

# Claude 3

BY ANTHROPIC

JT

Erstelle eine einfache Simulation in NetLogo, mit der die Nutzung eines Dienstes simuliert wird. Bürger haben eine normalverteilte Einstellung zu dem Dienst. Innerhalb eines Jahres bestehen mehrere Möglichkeiten den Dienst zu nutzen. Ist die Nutzung des Dienstes zu schwierig, so sinkt die Einstellung, wenn der Dienst leicht zu nutzen ist, dann steigt die Einstellung.

Hier ist eine einfache NetLogo-Simulation, die die Nutzung eines Dienstes durch Bürger mit unterschiedlichen Einstellungen darstellt:

```

netlogo

breed [citizens citizen]

citizens-own [
  attitude      ; Einstellung zum Dienst
  service-uses  ; Anzahl der Nutzungen des Dienstes
]

to setup
  clear-all
  setup-citizens
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set attitude random-normal 0 1 ; Normalverteilte Einstellung
    set service-uses 0
    set shape "person"
    set color green
  ]
end

to go
  ask citizens [
    let ease random 10 ; Zufälliger Schwierigkeitsgrad der Nutzung
    ifelse ease > 7
      [ set attitude attitude + 0.1 ] ; Dienst war leicht zu nutzen
      [ set attitude attitude - 0.1 ] ; Dienst war schwierig zu nutzen
    set attitude precision attitude 2
    set service-uses service-uses + 1
    recolor
  ]
  tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  ifelse attitude > 0
    [ set color (attitude * 5 + 55) ]
    [ set color (155 - abs attitude * 5) ]
end

```

In dieser Simulation werden 100 Bürger (citizens) mit zufälligen Positionen und einer normalverteilten Einstellung zum Dienst erstellt. In jedem Zeitschritt (tick) nutzt jeder Bürger den Dienst, und die Schwierigkeit dieser Nutzung wird zufällig bestimmt. Wenn die Nutzung leicht war (Schwierigkeitsgrad > 7), steigt die Einstellung um 0,1, andernfalls sinkt sie um 0,1.

Die Farbe der Bürger repräsentiert ihre Einstellung: Je grüner, desto positiver ist ihre Einstellung. Je roter, desto negativer ist ihre Einstellung.

Um die Simulation zu starten, geben Sie **setup** und dann **go** oder benutzen Sie die Schaltflächen in der NetLogo-Oberfläche.

Hinweis: Diese Simulation ist sehr einfach gehalten und kann nach Bedarf erweitert werden, z.B. durch die Berücksichtigung weiterer Faktoren, die die Einstellung beeinflussen, oder durch die Visualisierung zusätzlicher Daten.



Claude does not have the ability to run the code it generates yet.

JT

Erstelle eine einfache Simulation in NetLogo, mit der die Nutzung eines Dienstes simuliert wird. Bürger haben eine normalverteilte Einstellung zu dem Dienst. Simuliere Tageschritte. Innerhalb eines Jahres bestehen mehrere Möglichkeiten den Dienst zu nutzen. Ist die Nutzung des Dienstes zu schwierig, so sinkt die Einstellung, wenn der Dienst leicht zu nutzen ist, dann steigt die Einstellung.

Hier ist eine einfache NetLogo-Simulation, die die Nutzung eines Dienstes durch Bürger mit unterschiedlichen Einstellungen simuliert, wobei Tagesschritte simuliert werden:

```

netlogo

breed [citizens citizen]

citizens-own [
  attitude      ; Einstellung zum Dienst
  service-uses  ; Anzahl der Nutzungen des Dienstes
]

to setup
  clear-all
  setup-citizens
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set attitude random-normal 0 1 ; Normalverteilte Einstellung
    set service-uses 0
    set shape "person"
    set color green
  ]
end

to go
  repeat 365 [
    ask citizens [
      let ease random 10 ; Zufälliger Schwierigkeitsgrad der Nutzung
      ifelse ease > 7
        [ set attitude attitude + 0.01 ] ; Dienst war leicht zu nutzen
        [ set attitude attitude - 0.01 ] ; Dienst war schwierig zu
nutzen
      set attitude precision attitude 2
      set service-uses service-uses + 1
      recolor
    ]
    tick
  ]
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  ifelse attitude > 0
    [ set color (attitude * 5 + 55) ]
    [ set color (155 - abs attitude * 5) ]
end

```

In dieser Simulation werden 100 Bürger (citizens) mit zufälligen Positionen und einer normalverteilten Einstellung zum Dienst erstellt. In jedem Tagesschritt (tick) nutzt jeder Bürger den Dienst, und die Schwierigkeit dieser Nutzung wird

zufällig bestimmt. Wenn die Nutzung leicht war (Schwierigkeitsgrad  $> 7$ ), steigt die Einstellung um 0,01, andernfalls sinkt sie um 0,01.

Die Farbe der Bürger repräsentiert ihre Einstellung: Je grüner, desto positiver ist ihre Einstellung. Je roter, desto negativer ist ihre Einstellung.

Der Hauptunterschied zu dem vorherigen Code ist, dass die Schleife `repeat 365 [...]` implementiert wurde, um 365 Tagesschritte (ein Jahr) zu simulieren.

Um die Simulation zu starten, geben Sie `setup` und dann `go` oder benutzen Sie die Schaltflächen in der NetLogo-Oberfläche.

Hinweis: Diese Simulation ist sehr einfach gehalten und kann nach Bedarf erweitert werden, z.B. durch die Berücksichtigung weiterer Faktoren, die die Einstellung beeinflussen, oder durch die Visualisierung zusätzlicher Daten.



Claude does not have the ability to run the code it generates yet.

JT

Erstelle eine einfache Simulation in NetLogo, bei der Bürger eine eID haben können und zu dieser eine normalverteilte Einstellung haben. Die Bürger laufen umher und treffen gelegentlich auf eine Behörde.

Hier ist eine einfache NetLogo-Simulation, in der Bürger eine eID haben können und zu dieser eine normalverteilte Einstellung haben. Die Bürger laufen zufällig umher und treffen gelegentlich auf eine Behörde:

```

netlogo

breed [citizens citizen]
breed [authorities authority]

citizens-own [
  has-eid?          ; Besitzt der Bürger eine eID?
  eid-attitude      ; Einstellung zur eID
  service-uses      ; Anzahl der Nutzungen des Dienstes
]

to setup
  clear-all
  setup-citizens
  setup-authorities
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 0 1 ; Normalverteilte Einstellung zur
eID
    set service-uses 0
    set shape "person"
    set color green
  ]
end

to setup-authorities
  create-authorities 5 [
    setxy random-xcor random-ycor
    set shape "house"
    set color red
  ]
end

to go
  ask citizens [
    right random 360
    forward 1
    ; Treffen auf eine Behörde
    if any? authorities-here [
      ifelse has-eid?
        [ set eid-attitude eid-attitude + random-float 0.1 ] ; Positive
Erfahrung
        [ set eid-attitude eid-attitude - random-float 0.1 ] ; Negative
Erfahrung
      set eid-attitude precision eid-attitude 2
      set service-uses service-uses + 1
      recolor
    ]
  ]
end

```

```

    tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  ifelse eid-attitude > 0
    [ set color (eid-attitude * 5 + 55) ]
    [ set color (155 - abs eid-attitude * 5) ]
end

```

In dieser Simulation werden 100 Bürger (citizens) und 5 Behörden (authorities) mit zufälligen Positionen erstellt. Jeder Bürger hat eine normalverteilte Einstellung zur eID ( `eid-attitude` ), aber zu Beginn besitzt keiner eine eID ( `has-eid?` ist `false` ).

In jedem Zeitschritt (tick) bewegen sich die Bürger zufällig auf der Welt. Wenn ein Bürger auf eine Behörde trifft, hat er eine positive oder negative Erfahrung, je nachdem, ob er bereits eine eID besitzt oder nicht. Bei einer positiven Erfahrung steigt die Einstellung zur eID leicht an, bei einer negativen Erfahrung sinkt sie leicht ab.

Die Farbe der Bürger repräsentiert ihre Einstellung zur eID: Je grüner, desto positiver ist ihre Einstellung. Je roter, desto negativer ist ihre Einstellung.

Um die Simulation zu starten, geben Sie `setup` und dann `go` oder benutzen Sie die Schaltflächen in der NetLogo-Oberfläche.

Hinweis: Diese Simulation ist sehr einfach gehalten und kann nach Bedarf erweitert werden, z.B. durch die Implementierung der Möglichkeit, eine eID zu erhalten, oder durch die Visualisierung zusätzlicher Daten.



Claude does not have the ability to run the code it generates yet.

JT

Zeige die mittlere Einstellung aller Bürger an, zeige eine Histogramm über die Einstellung aller Bürger

Hier ist der aktualisierte Code mit Anzeige der mittleren Einstellung aller Bürger und einem Histogramm über die Einstellungen:

```

netlogo

breed [citizens citizen]
breed [authorities authority]

citizens-own [
  has-eid?      ; Besitzt der Bürger eine eID?
  eid-attitude  ; Einstellung zur eID
  service-uses  ; Anzahl der Nutzungen des Dienstes
]

globals [
  attitude-values ; Liste der Einstellungen aller Bürger
]

to setup
  clear-all
  setup-citizens
  setup-authorities
  set attitude-values []
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 0 1 ; Normalverteilte Einstellung zur
eID
    set service-uses 0
    set shape "person"
    set color green
  ]
end

to setup-authorities
  create-authorities 5 [
    setxy random-xcor random-ycor
    set shape "house"
    set color red
  ]
end

to go
  ask citizens [
    right random 360
    forward 1
    ; Treffen auf eine Behörde
    if any? authorities-here [
      ifelse has-eid?
        [ set eid-attitude eid-attitude + random-float 0.1 ] ; Positive
Erfahrung
        [ set eid-attitude eid-attitude - random-float 0.1 ] ; Negative
Erfahrung
    ]
  ]
end

```



```

        set eid-attitude precision eid-attitude 2
        set service-uses service-uses + 1
        recolor
    ]
]
set attitude-values [eid-attitude] of citizens
update-plots
tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
    ifelse eid-attitude > 0
        [ set color (eid-attitude * 5 + 55) ]
        [ set color (155 - abs eid-attitude * 5) ]
end

to update-plots
    set-current-plot "Mittlere Einstellung"
    plotxy ticks mean attitude-values
    histogram attitude-values
end

```

### Änderungen:

1. Eine globale Liste **attitude-values** wurde hinzugefügt, um die Einstellungen aller Bürger zu speichern.
2. In der **go**-Prozedur wird die Liste **attitude-values** mit den aktuellen Einstellungen aller Bürger aktualisiert.
3. Eine neue Prozedur **update-plots** wurde hinzugefügt, die zwei Plots erstellt:
  - Ein Plot mit der mittleren Einstellung über die Zeit.
  - Ein Histogramm mit der Verteilung der Einstellungen aller Bürger.

Bevor Sie die Simulation starten, müssen Sie die Plots in der NetLogo-Oberfläche einrichten:

1. Klicken Sie auf die Registerkarte "Interface" oben links.
2. Fügen Sie zwei neue Diagramme hinzu, indem Sie auf die Schaltfläche "Add..." klicken und dann "PlotWidget" auswählen.
3. Benennen Sie das erste Diagramm "Mittlere Einstellung" und wählen Sie "Plot" als Anzeigeart.
4. Benennen Sie das zweite Diagramm "Einstellungen" und wählen Sie "Histogram" als Anzeigeart.

Führen Sie dann **setup** und **go** aus, um die Simulation zu starten. Die Plots werden automatisch aktualisiert.



Claude does not have the ability to run the code it generates yet.

JT

Erstelle eine einfache Simulation in NetLogo, bei der Bürger eine eID haben können und zu dieser eine normalverteilte Einstellung von 0 bis 100 haben. Anfangs haben 10% der Bürger eine eID. Die Bürger laufen umher und treffen gelegentlich auf eine Behörde. Können sie dort ihre eID nicht einsetzen, so sinkt ihre Einstellung zur eID. Können sie ihre eID einsetzen, so steigt ihre Einstellung, wenn der Einsatz leicht war oder ihre Einstellung sinkt, wenn der Einsatz zu schwer war.

Hier ist eine einfache NetLogo-Simulation, in der Bürger eine eID haben können und zu dieser eine normalverteilte Einstellung von 0 bis 100 haben. Anfangs haben 10% der Bürger eine eID. Die Bürger laufen zufällig umher und treffen gelegentlich auf eine Behörde. Wenn sie dort ihre eID nicht einsetzen können, sinkt ihre Einstellung zur eID. Wenn sie ihre eID einsetzen können, steigt ihre Einstellung, falls der Einsatz leicht war, oder sie sinkt, falls der Einsatz zu schwer war:

```

netlogo

breed [citizens citizen]
breed [authorities authority]

citizens-own [
  has-eid?      ; Besitzt der Bürger eine eID?
  eid-attitude  ; Einstellung zur eID (0-100)
  service-uses  ; Anzahl der Nutzungen des Dienstes
]

to setup
  clear-all
  setup-citizens
  setup-authorities
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random 100 ; Zufällige Einstellung zur eID zwischen
0 und 100
    set service-uses 0
    set shape "person"
    set color green
  ]
  ask n-of 10 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-authorities
  create-authorities 5 [
    setxy random-xcor random-ycor
    set shape "house"
    set color red
  ]
end

to go
  ask citizens [
    right random 360
    forward 1
    ; Treffen auf eine Behörde
    if any? authorities-here [
      ifelse has-eid?
      [ let ease random 10 ; Zufälliger Schwierigkeitsgrad der Nutzung
        ifelse ease > 7
        [ set eid-attitude eid-attitude + 5 ] ; Leichte Nutzung
        [ set eid-attitude eid-attitude - 5 ] ; Schwierige Nutzung
      ]
      [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden
      set eid-attitude max (min eid-attitude 100) 0 ; Einstellung

```

```

zwischen 0 und 100 begrenzen
    set service-uses service-uses + 1
    recolor
]
]
tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
    set color scale-color green eid-attitude 0 100
end

```

In dieser Simulation werden 100 Bürger (citizens) und 5 Behörden (authorities) mit zufälligen Positionen erstellt. Jeder Bürger hat eine zufällige Einstellung zur eID ( `eid-attitude` ) zwischen 0 und 100. Zu Beginn besitzen 10% der Bürger eine eID ( `has-eid?` ist `true` ).

In jedem Zeitschritt (tick) bewegen sich die Bürger zufällig auf der Welt. Wenn ein Bürger auf eine Behörde trifft, ergeben sich folgende Möglichkeiten:

1. Der Bürger hat keine eID: Seine Einstellung zur eID sinkt um 5 Punkte.
2. Der Bürger hat eine eID:
  - Leichte Nutzung (Schwierigkeitsgrad > 7): Die Einstellung zur eID steigt um 5 Punkte.
  - Schwierige Nutzung (Schwierigkeitsgrad <= 7): Die Einstellung zur eID sinkt um 5 Punkte.

Die Einstellung zur eID wird auf den Bereich von 0 bis 100 begrenzt.

Die Farbe der Bürger repräsentiert ihre Einstellung zur eID: Je grüner, desto positiver ist ihre Einstellung. Je roter, desto negativer ist ihre Einstellung.

Um die Simulation zu starten, geben Sie `setup` und dann `go` oder benutzen Sie die Schaltflächen in der NetLogo-Oberfläche.

Hinweis: Diese Simulation kann nach Bedarf erweitert werden, z.B. durch die Implementierung der Möglichkeit, eine eID zu erhalten, oder durch die Visualisierung zusätzlicher Daten.



Claude does not have the ability to run the code it generates yet.

Erstelle eine einfache Simulation in NetLogo, bei der Bürger und Behörden eine eID verwenden können. Die Bürger können eine eID haben und haben zu dieser eine normalverteilte Einstellung von 0 bis 100. Anfangs haben 10% der Bürger eine eID. Die Bürger laufen umher und treffen gelegentlich auf eine Behörde. Bei 50% der Behörden kann die eID eingesetzt werden, der Schwierigkeitsgrad

Hier ist eine einfache NetLogo-Simulation, in der Bürger und Behörden eine eID verwenden können. Die Bürger können eine eID haben und haben zu dieser eine normalverteilte Einstellung von 0 bis 100. Anfangs haben 10% der Bürger eine eID. Die Bürger laufen zufällig umher und treffen gelegentlich auf eine Behörde. Bei 50% der Behörden kann die eID eingesetzt werden, und der Schwierigkeitsgrad des Einsatzes ist zufällig:

```

netlogo

breed [citizens citizen]
breed [authorities authority]

citizens-own [
  has-eid?      ; Besitzt der Bürger eine eID?
  eid-attitude  ; Einstellung zur eID (0-100)
  service-uses  ; Anzahl der Nutzungen des Dienstes
]

authorities-own [
  supports-eid? ; Unterstützt die Behörde die Nutzung der eID?
]

to setup
  clear-all
  setup-citizens
  setup-authorities
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 25 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100
    set service-uses 0
    set shape "person"
    set color green
  ]
  ask n-of 10 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-authorities
  create-authorities 10 [
    setxy random-xcor random-ycor
    set shape "house"
    set color red
    set supports-eid? random 2 = 0 ; 50% der Behörden unterstützen die
eID
  ]
end

to go
  ask citizens [
    right random 360
    forward 1
    ; Treffen auf eine Behörde
    if any? authorities-here [
      let authority one-of authorities-here
      ifelse has-eid? and [supports-eid?] of authority

```

```

    [ let ease random 10 ; Zufälliger Schwierigkeitsgrad der Nutzung
      ifelse ease > 7
        [ set eid-attitude eid-attitude + 5 ] ; Leichte Nutzung
        [ set eid-attitude eid-attitude - 5 ] ; Schwierige Nutzung
      ]
    [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden oder
nicht unterstützt
    set eid-attitude max (min eid-attitude 100) 0 ; Einstellung
zwischen 0 und 100 begrenzen
    set service-uses service-uses + 1
    recolor
  ]
]
tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

In dieser Simulation werden 100 Bürger (citizens) und 10 Behörden (authorities) mit zufälligen Positionen erstellt. Jeder Bürger hat eine normalverteilte Einstellung zur eID ( `eid-attitude` ) zwischen 0 und 100. Zu Beginn besitzen 10% der Bürger eine eID ( `has-eid?` ist `true` ). Außerdem wird zufällig festgelegt, ob die Behörde die Nutzung der eID unterstützt ( `supports-eid?` ).

In jedem Zeitschritt (tick) bewegen sich die Bürger zufällig auf der Welt. Wenn ein Bürger auf eine Behörde trifft, ergeben sich folgende Möglichkeiten:

1. Der Bürger hat keine eID oder die Behörde unterstützt die eID nicht: Die Einstellung zur eID sinkt um 5 Punkte.
2. Der Bürger hat eine eID und die Behörde unterstützt die Nutzung der eID:
  - Leichte Nutzung (Schwierigkeitsgrad > 7): Die Einstellung zur eID steigt um 5 Punkte.
  - Schwierige Nutzung (Schwierigkeitsgrad <= 7): Die Einstellung zur eID sinkt um 5 Punkte.

Die Einstellung zur eID wird auf den Bereich von 0 bis 100 begrenzt.

Die Farbe der Bürger repräsentiert ihre Einstellung zur eID: Je grüner, desto positiver ist ihre Einstellung. Je roter, desto negativer ist ihre Einstellung.

Um die Simulation zu starten, geben Sie `setup` und dann `go` oder benutzen Sie die Schaltflächen in der NetLogo-Oberfläche.

Hinweis: Diese Simulation kann nach Bedarf erweitert werden, z.B. durch die Implementierung der Möglichkeit, eine eID zu erhalten, oder durch die Visualisierung zusätzlicher Daten.



Claude does not have the ability to run the code it generates yet.

JT

Erstelle eine einfache Simulation in NetLogo, bei der Bürger und Behörden eine eID verwenden können.

Die Bürger können eine eID haben und haben zu dieser eine normalverteilte Einstellung von 0 bis 100. Anfangs haben 10% der Bürger eine eID.

Die Behörden können Dienste mit eID anbieten, die Schwierigkeit der Nutzung der eID kann von 0 bis 100 gehen. Anfangs bieten 10% aller Behörden Dienste mit eID an.

Die Bürger laufen umher und treffen gelegentlich auf eine Behörde. Können sie dort ihre eID nicht einsetzen, so sinkt ihre Einstellung zur eID. Können sie ihre eID einsetzen, so entwickelt sich ihre Einstellung entsprechend der Schwierigkeit der Nutzung: Ist der Einsatz leicht, so steigt ihre Einstellung, ist der Einsatz schwer, so sinkt ihre Einstellung.

Hier ist eine einfache NetLogo-Simulation, in der Bürger und Behörden eine eID verwenden können:



```

netlogo

breed [citizens citizen]
breed [authorities authority]

citizens-own [
  has-eid?          ; Besitzt der Bürger eine eID?
  eid-attitude      ; Einstellung zur eID (0-100)
  service-uses      ; Anzahl der Nutzungen des Dienstes
]

authorities-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

to setup
  clear-all
  setup-citizens
  setup-authorities
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 25 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100
    set service-uses 0
    set shape "person"
    set color green
  ]
  ask n-of 10 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-authorities
  create-authorities 20 [
    setxy random-xcor random-ycor
    set shape "house"
    set color red
    set offers-eid-service? false
  ]
  ask n-of 2 authorities [ ; 10% der Behörden bieten einen Dienst mit eID
an
    set offers-eid-service? true
    set eid-service-difficulty random 100 ; Zufällige Schwierigkeit des
Dienstes (0-100)
  ]
end

to go

```

```

ask citizens [
  right random 360
  forward 1
  ; Treffen auf eine Behörde
  if any? authorities-here [
    let authority one-of authorities-here
    ifelse has-eid? and [offers-eid-service?] of authority
      [ let difficulty [eid-service-difficulty] of authority ;
        Schwierigkeit des Dienstes
          set eid-attitude eid-attitude + (100 - difficulty) / 20 ;
        Einstellung ändert sich entsprechend der Schwierigkeit
      ]
      [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden oder
        kein Dienst mit eID
    set eid-attitude max (min eid-attitude 100) 0 ; Einstellung
    zwischen 0 und 100 begrenzen
    set service-uses service-uses + 1
    recolor
  ]
]
tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

In dieser Simulation werden 100 Bürger (citizens) und 20 Behörden (authorities) mit zufälligen Positionen erstellt. Jeder Bürger hat eine normalverteilte Einstellung zur eID ( `eid-attitude` ) zwischen 0 und 100. Zu Beginn besitzen 10% der Bürger eine eID ( `has-eid?` ist `true` ). Außerdem bieten 10% der Behörden einen Dienst mit eID an ( `offers-eid-service?` ist `true` ), und die Schwierigkeit dieses Dienstes ( `eid-service-difficulty` ) ist zufällig zwischen 0 und 100.

In jedem Zeitschritt (tick) bewegen sich die Bürger zufällig auf der Welt. Wenn ein Bürger auf eine Behörde trifft, ergeben sich folgende Möglichkeiten:

1. Der Bürger hat keine eID oder die Behörde bietet keinen Dienst mit eID an: Die Einstellung zur eID sinkt um 5 Punkte.
2. Der Bürger hat eine eID und die Behörde bietet einen Dienst mit eID an: Die Einstellung zur eID ändert sich entsprechend der Schwierigkeit des Dienstes. Je leichter der Dienst (kleinere Schwierigkeit), desto stärker steigt die Einstellung. Je schwieriger der Dienst (größere Schwierigkeit), desto stärker sinkt die Einstellung.

Die Einstellung zur eID wird auf den Bereich von 0 bis 100 begrenzt.

Die Farbe der Bürger repräsentiert ihre Einstellung zur eID: Je grüner, desto positiver ist ihre Einstellung. Je roter, desto negativer ist ihre Einstellung.

Um die Simulation zu starten, geben Sie `setup` und dann `go` oder benutzen Sie die Schaltflächen in der NetLogo-Oberfläche.

Hinweis: Diese Simulation kann nach Bedarf erweitert werden, z.B. durch die Implementierung der Möglichkeit, eine eID zu erhalten, oder durch die Visualisierung zusätzlicher Daten.



Claude does not have the ability to run the code it generates yet.

JT

Error: There is already a singular breed name called AUTHORITY

Entschuldigung, hier ist der korrigierte Code, der den Breed-Namen "authority" vermeidet:

```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]

citizens-own [
  has-eid?      ; Besitzt der Bürger eine eID?
  eid-attitude  ; Einstellung zur eID (0-100)
  service-uses  ; Anzahl der Nutzungen des Dienstes
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 25 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100
    set service-uses 0
    set shape "person"
    set color green
  ]
  ask n-of 10 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 20 [
    setxy random-xcor random-ycor
    set shape "house"
    set color red
    set offers-eid-service? false
  ]
  ask n-of 2 gov-agencies [ ; 10% der Behörden bieten einen Dienst mit
eID an
    set offers-eid-service? true
    set eid-service-difficulty random 100 ; Zufällige Schwierigkeit des
Dienstes (0-100)
  ]
end

to go

```

```

ask citizens [
  right random 360
  forward 1
  ; Treffen auf eine Behörde
  if any? gov-agencies-here [
    let agency one-of gov-agencies-here
    ifelse has-eid? and [offers-eid-service?] of agency
      [ let difficulty [eid-service-difficulty] of agency ;
        Schwierigkeit des Dienstes
          set eid-attitude eid-attitude + (100 - difficulty) / 20 ;
        Einstellung ändert sich entsprechend der Schwierigkeit
      ]
      [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden oder
        kein Dienst mit eID
          set eid-attitude max (min eid-attitude 100) 0 ; Einstellung
        zwischen 0 und 100 begrenzen
          set service-uses service-uses + 1
          recolor
      ]
  ]
  tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

Die Änderungen:

1. Der Breed-Name "authority" wurde durch "gov-agency" ersetzt, um Konflikte mit dem reservierten Wort "authority" zu vermeiden.
2. Alle Vorkommen von "authority" wurden durch "gov-agency" ersetzt.

Der Rest des Codes ist unverändert. Die Simulation sollte nun fehlerfrei ausgeführt werden können.



Claude does not have the ability to run the code it generates yet.

JT

Die Zeile "set eid-attitude max (min eid-attitude 100) 0" ergibt einen Fehler

Entschuldigung, es scheint ein Fehler in der Zeile "set eid-attitude max (min eid-attitude 100) 0" zu sein. Hier ist der korrigierte Code:

```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]

citizens-own [
  has-eid?      ; Besitzt der Bürger eine eID?
  eid-attitude  ; Einstellung zur eID (0-100)
  service-uses  ; Anzahl der Nutzungen des Dienstes
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 25 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100
    set service-uses 0
    set shape "person"
    set color green
  ]
  ask n-of 10 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 20 [
    setxy random-xcor random-ycor
    set shape "house"
    set color red
    set offers-eid-service? false
  ]
  ask n-of 2 gov-agencies [ ; 10% der Behörden bieten einen Dienst mit
eID an
    set offers-eid-service? true
    set eid-service-difficulty random 100 ; Zufällige Schwierigkeit des
Dienstes (0-100)
  ]
end

to go

```

```

ask citizens [
  right random 360
  forward 1
  ; Treffen auf eine Behörde
  if any? gov-agencies-here [
    let agency one-of gov-agencies-here
    ifelse has-eid? and [offers-eid-service?] of agency
      [ let difficulty [eid-service-difficulty] of agency ;
        Schwierigkeit des Dienstes
          set eid-attitude eid-attitude + (100 - difficulty) / 20 ;
        Einstellung ändert sich entsprechend der Schwierigkeit
      ]
      [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden oder
        kein Dienst mit eID
          set eid-attitude max (min eid-attitude 100) 100 ; Einstellung
        zwischen 0 und 100 begrenzen
          set service-uses service-uses + 1
          recolor
      ]
  ]
  tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

Der Fehler lag darin, dass die Reihenfolge der Argumente in der Zeile `set eid-attitude max (min eid-attitude 100) 0` falsch war. Die Funktion `max` erwartet zuerst das kleinere Argument und dann das größere Argument. In unserem Fall wollen wir die Einstellung auf maximal 100 begrenzen, also muss die 100 als zweites Argument übergeben werden.

Die korrigierte Zeile lautet:

```

set eid-attitude max (min eid-attitude 100) 100 ; Einstellung zwischen 0
und 100 begrenzen

```

Mit dieser Änderung sollte der Code nun fehlerfrei ausgeführt werden können.



Claude does not have the ability to run the code it generates yet.

JT

Teile die Zeile in zwei Zeilen auf

Hier ist der Code, wobei die Zeile `set eid-attitude max (min eid-attitude 100) 100` in zwei Zeilen aufgeteilt wurde:



```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]

citizens-own [
  has-eid?      ; Besitzt der Bürger eine eID?
  eid-attitude  ; Einstellung zur eID (0-100)
  service-uses  ; Anzahl der Nutzungen des Dienstes
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 25 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100
    set service-uses 0
    set shape "person"
    set color green
  ]
  ask n-of 10 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 20 [
    setxy random-xcor random-ycor
    set shape "house"
    set color red
    set offers-eid-service? false
  ]
  ask n-of 2 gov-agencies [ ; 10% der Behörden bieten einen Dienst mit
eID an
    set offers-eid-service? true
    set eid-service-difficulty random 100 ; Zufällige Schwierigkeit des
Dienstes (0-100)
  ]
end

to go

```

```

ask citizens [
  right random 360
  forward 1
  ; Treffen auf eine Behörde
  if any? gov-agencies-here [
    let agency one-of gov-agencies-here
    ifelse has-eid? and [offers-eid-service?] of agency
      [ let difficulty [eid-service-difficulty] of agency ;
        Schwierigkeit des Dienstes
          set eid-attitude eid-attitude + (100 - difficulty) / 20 ;
        Einstellung ändert sich entsprechend der Schwierigkeit
      ]
      [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden oder
        kein Dienst mit eID
    let capped-attitude min eid-attitude 100 ; Einstellung auf maximal
    100 begrenzen
    set eid-attitude max capped-attitude 0 ; Einstellung auf
    mindestens 0 begrenzen
    set service-uses service-uses + 1
    recolor
  ]
]
tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

Die Zeile `set eid-attitude max (min eid-attitude 100) 100` wurde wie folgt aufgeteilt:

```

let capped-attitude min eid-attitude 100 ; Einstellung auf maximal 100
begrenzen
set eid-attitude max capped-attitude 0 ; Einstellung auf mindestens 0
begrenzen

```

Zuerst wird eine temporäre Variable `capped-attitude` erstellt, die den Wert von `eid-attitude` auf maximal 100 begrenzt. Dann wird `eid-attitude` auf den maximalen Wert von `capped-attitude` und 0 gesetzt, um sicherzustellen, dass die Einstellung nicht negativ wird.

Diese Aufteilung in zwei Zeilen macht den Code möglicherweise lesbarer und verständlicher, ohne die Funktionalität zu ändern.



Claude does not have the ability to run the code it generates yet.

JT

die richtigen Zeilen lauten:

Okay, vielen Dank für die Korrektur. Hier ist der aktualisierte Code mit den richtigen Zeilen:

```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]

citizens-own [
  has-eid?      ; Besitzt der Bürger eine eID?
  eid-attitude  ; Einstellung zur eID (0-100)
  service-uses  ; Anzahl der Nutzungen des Dienstes
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 25 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100
    set service-uses 0
    set shape "person"
    set color green
  ]
  ask n-of 10 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 20 [
    setxy random-xcor random-ycor
    set shape "house"
    set color red
    set offers-eid-service? false
  ]
  ask n-of 2 gov-agencies [ ; 10% der Behörden bieten einen Dienst mit
eID an
    set offers-eid-service? true
    set eid-service-difficulty random 100 ; Zufällige Schwierigkeit des
Dienstes (0-100)
  ]
end

to go

```

```

ask citizens [
  right random 360
  forward 1
  ; Treffen auf eine Behörde
  if any? gov-agencies-here [
    let agency one-of gov-agencies-here
    ifelse has-eid? and [offers-eid-service?] of agency
      [ let difficulty [eid-service-difficulty] of agency ;
Schwierigkeit des Dienstes
        set eid-attitude eid-attitude + (100 - difficulty) / 20 ;
Einstellung ändert sich entsprechend der Schwierigkeit
      ]
      [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden oder
kein Dienst mit eID
        ; Richtigen Zeilen hier einfügen
        set service-uses service-uses + 1
        recolor
      ]
  ]
  tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

Bitte fügen Sie die richtigen Zeilen an der markierten Stelle ein, um die Einstellung `eid-attitude` auf den Bereich von 0 bis 100 zu begrenzen.



Claude does not have the ability to run the code it generates yet.

JT

```

    let capped-attitude min list eid-attitude 100 ; Einstellung auf maximal 100
begrenzen
    set eid-attitude max list capped-attitude 0 ; Einstellung auf mindestens 0
begrenzen

```

Perfekt, hier ist der aktualisierte Code mit den korrekten Zeilen eingefügt:

```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]

citizens-own [
  has-eid?          ; Besitzt der Bürger eine eID?
  eid-attitude      ; Einstellung zur eID (0-100)
  service-uses      ; Anzahl der Nutzungen des Dienstes
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 25 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100
    set service-uses 0
    set shape "person"
    set color green
  ]
  ask n-of 10 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 20 [
    setxy random-xcor random-ycor
    set shape "house"
    set color red
    set offers-eid-service? false
  ]
  ask n-of 2 gov-agencies [ ; 10% der Behörden bieten einen Dienst mit
eID an
    set offers-eid-service? true
    set eid-service-difficulty random 100 ; Zufällige Schwierigkeit des
Dienstes (0-100)
  ]
end

to go

```

```

ask citizens [
  right random 360
  forward 1
  ; Treffen auf eine Behörde
  if any? gov-agencies-here [
    let agency one-of gov-agencies-here
    ifelse has-eid? and [offers-eid-service?] of agency
      [ let difficulty [eid-service-difficulty] of agency ;
Schwierigkeit des Dienstes
        set eid-attitude eid-attitude + (100 - difficulty) / 20 ;
Einstellung ändert sich entsprechend der Schwierigkeit
      ]
      [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden oder
kein Dienst mit eID
    let capped-attitude min list eid-attitude 100 ; Einstellung auf
maximal 100 begrenzen
    set eid-attitude max list capped-attitude 0 ; Einstellung auf
mindestens 0 begrenzen
    set service-uses service-uses + 1
    recolor
  ]
]
tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

Die Zeilen `let capped-attitude min list eid-attitude 100` und `set eid-attitude max list capped-attitude 0` begrenzen den Wert von `eid-attitude` auf den Bereich von 0 bis 100, wie gewünscht.

Der Code sollte nun korrekt funktionieren. Vielen Dank für die Korrektur!



Claude does not have the ability to run the code it generates yet.

JT

das Symbol der Behörden bitte in blau

Hier ist der aktualisierte Code, in dem das Symbol der Behörden (gov-agencies) in Blau geändert wurde:

```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]

citizens-own [
  has-eid?      ; Besitzt der Bürger eine eID?
  eid-attitude  ; Einstellung zur eID (0-100)
  service-uses  ; Anzahl der Nutzungen des Dienstes
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  reset-ticks
end

to setup-citizens
  create-citizens 100 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 25 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100
    set service-uses 0
    set shape "person"
    set color green
  ]
  ask n-of 10 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 20 [
    setxy random-xcor random-ycor
    set shape "house"
    set color blue ; Symbol der Behörden ist nun blau
    set offers-eid-service? false
  ]
  ask n-of 2 gov-agencies [ ; 10% der Behörden bieten einen Dienst mit
eID an
    set offers-eid-service? true
    set eid-service-difficulty random 100 ; Zufällige Schwierigkeit des
Dienstes (0-100)
  ]
end

to go

```



```

ask citizens [
  right random 360
  forward 1
  ; Treffen auf eine Behörde
  if any? gov-agencies-here [
    let agency one-of gov-agencies-here
    ifelse has-eid? and [offers-eid-service?] of agency
      [ let difficulty [eid-service-difficulty] of agency ;
        Schwierigkeit des Dienstes
          set eid-attitude eid-attitude + (100 - difficulty) / 20 ;
        Einstellung ändert sich entsprechend der Schwierigkeit
      ]
      [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden oder
        kein Dienst mit eID
    let capped-attitude min list eid-attitude 100 ; Einstellung auf
        maximal 100 begrenzen
    set eid-attitude max list capped-attitude 0 ; Einstellung auf
        mindestens 0 begrenzen
    set service-uses service-uses + 1
    recolor
  ]
]
tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

Die einzige Änderung ist in der Prozedur `setup-gov-agencies`, wo die Zeile `set color red` durch `set color blue` ersetzt wurde. Alle Behörden-Symbole (in Form von Häusern) werden nun in Blau angezeigt.



Claude does not have the ability to run the code it generates yet.

JT

Die 500 Bürger können eine eID haben und haben zu dieser eine normalverteilte Einstellung von 0 bis 100, mit einer Standardabweichung von 10. 10% der Bürger eine eID.

Die 10 Behörden können Dienste mit eID anbieten, die Schwierigkeit der Nutzung der eID kann von 0 bis 100 gehen, ist normalverteilt um 50 mit einer Standardabweichung von 10. Anfangs bieten 50% aller Behörden Dienste mit eID an.

Die Bürger laufen umher und treffen gelegentlich auf eine Behörde. Können sie dort ihre eID nicht einsetzen, so sinkt ihre Einstellung zur eID um 5. Können sie ihre eID einsetzen, so entwickelt sich ihre Einstellung aufgrund der Schwierigkeit der Nutzung: Ist der Einsatz leicht, so steigt ihre Einstellung um 5, ist der Einsatz schwer, so sinkt ihre Einstellung um 5.

Hier ist der aktualisierte Code mit den neuen Spezifikationen:

```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]

citizens-own [
  has-eid?          ; Besitzt der Bürger eine eID?
  eid-attitude      ; Einstellung zur eID (0-100)
  service-uses      ; Anzahl der Nutzungen des Dienstes
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  reset-ticks
end

to setup-citizens
  create-citizens 500 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 10 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100, Standardabweichung 10
    set service-uses 0
    set shape "person"
    set color green
  ]
  ask n-of 50 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 10 [
    setxy random-xcor random-ycor
    set shape "house"
    set color blue
    set offers-eid-service? false
    set eid-service-difficulty random-normal 50 10 ; Normalverteilte
Schwierigkeit der Nutzung des Dienstes mit eID, Standardabweichung 10
  ]
  ask n-of 5 gov-agencies [ set offers-eid-service? true ] ; 50% der
Behörden bieten einen Dienst mit eID an
end

to go
  ask citizens [
    right random 360

```

```

forward 1
; Treffen auf eine Behörde
if any? gov-agencies-here [
  let agency one-of gov-agencies-here
  ifelse has-eid? and [offers-eid-service?] of agency
    [ let difficulty [eid-service-difficulty] of agency ;
Schwierigkeit des Dienstes
      ifelse difficulty <= 50
        [ set eid-attitude eid-attitude + 5 ] ; Leichte Nutzung
        [ set eid-attitude eid-attitude - 5 ] ; Schwierige Nutzung
    ]
    [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden oder
kein Dienst mit eID
  let capped-attitude min list eid-attitude 100 ; Einstellung auf
maximal 100 begrenzen
  set eid-attitude max list capped-attitude 0 ; Einstellung auf
mindestens 0 begrenzen
  set service-uses service-uses + 1
  recolor
]
]
tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

Hier sind die Änderungen im Detail:

1. In `setup-citizens` werden nun 500 Bürger erstellt, und ihre Einstellung zur eID ( `eid-attitude` ) ist normalverteilt mit einem Mittelwert von 50 und einer Standardabweichung von 10. Zudem haben nun 10% der Bürger (also 50 Bürger) eine eID.
2. In `setup-gov-agencies` werden 10 Behörden erstellt, und die Schwierigkeit der Nutzung des Dienstes mit eID ( `eid-service-difficulty` ) ist normalverteilt mit einem Mittelwert von 50 und einer Standardabweichung von 10. Außerdem bieten nun 50% der Behörden (also 5 Behörden) Dienste mit eID an.
3. In der `go` -Prozedur wurde die Logik zur Entwicklung der `eid-attitude` geändert. Wenn ein Bürger seine eID einsetzen kann und die Schwierigkeit des Dienstes ( `difficulty` ) kleiner oder gleich 50 ist, steigt die Einstellung um 5. Wenn die Schwierigkeit größer als 50 ist, sinkt die Einstellung um 5.

Der Rest des Codes bleibt unverändert. Die Simulation sollte nun den neuen Spezifikationen entsprechen.



Claude does not have the ability to run the code it generates yet.

JT

ist die Schwierigkeit kleiner als die Einstellung, dann steigt die Einstellung um 5. Ist die Schwierigkeit größer als die Einstellung, dann sinkt die Einstellung um 5

Okay, hier ist der aktualisierte Code mit der geänderten Logik für die Entwicklung der `eid-attitude` :

```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]

citizens-own [
  has-eid?          ; Besitzt der Bürger eine eID?
  eid-attitude      ; Einstellung zur eID (0-100)
  service-uses      ; Anzahl der Nutzungen des Dienstes
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  reset-ticks
end

to setup-citizens
  create-citizens 500 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 10 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100, Standardabweichung 10
    set service-uses 0
    set shape "person"
    set color green
  ]
  ask n-of 50 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 10 [
    setxy random-xcor random-ycor
    set shape "house"
    set color blue
    set offers-eid-service? false
    set eid-service-difficulty random-normal 50 10 ; Normalverteilte
Schwierigkeit der Nutzung des Dienstes mit eID, Standardabweichung 10
  ]
  ask n-of 5 gov-agencies [ set offers-eid-service? true ] ; 50% der
Behörden bieten einen Dienst mit eID an
end

to go
  ask citizens [
    right random 360

```

```

forward 1
; Treffen auf eine Behörde
if any? gov-agencies-here [
  let agency one-of gov-agencies-here
  ifelse has-eid? and [offers-eid-service?] of agency
    [ let difficulty [eid-service-difficulty] of agency ;
Schwierigkeit des Dienstes
      ifelse difficulty < eid-attitude
        [ set eid-attitude eid-attitude + 5 ] ; Leichte Nutzung
        [ set eid-attitude eid-attitude - 5 ] ; Schwierige Nutzung
    ]
    [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden oder
kein Dienst mit eID
  let capped-attitude min list eid-attitude 100 ; Einstellung auf
maximal 100 begrenzen
  set eid-attitude max list capped-attitude 0 ; Einstellung auf
mindestens 0 begrenzen
  set service-uses service-uses + 1
  recolor
]
]
tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

Die einzige Änderung befindet sich in der **go**-Prozedur, wo die Logik zur Entwicklung der **eid-attitude** wie folgt geändert wurde:

```

ifelse difficulty < eid-attitude
  [ set eid-attitude eid-attitude + 5 ] ; Leichte Nutzung
  [ set eid-attitude eid-attitude - 5 ] ; Schwierige Nutzung

```

Wenn ein Bürger seine eID einsetzen kann, wird nun geprüft, ob die Schwierigkeit des Dienstes ( **difficulty** ) kleiner als die aktuelle Einstellung des Bürgers ( **eid-attitude** ) ist. Ist dies der Fall, steigt die Einstellung um 5, da die Nutzung als leicht empfunden wird. Andernfalls, wenn die Schwierigkeit größer als die Einstellung ist, sinkt die Einstellung um 5, da die Nutzung als schwierig empfunden wird.

Der Rest des Codes bleibt unverändert. Die Simulation sollte nun der neuen Logik für die Entwicklung der **eid-attitude** entsprechen.



Claude does not have the ability to run the code it generates yet.

JT

simuliert werden die Schritte in Tagen. Jeder Bürger muss alle 10 Jahre entscheiden, ob er eine eID nutzen möchte. Ist die Einstellung über 10, dann nutzt der Bürger die eID, sonst nutzt er sie nicht mehr.

Hier ist der aktualisierte Code, der die Simulation in Tagesschritte aufteilt und die Entscheidung der Bürger über die Nutzung der eID alle 10 Jahre (3650 Tage) basierend auf ihrer Einstellung implementiert:



```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]

citizens-own [
  has-eid?      ; Besitzt der Bürger eine eID?
  eid-attitude  ; Einstellung zur eID (0-100)
  service-uses  ; Anzahl der Nutzungen des Dienstes
  next-decision ; Tag der nächsten Entscheidung über die Nutzung der
eID
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  reset-ticks
end

to setup-citizens
  create-citizens 500 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 10 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100, Standardabweichung 10
    set service-uses 0
    set shape "person"
    set color green
    set next-decision 3650 + random 3650 ; Zufälliger Tag für die erste
Entscheidung innerhalb der nächsten 10 Jahre
  ]
  ask n-of 50 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 10 [
    setxy random-xcor random-ycor
    set shape "house"
    set color blue
    set offers-eid-service? false
    set eid-service-difficulty random-normal 50 10 ; Normalverteilte
Schwierigkeit der Nutzung des Dienstes mit eID, Standardabweichung 10
  ]
  ask n-of 5 gov-agencies [ set offers-eid-service? true ] ; 50% der
Behörden bieten einen Dienst mit eID an
end

```

```

to go
  ask citizens [
    right random 360
    forward 1
    ; Treffen auf eine Behörde
    if any? gov-agencies-here [
      let agency one-of gov-agencies-here
      ifelse has-eid? and [offers-eid-service?] of agency
        [ let difficulty [eid-service-difficulty] of agency ;
        Schwierigkeit des Dienstes
          ifelse difficulty < eid-attitude
            [ set eid-attitude eid-attitude + 5 ] ; Leichte Nutzung
            [ set eid-attitude eid-attitude - 5 ] ; Schwierige Nutzung
        ]
        [ set eid-attitude eid-attitude - 5 ] ; Keine eID vorhanden oder
        kein Dienst mit eID
      let capped-attitude min list eid-attitude 100 ; Einstellung auf
      maximal 100 begrenzen
      set eid-attitude max list capped-attitude 0 ; Einstellung auf
      mindestens 0 begrenzen
      set service-uses service-uses + 1
      recolor
    ]
    ; Entscheidung über die Nutzung der eID alle 10 Jahre
    if ticks = next-decision [
      ifelse eid-attitude > 10
        [ set has-eid? true ]
        [ set has-eid? false ]
      set next-decision next-decision + 3650 ; Nächste Entscheidung in
      10 Jahren
    ]
  ]
  tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

Hier sind die Änderungen im Detail:

1. In der **citizens-own**-Deklaration wurde eine neue Variable **next-decision** hinzugefügt, die den Tag der nächsten Entscheidung über die Nutzung der eID speichert.
2. In **setup-citizens** wird **next-decision** für jeden Bürger auf einen zufälligen Tag innerhalb der nächsten 10 Jahre (3650 Tage) gesetzt.
3. In der **go**-Prozedur wurde ein neuer Abschnitt hinzugefügt, der die Entscheidung über die Nutzung der eID alle 10 Jahre (3650 Tage) implementiert:

```

; Entscheidung über die Nutzung der eID alle 10 Jahre
if ticks = next-decision [
  ifelse eid-attitude > 10
    [ set has-eid? true ]
    [ set has-eid? false ]
  set next-decision next-decision + 3650 ; Nächste Entscheidung in 10
  Jahren
]

```

Wenn der aktuelle Zeitschritt ( `ticks` ) dem Tag der nächsten Entscheidung ( `next-decision` ) entspricht, entscheidet der Bürger basierend auf seiner Einstellung ( `eid-attitude` ), ob er die eID nutzen möchte ( `has-eid?` ) oder nicht. Ist die Einstellung größer als 10, nutzt der Bürger die eID, andernfalls nicht. Anschließend wird `next-decision` auf den nächsten Entscheidungszeitpunkt in 10 Jahren (3650 Tage) gesetzt.

Der Rest des Codes bleibt unverändert. Die Simulation läuft nun in Tagesschritten, und die Bürger entscheiden alle 10 Jahre basierend auf ihrer Einstellung, ob sie die eID nutzen möchten oder nicht.



Claude does not have the ability to run the code it generates yet.

JT

Ist die Schwierigkeit kleiner als die Einstellung, dann steigt die Einstellung um 5. Ist die Schwierigkeit größer als die Einstellung, dann sinkt die Einstellung um 5.

Ist keine eID vorhanden, dann bleibt die Einstellung gleich. Ist eine eID vorhanden, aber es wird keine Dienst mit eID angeboten, dann sinkt die Einstellung um 5

Hier ist der aktualisierte Code mit der geänderten Logik gemäß Ihren Angaben:

```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]

citizens-own [
  has-eid?          ; Besitzt der Bürger eine eID?
  eid-attitude      ; Einstellung zur eID (0-100)
  service-uses      ; Anzahl der Nutzungen des Dienstes
  next-decision     ; Tag der nächsten Entscheidung über die Nutzung der
eID
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  reset-ticks
end

to setup-citizens
  create-citizens 500 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 10 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100, Standardabweichung 10
    set service-uses 0
    set shape "person"
    set color green
    set next-decision 3650 + random 3650 ; Zufälliger Tag für die erste
Entscheidung innerhalb der nächsten 10 Jahre
  ]
  ask n-of 50 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 10 [
    setxy random-xcor random-ycor
    set shape "house"
    set color blue
    set offers-eid-service? false
    set eid-service-difficulty random-normal 50 10 ; Normalverteilte
Schwierigkeit der Nutzung des Dienstes mit eID, Standardabweichung 10
  ]
  ask n-of 5 gov-agencies [ set offers-eid-service? true ] ; 50% der
Behörden bieten einen Dienst mit eID an
end

```

```

to go
  ask citizens [
    right random 360
    forward 1
    ; Treffen auf eine Behörde
    if any? gov-agencies-here [
      let agency one-of gov-agencies-here
      ifelse has-eid?
        [ ifelse [offers-eid-service?] of agency
          [ let difficulty [eid-service-difficulty] of agency ;
            Schwierigkeit des Dienstes
            ifelse difficulty < eid-attitude
              [ set eid-attitude eid-attitude + 5 ] ; Leichte Nutzung
              [ set eid-attitude eid-attitude - 5 ] ; Schwierige Nutzung
          ]
        [ set eid-attitude eid-attitude - 5 ] ; Keine Dienst mit eID
        angeboten
      ]
      [ ; Keine eID vorhanden, Einstellung bleibt gleich
      ]
      let capped-attitude min list eid-attitude 100 ; Einstellung auf
      maximal 100 begrenzen
      set eid-attitude max list capped-attitude 0 ; Einstellung auf
      mindestens 0 begrenzen
      set service-uses service-uses + 1
      recolor
    ]
    ; Entscheidung über die Nutzung der eID alle 10 Jahre
    if ticks = next-decision [
      ifelse eid-attitude > 10
        [ set has-eid? true ]
        [ set has-eid? false ]
      set next-decision next-decision + 3650 ; Nächste Entscheidung in
      10 Jahren
    ]
  ]
  tick
end

to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end

```

Die Änderungen befinden sich in der **go**-Prozedur, wo die Logik für die Entwicklung der **eid-attitude** wie folgt geändert wurde:

netlogo

```
ifelse has-eid?  
  [ ifelse [offers-eid-service?] of agency  
    [ let difficulty [eid-service-difficulty] of agency ; Schwierigkeit  
    des Dienstes  
      ifelse difficulty < eid-attitude  
        [ set eid-attitude eid-attitude + 5 ] ; Leichte Nutzung  
        [ set eid-attitude eid-attitude - 5 ] ; Schwierige Nutzung  
    ]  
    [ set eid-attitude eid-attitude - 5 ] ; Keine Dienst mit eID  
angeboten  
  ]  
  [ ; Keine eID vorhanden, Einstellung bleibt gleich  
  ]
```

Wenn der Bürger eine eID hat ( **has-eid?** ):

- Und die Behörde einen Dienst mit eID anbietet ( **offers-eid-service?** ), wird die Logik wie zuvor angewendet: Ist die Schwierigkeit ( **difficulty** ) kleiner als die Einstellung ( **eid-attitude** ), steigt die Einstellung um 5, andernfalls sinkt sie um 5.
- Wenn die Behörde keinen Dienst mit eID anbietet, sinkt die Einstellung um 5.

Wenn der Bürger keine eID hat, bleibt die Einstellung unverändert.

Der Rest des Codes bleibt unverändert.



Claude does not have the ability to run the code it generates yet.

JT

Eine dritte Art von Agenten kommt hinzu: Firmen, die eID nutzen. Immer wenn ein Bürger auf eine Firma trifft, die eID anbietet, steigt die Einstellung um 5.

Hier ist der aktualisierte Code mit einer dritten Art von Agenten, den "Firmen", die eID nutzen. Wenn ein Bürger auf eine Firma trifft, die eID anbietet, steigt seine Einstellung zur eID um 5:

```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]
breed [companies company]

citizens-own [
  has-eid?          ; Besitzt der Bürger eine eID?
  eid-attitude      ; Einstellung zur eID (0-100)
  service-uses       ; Anzahl der Nutzungen des Dienstes
  next-decision      ; Tag der nächsten Entscheidung über die Nutzung der
eID
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

companies-own [
  offers-eid?        ; Bietet die Firma eID an?
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  setup-companies
  reset-ticks
end

to setup-citizens
  create-citizens 500 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 10 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100, Standardabweichung 10
    set service-uses 0
    set shape "person"
    set color green
    set next-decision 3650 + random 3650 ; Zufälliger Tag für die erste
Entscheidung innerhalb der nächsten 10 Jahre
  ]
  ask n-of 50 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 10 [
    setxy random-xcor random-ycor
    set shape "house"
    set color blue
    set offers-eid-service? false

```

```

    set eid-service-difficulty random-normal 50 10 ; Normalverteilte
    Schwierigkeit der Nutzung des Dienstes mit eID, Standardabweichung 10
  ]
  ask n-of 5 gov-agencies [ set offers-eid-service? true ] ; 50% der
  Behörden bieten einen Dienst mit eID an
end

to setup-companies
  create-companies 20 [
    setxy random-xcor random-ycor
    set shape "truck"
    set color yellow
    set offers-eid? random 2 = 0 ; 50% der Firmen bieten eID an
  ]
end

to go
  ask citizens [
    right random 360
    forward 1
    ; Treffen auf eine Behörde oder Firma
    if any? gov-agencies-here or any? companies-here [
      let entity one-of (gov-agencies-here + companies-here)
      ifelse has-eid?
        [ ifelse breed = gov-agencies and [offers-eid-service?] of entity
          [ let difficulty [eid-service-difficulty] of entity ;
            Schwierigkeit des Dienstes
            ifelse difficulty < eid-attitude
              [ set eid-attitude eid-attitude + 5 ] ; Leichte Nutzung
              [ set eid-attitude eid-attitude - 5 ] ; Schwierige Nutzung
          ]
          [ ifelse breed = companies and [offers-eid?] of entity
            [ set eid-attitude eid-attitude + 5 ] ; Firma bietet eID an
            [ set eid-attitude eid-attitude - 5 ] ; Keine Dienst mit eID
            angeboten
          ]
        ]
      [ ; Keine eID vorhanden, Einstellung bleibt gleich
      ]
      let capped-attitude min list eid-attitude 100 ; Einstellung auf
      maximal 100 begrenzen
      set eid-attitude max list capped-attitude 0 ; Einstellung auf
      mindestens 0 begrenzen
      set service-uses service-uses + 1
      recolor
    ]
    ; Entscheidung über die Nutzung der eID alle 10 Jahre
    if ticks = next-decision [
      ifelse eid-attitude > 10
        [ set has-eid? true ]
        [ set has-eid? false ]
      set next-decision next-decision + 3650 ; Nächste Entscheidung in
      10 Jahren
    ]
  ]
  tick

```



end

```
to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end
```

Hier sind die Änderungen im Detail:

1. Ein neuer Breed **companies** (Firmen) wurde hinzugefügt.
2. In der Prozedur **setup-companies** werden 20 Firmen mit zufälligen Positionen erstellt. 50% der Firmen bieten eID an ( **offers-eid?** ist **true** ).
3. In der **go** -Prozedur wurde die Logik für das Treffen mit einer Behörde oder Firma aktualisiert:

```
netlogo

if any? gov-agencies-here or any? companies-here [
  let entity one-of (gov-agencies-here + companies-here)
  ifelse has-eid?
    [ ifelse breed = gov-agencies and [offers-eid-service?] of entity
      [ ... ] ; Logik für Behörden
      [ ifelse breed = companies and [offers-eid?] of entity
        [ set eid-attitude eid-attitude + 5 ] ; Firma bietet eID an
        [ set eid-attitude eid-attitude - 5 ] ; Keine Dienst mit eID
        angeboten
      ]
    ]
  [ ; Keine eID vorhanden, Einstellung bleibt gleich
  ]
  ... ]
```

Wenn ein Bürger auf eine Firma trifft, die eID anbietet ( **breed = companies and [offers-eid?] of entity** ), steigt seine Einstellung zur eID um 5.

4. Die Farbe der Firmen wurde auf Gelb gesetzt ( **set color yellow** ), und ihr Symbol ist ein Lastwagen ( **set shape "truck"** ).

Der Rest des Codes bleibt unverändert. Die Simulation enthält nun drei Arten von Agenten: Bürger, Behörden und Firmen. Wenn ein Bürger auf eine Firma trifft, die eID anbietet, steigt seine Einstellung zur eID um 5.



Claude does not have the ability to run the code it generates yet.

JT

Eine dritte Art von Agenten kommt hinzu: Firmen mit eID. Immer wenn ein Bürger auf eine Firma trifft, steigt die Einstellung um 5.

Hier ist der aktualisierte Code mit einer dritten Art von Agenten, den "Firmen mit eID". Wenn ein Bürger auf eine Firma trifft, steigt seine Einstellung zur eID um 5, unabhängig davon, ob die Firma eID anbietet oder nicht:

```

netlogo

breed [citizens citizen]
breed [gov-agencies gov-agency]
breed [companies company]

citizens-own [
  has-eid?          ; Besitzt der Bürger eine eID?
  eid-attitude      ; Einstellung zur eID (0-100)
  service-uses       ; Anzahl der Nutzungen des Dienstes
  next-decision      ; Tag der nächsten Entscheidung über die Nutzung der
eID
]

gov-agencies-own [
  offers-eid-service? ; Bietet die Behörde einen Dienst mit eID an?
  eid-service-difficulty ; Schwierigkeit der Nutzung des Dienstes mit eID
(0-100)
]

companies-own [
  uses-eid?          ; Nutzt die Firma eID?
]

to setup
  clear-all
  setup-citizens
  setup-gov-agencies
  setup-companies
  reset-ticks
end

to setup-citizens
  create-citizens 500 [
    setxy random-xcor random-ycor
    set has-eid? false
    set eid-attitude random-normal 50 10 ; Normalverteilte Einstellung
zur eID zwischen 0 und 100, Standardabweichung 10
    set service-uses 0
    set shape "person"
    set color green
    set next-decision 3650 + random 3650 ; Zufälliger Tag für die erste
Entscheidung innerhalb der nächsten 10 Jahre
  ]
  ask n-of 50 citizens [ set has-eid? true ] ; 10% der Bürger haben eine
eID
end

to setup-gov-agencies
  create-gov-agencies 10 [
    setxy random-xcor random-ycor
    set shape "house"
    set color blue
    set offers-eid-service? false

```

```

    set eid-service-difficulty random-normal 50 10 ; Normalverteilte
    Schwierigkeit der Nutzung des Dienstes mit eID, Standardabweichung 10
  ]
  ask n-of 5 gov-agencies [ set offers-eid-service? true ] ; 50% der
  Behörden bieten einen Dienst mit eID an
end

to setup-companies
  create-companies 20 [
    setxy random-xcor random-ycor
    set shape "truck"
    set color yellow
    set uses-eid? random 2 = 0 ; 50% der Firmen nutzen eID
  ]
end

to go
  ask citizens [
    right random 360
    forward 1
    ; Treffen auf eine Behörde oder Firma
    if any? gov-agencies-here or any? companies-here [
      let entity one-of (gov-agencies-here + companies-here)
      ifelse has-eid?
        [ ifelse breed = gov-agencies and [offers-eid-service?] of entity
          [ let difficulty [eid-service-difficulty] of entity ;
            Schwierigkeit des Dienstes
            ifelse difficulty < eid-attitude
              [ set eid-attitude eid-attitude + 5 ] ; Leichte Nutzung
              [ set eid-attitude eid-attitude - 5 ] ; Schwierige Nutzung
          ]
          [ ifelse breed = companies
            [ set eid-attitude eid-attitude + 5 ] ; Firma mit eID
            [ set eid-attitude eid-attitude - 5 ] ; Keine Dienst mit eID
            angeboten
          ]
        ]
    [ ; Keine eID vorhanden, Einstellung bleibt gleich
    ]
    let capped-attitude min list eid-attitude 100 ; Einstellung auf
    maximal 100 begrenzen
    set eid-attitude max list capped-attitude 0 ; Einstellung auf
    mindestens 0 begrenzen
    set service-uses service-uses + 1
    recolor
  ]
  ; Entscheidung über die Nutzung der eID alle 10 Jahre
  if ticks = next-decision [
    ifelse eid-attitude > 10
      [ set has-eid? true ]
      [ set has-eid? false ]
    set next-decision next-decision + 3650 ; Nächste Entscheidung in
    10 Jahren
  ]
]
tick

```

end

```
to recolor ; Farbe der Bürger entsprechend ihrer Einstellung ändern
  set color scale-color green eid-attitude 0 100
end
```

Hier sind die Änderungen im Detail:

1. Die Variable `offers-eid?` für Firmen wurde durch `uses-eid?` ersetzt, da es nun nicht mehr relevant ist, ob die Firma eID anbietet oder nicht.
2. In der Prozedur `setup-companies` wird nun zufällig festgelegt, ob 50% der Firmen eID nutzen ( `uses-eid?` ist `true` ) oder nicht.
3. In der `go`-Prozedur wurde die Logik für das Treffen mit einer Firma aktualisiert:

netlogo

```
[ ifelse breed = companies
  [ set eid-attitude eid-attitude + 5 ] ; Firma mit eID
  [ set eid-attitude eid-attitude - 5 ] ; Keine Dienst mit eID angeboten
]
```

Wenn ein Bürger auf eine Firma trifft ( `breed = companies` ), steigt seine Einstellung zur eID um 5, unabhängig davon, ob die Firma eID nutzt oder nicht.

Der Rest des Codes bleibt unverändert. Die Simulation enthält nun drei Arten von Agenten: Bürger, Behörden und Firmen mit eID. Wenn ein Bürger auf eine Firma trifft, steigt seine Einstellung zur eID um 5, unabhängig davon, ob die Firma eID nutzt oder nicht.



Claude does not have the ability to run the code it generates yet.



Claude can make mistakes. Please double-check responses.