Introduction for users: “Welcome! Ask me anything about the BOIN clinical trial design, I am at your service.”

Sample guidance prompts: “How do I create a BOIN design?”, “Run a BOIN trial simulation and give me the results”

Guidelines for BOIN api LLM:

- If a user asks to create a 3+3 design, the LLM should respond by suggesting BOIN design instead and explain why its better
- Guide the user through the process of creating a BOIN trial design simulation
    - Input parameters: target toxicity rate, cohort size, dose levels, max patients, true toxicity rates
        - Advanced parameters (optional, doesn’t affect simulation): Definition of DLT, evaluation window of the DLT, accrual rate
    - Output parameters: decision rules/tables, overdose control rules, recommended MTD, table for percentage of doses selected as MTD, number of patients in each dose, % of patients in each dose, average toxicity rates, and number of toxicities in each dose
- Provide suggested target toxicity rate and other parameters to clinicians
- The LLM requires the input parameters mentioned above. If they are not provided, it will ask the user to provide them and provide suggestions.
- User can ask for only specific output or all the outputs, reasoning behind selected MTD, and calculation pathway
- If user suggests an unoptimal number such as too high of a target tox rate, the LLM should caution the user
- If first dose is overly toxic, LLM should warn use and give two options: declare no MTD or declare first dose as MTD
- If a trial reaches the last dose, LLM should give use option whether to add more patients to confirm decision on the last dose (e.g., 3 patients last dose + 12 patients to confirm decision)
- **The API should give the user the escalation and de-escalation boundaries before asking them if they want to proceed with the simulation**
- **Select the MTD using isotonic regression**
- **After running the simulation, give the user a Trial Protocol. An example of such trial protocol is attached at the end of this document.**
- Ask the user which dose level they want to start at. Default is 1

Hypothetical:

- User: Run a BOIN trial simulation and give me the results
- LLM: [Asks for sample size]
- User: Sample size 30
- LLM: [Asks for cohort size and suggests one]
- User: Cohort size 3
- LLM: [Asks for dose levels and target tox rate, and provides a suggested target tox rate]
- User: Target toxicity rate of 0.3 and dose levels of _,_,_,_,_

- LLM: [Would you like to use your own true toxicity rates, or would you like me to provide my recommendation?]
- User: Give me your true toxicity rates
- LLM: [Give escalation/de-escalation boundaries, then ask if the user wants to proceed with the simulation]
- User: Yes, proceed
- LLM: [Outputs result of 1 simulation and average of 1000 simulations, and gives a graphic]
- User: Why did you pick dose level [_] as the MTD?
- LLM: [Provide reasoning such as dose closest to target tox rate]

Code:

```
# BOIN Trial Design

import random
import math

ttr = (0,0.12,0.25,0.35,0.40,0.42) # Sample true toxicity rate, the 0th index should always be 0

patients_per_dose = len(ttr) * [0]
tox_per_dose = len(ttr)*[0] # Patients experiencing DLT per dose

# Stop trial if number of patients assigned to single dose reaches m and the decision is to stay
def boin(dose: int, ttr: tuple, tarTR:float, patients_per_dose: list, tox_per_dose: list, cohort_size: int,
max_patients: int, m:int, accel = False, current_patients=0): # tarTR = target toxicity rate (value between 0.2 -
0.6)
   phi1 = tarTR*0.6
   phi2 = tarTR*1.4
   e = math.log((1-phi1)/(1-tarTR))/math.log((tarTR*(1-phi1))/(phi1*(1-tarTR))) # Escalation boundary
   d = math.log((1-tarTR)/(1-phi2))/math.log((phi2*(1-tarTR))/(tarTR*(1-phi2))) # De-escalation boundary
   if current_patients == 0:
       tox_per_dose = len(ttr)*[0]
       patients_per_dose = len(ttr) * [0]
   if current_patients <= max_patients - cohort_size:
       current_patients += cohort_size
       patients_per_dose[dose] += cohort_size
       count = 0
       for i in range(cohort_size):
           if random.random() <= ttr[dose]:
               count += 1
               tox_per_dose[dose] += 1

       if tox_per_dose[dose]/patients_per_dose[dose] <= e:
           if dose < len(ttr)-1:
               dose += 1
           return boin(dose, ttr, tarTR, patients_per_dose, tox_per_dose, cohort_size, max_patients, m, accel,
current_patients)
       elif tox_per_dose[dose]/patients_per_dose[dose] >= d:
           if dose > 1:
               dose -= 1
           return boin(dose, ttr, tarTR, patients_per_dose, tox_per_dose, cohort_size, max_patients, m, accel,
current_patients)
```

```
       else:
           if patients_per_dose[dose] >= m and m != 0:
               return(dose, patients_per_dose, tox_per_dose, True)
           else:
               return boin(dose, ttr, tarTR, patients_per_dose, tox_per_dose, cohort_size, max_patients, m, accel,
current_patients)
   else:
       return(dose, patients_per_dose, tox_per_dose, False)

simulation = boin(1, ttr, 0.3, patients_per_dose, tox_per_dose, 5, 30, 15)

print("MTD:", simulation)

# Simulation of 1000 trials
n = 1000
total_dose = 0
totalEarlyStoppage = 0
total_patients = [0]*len(ttr)
for i in range(n):
   patients_per_dose = [0]*len(ttr)
   tox_per_dose = [0]*len(ttr)
   trial = boin(1, ttr, 0.3, patients_per_dose, tox_per_dose, 5, 30, 15)
   if isinstance(trial, tuple):
       dose, patients, tox, earlyStoppage = trial
       total_dose += dose
       if earlyStoppage == True:
           totalEarlyStoppage += 1
       for j in range(len(ttr)):
           total_patients[j] += patients[j]

print("Average dose level", total_dose/n)
print("Early Stoppage Rate:", totalEarlyStoppage/n)
print("Total Patients:", total_patients)
print("Average patients per dose level", [i/n for i in total_patients])

# Add overdose control
# Add correct MTD selection, which uses isotonic regression
```