problème des 100 lampes
programmes python

programme 1

```
#Problème des 100 lampes

def lights(n):
    L=[]
    for k in range(100):
        L.append(0)   #Initialisation de la liste de 100 lampes, 0 indique une lampe éteinte
    for i in range(1, n+1): #Balayage des étapes 1 à n
        for k in range(100):
            if (k+1)%i==0:  #Commutation des lampes d'indice divisible par i
                if L[k] == 0:
                    L[k]= 1   #Une lampe éteinte est allumée
                else:
                    L[k] = 0  #Une lampe allumée est éteinte
    return(L)
```

programme 2

```
#Problème des 100 lampes

def lights(n):
    L = [0] * 100  # Initialiser toutes les lampes à 0 (éteintes)
    for switch in range(1, n+1):
        for i in range(switch - 1, 100, switch):
            L[i] = 1 - L[i]  # Inverser l'état de la lampe
    return L
```

programme 3

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
@author: patrickjanc
"""
import numpy as np
import matplotlib.pyplot as plt

def lights(n):
    L = [0] * 100  # Initialiser toutes les lampes à 0 (éteintes)

    for switch in range(1, n+1):
        for i in range(switch - 1, 100, switch):
            L[i] = 1 - L[i]  # Inverser l'état de la lampe
    T = np.array(L).reshape((10, 10))
    return T


n = int(input("Entrer le nombre de commutations : "))
result = lights(n)
plt.imshow(result, cmap='gray', interpolation='nearest')
plt.title(f"Problème des 100 lampes - {n} commutations")
plt.show()
```

Programme 4

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
@author: patrickjanc
"""
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation

def lights(n):
    L = [0] * 100  # Initialiser toutes les lampes à 0 (éteintes)

    for switch in range(1, n+1):
        for i in range(switch - 1, 100, switch):
            L[i] = 1 - L[i]  # Inverser l'état de la lampe
    T = np.array(L).reshape((10, 10))
    return T

# Fonction pour mettre à jour l'animation
def update(frame):
    ax.clear()
    result = lights(frame)
    ax.imshow(result, cmap='gray', interpolation='nearest')
    ax.set_title(f"État des lampes après {frame} basculements")

# Créer la figure et l'axe
fig, ax = plt.subplots()

# Définir le nombre total de basculements (n)
total_basculements = 100

# Créer l'animation
animation = FuncAnimation(fig, update,
frames=range(1, total_basculements + 1), interval=500, repeat=False)
animation.save('lampes.gif', writer='pillow')
# Afficher l'animation
plt.show()
```