog
def primal_dual(C, A, b):
    A = np.array(A)
    b = np.array(b)
    c = np.array(C)

    primal = linprog(
        c=-c,
        A_ub=A,
        b_ub=b,
        bounds=[(0, None)] * len(c),
        method='highs'
    )

    dual = linprog(
        c=b,
        A_ub=-A.T,
        b_ub=-c,
        bounds=[(0, None)] * len(b),
        method='highs'
    )

    print(primal.x)
    print(primal.fun)
    print(dual.x)
C = [3, 5]
A = [[1, 1], [1, 3]]
B = [6, 9]
primal_dual(C, A, B)
```

```
[4.5 1.5]
-21.0
[2. 1.]
```