```python
# Wariant 6 (plik war6.csv)
import numpy as np
import pandas as pd

data = pd.read_csv("war6.csv", delimiter=";", decimal=",", header=0)

print(data)
```

```
      x1   x2            y
0      1    2     4.361171
1      3    6    32.029778
2      5   10    63.867159
3      7   14    71.202104
4      9   18    16.762118
..   ...  ...          ...
105  211  422  1706.335214
106  213  426  3341.097267
107  215  430  2668.042391
108  217  434  3411.210960
109  219  438  2291.258104

[110 rows x 3 columns]
```

```python
x1 = data["x1"].values.astype(float)
x2 = data["x2"].values.astype(float)
y = data["y"].values

# Tworzenie macierzy A do SVD
A = np.column_stack((x1, x2))

U,S,VT = np.linalg.svd(A, full_matrices=False)

print("-----U-----")
print(U)
print("-----VT-----")
print(VT)
```

```
-----U-----
[[-7.50664867e-04  9.38761926e-01]
 [-2.25199460e-03  1.29184778e-03]
 [-3.75332434e-03 -1.41264526e-03]
 [-5.25465407e-03  1.23144904e-03]
 [-6.75598381e-03 -1.47304401e-03]
 [-8.25731354e-03  1.17105030e-03]
 [-9.75864328e-03 -1.53344275e-03]
 [-1.12599730e-02 -4.23793579e-03]
 [-1.27613027e-02  3.75474587e-03]
 [-1.42626325e-02  1.05025282e-03]
 [-1.57639622e-02 -1.65424023e-03]
 [-1.72652920e-02  6.33844144e-03]
 [-1.87666217e-02 -7.06322632e-03]
```

```
[-2.02679514e-02  9.29455340e-04]
[-2.17692812e-02  8.92213700e-03]
[-2.32706109e-02 -4.47953076e-03]
[-2.47719406e-02  3.51315091e-03]
[-2.62732704e-02 -9.88851685e-03]
[-2.77746001e-02 -1.89583519e-03]
[-2.92759298e-02  6.09684647e-03]
[-3.07772596e-02 -7.30482129e-03]
[-3.22785893e-02  6.87860377e-04]
[-3.37799190e-02  8.68054204e-03]
[-3.52812488e-02  1.66732237e-02]
[-3.67825785e-02 -1.81227935e-02]
[-3.82839082e-02 -1.01301118e-02]
[-3.97852380e-02 -2.13743015e-03]
[-4.12865677e-02  5.85525151e-03]
[-4.27878974e-02  1.38479332e-02]
[-4.42892272e-02 -2.09480840e-02]
[-4.57905569e-02 -1.29554023e-02]
[-4.72918866e-02 -4.96272068e-03]
[-4.87932164e-02  3.02996098e-03]
[-5.02945461e-02  1.10226426e-02]
[-5.17958759e-02  1.90153243e-02]
[-5.32972056e-02 -1.57806929e-02]
[-5.47985353e-02 -7.78801121e-03]
[-5.62998651e-02  2.04670453e-04]
[-5.78011948e-02  8.19735212e-03]
[-5.93025245e-02  1.61900338e-02]
[-6.08038543e-02 -1.86059834e-02]
[-6.23051840e-02 -1.06133017e-02]
[-6.38065137e-02 -2.62062008e-03]
[-6.53078435e-02  5.37206159e-03]
[-6.68091732e-02  1.33647433e-02]
[-6.83105029e-02  2.13574249e-02]
[-6.98118327e-02  2.93501066e-02]
[-7.13131624e-02  3.73427882e-02]
[-7.28144921e-02 -4.02419278e-02]
[-7.43158219e-02 -3.22492461e-02]
[-7.58171516e-02 -2.42565645e-02]
[-7.73184813e-02 -1.62638828e-02]
[-7.88198111e-02 -8.27120114e-03]
[-8.03211408e-02 -2.78519472e-04]
[-8.18224705e-02  7.71416219e-03]
[-8.33238003e-02  1.57068439e-02]
[-8.48251300e-02  2.36995255e-02]
[-8.63264598e-02  3.16922072e-02]
[-8.78277895e-02  3.96848888e-02]
[-8.93291192e-02 -3.78998272e-02]
[-9.08304490e-02 -2.99071455e-02]
[-9.23317787e-02 -2.19144639e-02]
```

```
 [-9.38331084e-02 -1.39217822e-02]
 [-9.53344382e-02 -5.92910053e-03]
 [-9.68357679e-02  2.06358113e-03]
 [-9.83370976e-02  1.00562628e-02]
 [-9.98384274e-02  1.80489445e-02]
 [-1.01339757e-01  2.60416261e-02]
 [-1.02841087e-01  3.40343078e-02]
 [-1.04342417e-01 -4.35504082e-02]
 [-1.05843746e-01 -3.55577266e-02]
 [-1.07345076e-01 -2.75650449e-02]
 [-1.08846406e-01 -1.95723633e-02]
 [-1.10347736e-01 -1.15796816e-02]
 [-1.11849065e-01 -3.58699993e-03]
 [-1.13350395e-01  4.40568174e-03]
 [-1.14851725e-01  1.23983634e-02]
 [-1.16353054e-01  2.03910451e-02]
 [-1.17854384e-01  2.83837267e-02]
 [-1.19355714e-01  3.63764084e-02]
 [-1.20857044e-01 -4.12083076e-02]
 [-1.22358373e-01 -3.32156260e-02]
 [-1.23859703e-01 -2.52229443e-02]
 [-1.25361033e-01 -1.72302627e-02]
 [-1.26862363e-01 -9.23758099e-03]
 [-1.28363692e-01 -1.24489932e-03]
 [-1.29865022e-01  6.74778234e-03]
 [-1.31366352e-01  1.47404640e-02]
 [-1.32867682e-01  2.27331457e-02]
 [-1.34369011e-01  3.07258273e-02]
 [-1.35870341e-01  3.87185090e-02]
 [-1.37371671e-01  4.67111907e-02]
 [-1.38873000e-01  5.47038723e-02]
 [-1.40374330e-01  6.26965540e-02]
 [-1.41875660e-01  7.06892357e-02]
 [-1.43376990e-01  7.86819173e-02]
 [-1.44878319e-01 -1.27025215e-01]
 [-1.46379649e-01 -1.25616881e-01]
 [-1.47880979e-01 -1.24208547e-01]
 [-1.49382309e-01 -1.22800213e-01]
 [-1.50883638e-01  1.00728444e-02]
 [-1.52384968e-01  1.14811784e-02]
 [-1.53886298e-01  1.28895124e-02]
 [-1.55387628e-01  1.42978464e-02]
 [-1.56888957e-01  1.57061804e-02]
 [-1.58390287e-01  1.71145144e-02]
 [-1.59891617e-01  1.85228484e-02]
 [-1.61392946e-01  1.99311824e-02]
 [-1.62894276e-01  2.13395164e-02]
 [-1.64395606e-01  2.27478505e-02]]
-----VT-----
```

```
[[-0.4472136  -0.89442719]
 [ 0.89442719 -0.4472136 ]]
```

```
# Obliczamy współczynniki a i b
# Na podstawie transponowanej macierzy VT, U, wektora y oraz odwrotności macierzy S
# (zgodnie z instrukcją)
a,b = VT.T @ np.linalg.inv(np.diag(S)) @ U.T @ y

print("Obliczone współczynniki: a = " + str(a) + ", b = " + str(b))
```

```
Obliczone współczynniki: a = -2350160115972010.0, b = 1175080057986011.0
```