**Практика 1. Подбор параметров схемы 1 с помощью генетического алгоритма.**

На рисунке 1 представлена электрическая схема. Известны входной ток и напряжение (таблица 1). Необходимо найти параметры резисторов () и диода ().

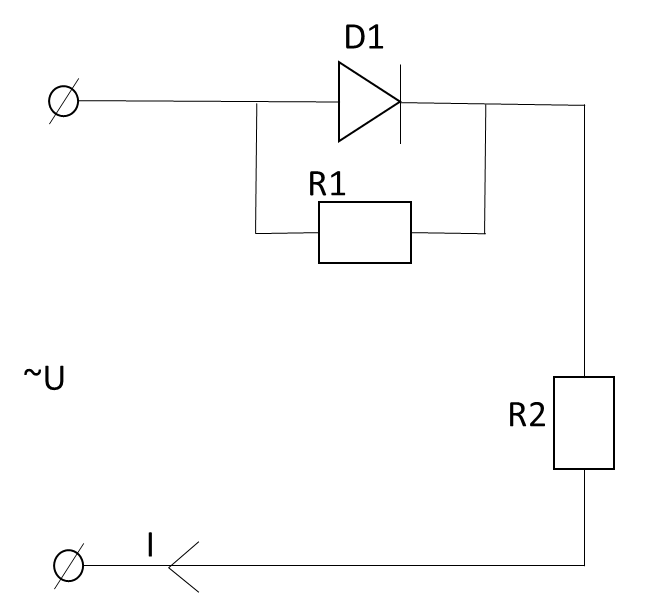


Рисунок 1 – Электрическая схема

Таблица 1 – Исходные данные

|  |  |
| --- | --- |
|  |  |
| 0 | 0 |
| 0,136 | 0,6 |
| 0,324 | 1,449 |
| 0,591 | 2,67 |
| 0,9735 | 4,445 |

Ток через диод вычисляется по формуле (1):

**def** population(N\_person):  
 *'''Create population'''* df\_population= pd.DataFrame([])  
 **for** i **in** range(0, N\_person):  
 I0 = round(1 + random.random() \* 5, 3)  
 R1 = round(1 + random.random() \* 10, 3)  
 R2 = round(1 + random.random() \* 10, 3)  
  
 df\_population.loc[i, **'I0'**] = I0  
 df\_population.loc[i, **'R1'**] = R1  
 df\_population.loc[i, **'R2'**] = R2  
 **return** df\_population

**def** choose\_2\_person(df\_population):  
 P1,P2 = 0,0  
  
 **while** P1 == P2:  
 P1 = int(random.choice(np.linspace(0,len(df\_population)-1,len(df\_population))))  
 P2 = int(random.choice(np.linspace(0,len(df\_population)-1,len(df\_population))))  
  
 **return** P1,P2  
  
  
**def** diviation(df,string):  
 df\_initial = pd.DataFrame({**'U'**:[0,0.136,0.324,0.591,0.9735],  
 **'I'**:[0,0.6,1.449,2.67,4.445]})  
  
 I0 = df[**'I0'**].iloc[string]  
 R1 = df[**'R1'**].iloc[string]  
 R2 = df[**'R2'**].iloc[string]  
  
 df\_task = pd.DataFrame([])  
  
 *#TASK* **for** i **in** range(0,5):  
 Ud = i  
 Id = I0 \* (np.exp(1 \* Ud) - 1)  
 I = Id + Ud / R1  
 U = Ud + I \* R2  
  
 df\_task.loc[i, **'U'**] = round(U, 3)  
 df\_task.loc[i, **'I'**] = round(I, 3)  
  
 Error = 0  
 **for** i **in** range(len(df\_task)):  
 Error = Error + abs(df\_initial.loc[i, **'U'**] - df\_task.loc[i, **'U'**])/(df\_initial.loc[i, **'U'**]/100 +0.01) + \  
 abs(df\_initial.loc[i, **'I'**] - df\_task.loc[i, **'I'**])/(df\_initial.loc[i, **'I'**]/100 +0.01)  
  
 df.loc[string, **'error'**] = Error  
  
  
**def** crossover(df\_population,df\_new\_generation,P1,P2,string):  
  
 P1 = df\_population.loc[P1].to\_frame(name=**'P1'**)  
 P2 = df\_population.loc[P2].to\_frame(name=**'P2'**)  
  
 *# random choose childs characteristics* I0 = random.choice([P1.iloc[0][0], P2.iloc[0][0]])  
 R1 = random.choice([P1.iloc[1][0], P2.iloc[1][0]])  
 R2 = random.choice([P1.iloc[2][0], P2.iloc[2][0]])  
  
 df\_new\_generation.loc[string,**'I0'**] = I0  
 df\_new\_generation.loc[string, **'R1'**] = R1  
 df\_new\_generation.loc[string, **'R2'**] = R2  
  
  
**def** main():  
 Er = 1200  
 k, ITER = 0, 0  
 df\_new\_generation = pd.DataFrame([])  
 P = population(20)  
 print(P)  
  
 **while** Er > 1000:  
 **if** ITER >= 14:  
 print(**f'число итераций достигло = {**ITER**}'**)  
 find = P.loc[P[**'error'**] == Er]  
 print(find)  
 **return None** k +=1  
 ITER += 1  
 print(**f' \n ITERATION {**ITER**}'**)  
  
 **for** i **in** range(0, len(P)):  
 two\_parents = choose\_2\_person(P)  
 P1 = two\_parents[0]  
 P2 = two\_parents[1]  
 crossover(df\_population=P, df\_new\_generation=df\_new\_generation, P1=P1, P2=P2, string=i)  
  
 **for** i **in** range(0, len(P)):  
 diviation(df=P, string=i)  
 **for** i **in** range(0, len(df\_new\_generation)):  
 diviation(df=df\_new\_generation, string=i)  
  
 df\_next = pd.concat([P, df\_new\_generation], ignore\_index=**True**)  
 print(df\_next)  
  
 R = df\_next[df\_next[**'error'**] < 1e80]  
 P = R[R[**'I0'**] > 0]  
 P = P.reset\_index()  
 Er = P[**'error'**].min()  
  
 **if** len(P) <= 1:  
 print(**'i cant do this, try again\_\_\_\_length massive = '**, len(P))  
 **return** ()  
 **if** k == 2:  
 P = P.drop([**'index'**, **'level\_0'**], axis=1)  
 k = 0  
  
 print(P)  
 print(**f'min\_er={**Er**}'**)  
  
 find = P.loc[P[**'error'**] == Er]  
  
 print(find)  
 print(**f'Iterarions={**ITER**}'**)  
  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 main()