```
1
2   % Programm für die Berechnung der Gasgestehungskosten
3   function [costGasLevelised, errorInLoopsFlag] = ...
4       BgaPtg2CalcGasCost(handles, costChainAnnuityTotal, techSimuChainVec, paramChain)
5
6   % Outputs definieren
7   costGasLevelised = -1;
8   errorInLoopsFlag = false;
9   %return;
10
11
12  % Zuweisung technischer Berechnungen
13
14  % Volllaststunden auf PK-Ebene [h]
15  % Energiemenge SNG [kWh th]
16  % Wärmebedarf auf PK-Ebene [kWh th]
17  % Wärme Überschuss auf PK-Ebene [kWh th]
18  % Biogas Bedarf auf PK-Ebene [kWh th]
19  % BioMethan Überschuss auf PK-Ebene [kWh th]
20  % Strombedarf für übergeordnete Etechnik [kWh el]
21  % Leistung der Methanisierung wird auf Komp.Ebene zur Berechnung benötigt [kW th]
22  % chainWater4H2
23  % chainElyPers
24
25  %techChainFLH                  = techSimuChainVec(1);
26  techChainEnergySNG            = techSimuChainVec(2);
27  % techChainHeatDemand           = techSimuChainVec(3);
28  % techChainHeatTotal_real       = techSimuChainVec(4);
29  % techChainVolFlowPG            = techSimuChainVec(5);
30  % techChainEnergyBioM           = techSimuChainVec(6);
31  % techChainElecDemand           = techSimuChainVec(7);
32
33
34
35  % Gesamtannuität [ct/kWh]
36  if techChainEnergySNG == 0
37      msg = 'Variable "techChainEnergySNG" ist gleich null. Division durch null.';
38      msg = [char(paramChain.txtChainName), ': ', msg, ' Berechnungen für diese
        Prozesskette werden unterbrochen.'];
39      if startGuiFlag == true
40          BgaPtg2UpdateMessageBox(handles,'apend',msg);
41      else
42          disp(msg);
43      end
44      errorInLoopsFlag = true;     return;
45
46  else
47      % *100 für Umrechnung von 0,xx €/kWh auf xx ct/kWh
48      costGasLevelised = (costChainAnnuityTotal/techChainEnergySNG)*100;
49  end;
```