Application : base de donnée Internation Stroke Trials

In [12]:

```
import pandas as pd
```

In [13]:

```
#Traitement des données

df = pd.read_csv('ist.csv', low_memory=False)
df = df[['SEX', 'AGE','RXASP', 'RXHEP', 'DDEAD']] #on sélectionne les colonnes q
ui nous sont utiles
df['RXHEP'] = df['RXHEP'].replace(['H'], 'M') #la dose "high" d'héparine corresp
ond en fait à une dose "medium"
df = df.replace(np.nan, 'N', regex=True)
```

In [14]:

```
#Création du problème de bandit associé

#pour chacun des 6 traitements administrés dans l'essai clinique,
#on calcule la probabilité observée de guérison.
#cela va nous permettre de construire un bandit à 6 bras.

doses_aspirine = ["Y", "N"]
doses_heparine = ["M", "L", "N"]

p = {} #associe à chaque action sa probabilité q(a)

for dose_a in doses_aspirine:
    for dose_h in doses_heparine:
        #ombre de patients ayant reçu comme traitement dose_a et dose_h
        nb_patients = df.loc[(df['RXASP'] == dose_a) & (df['RXHEP'] == dose_h)].
shape[0]

        #nombre de patients ayant reçu dose_a et dose_h et ayant survécu à J+14
        nb_vivants = df.loc[(df['RXASP'] == dose_a) & (df['RXHEP'] == dose_h) &
(df['DDEAD'] == "N")].shape[0]

        #on calcule la probabilité q(a) observée pour ce traitement
        p[dose_a+dose_h] = nb_vivants/nb_patients

p
```

Out[14]:

```
{'YM': 0.8893004115226337,
 'YL': 0.907483552631579,
 'YN': 0.8962536023054755,
 'NM': 0.8928276999175597,
 'NL': 0.892136681762042,
 'NN': 0.8930041152263375}
```

In [15]:

```
actions_q = list(p.values())
```

In [16]:

```
H = 20000

actions_Q = [0, 0, 0, 0, 0, 0]
actions_U = [0, 0, 0, 0, 0, 0]
actions_N = [1, 1, 1, 1, 1, 1]

actions_compteurs = [[0], [0], [0], [0], [0], [0]]
```

In [17]:

```
for t in range(1, H):
    for a in range(len(actions_q)):
        actions_U[a] = np.sqrt(2 * np.log(t) / actions_N[a])

    a = indice_max(ajouter_listes(actions_Q, actions_U))

    actions_N[a] += 1
    r = generer_reponse(actions_q, a)
    actions_Q[a] = actions_Q[a] + (1/actions_N[a]) * (r - actions_Q[a])

    actions_compteurs[a].append(actions_compteurs[a][-1] + 1)
    for b in range(len(actions_q)):
        if b != a:
            actions_compteurs[b].append(actions_compteurs[b][-1])
```

In [18]:

```
actions_N
```

Out[18]:

```
[2614, 4906, 3015, 3252, 3165, 3053]
```

In [19]:

```
plt.plot(actions_compteurs[0])
plt.plot(actions_compteurs[1], label='YL')
plt.plot(actions_compteurs[2])
plt.plot(actions_compteurs[3])
plt.plot(actions_compteurs[4])
plt.plot(actions_compteurs[5])

plt.xlabel('temps')
plt.ylabel('actions cumulées')
plt.legend()

plt.savefig('ucb_ist.png', dpi=300)
```