```
"""
Core Idea

Call init to get a session_id and start_clock

LOOP:
Start iterating from start_clock and increment the clock by 1:
    Get new jobs from the /job endpoint
    /job will return a list of jobs that are available at the current
clock time
       - place new in new queue
       - 1 tick later move to ready queue

    - For jobs on a cpu decrement burst time for running CPU job
        - If a jobs burst time gets to 0, move to wait queue
        - get jobs next burst from the /burst endpoint
    - For jobs in the ready queue (jobs waiting for cpu) increment wait
time

    - For jobs using an IO device decrement burst time for that running IO
job
        - If a jobs burst time gets to 0, move to ready queue
        - get jobs next burst from the /burst endpoint
    - For jobs in the wait queue (waiting for IO device) increment wait
time

    - if burst is EXIT move job to terminated

"""

import requests
import json
import os
from rich import print
import random

def getConfig(client_id):
    return {
        "client_id": client_id,
        "min_jobs": 5,
        "max_jobs": 20,
        "min_bursts": 10,
        "max_bursts": 50,
        "min_job_interval": 1,
        "max_job_interval": 5,
        "burst_type_ratio": 0.5,
        "min_cpu_burst_interval": 15,
        "max_cpu_burst_interval": 50,
        "min_io_burst_interval": 30,
        "max_io_burst_interval": 70,
        "min_ts_interval": 5,
        "max_ts_interval": 5,
```

```
        "prioritys": [1,2,3,4,5]
    }

def init(config):
    """
    Description:
        This function will initialize the client and return the
`session_id` (next integer used by your client_id) and `clock_start`
    Args:
        config (dict): A dictionary containing the configuration for the
client
    Returns:
        dict: A dictionary containing the session_id and clock_start
    """
    route = f"http://profgriffin.com:8000/init"
    r = requests.post(route,json=config)
    if r.status_code == 200:
        response = r.json()
        return response
    else:
        print(f"Error: {r.status_code}")
        return None


def getJob(client_id,session_id,clock_time):
    """
    Description:
        This function will get the jobs available at the current clock
time

    Args:
        client_id (str): The client_id
        session_id (int): The session_id
        clock_time (int): The current clock time
    Returns:
        dict: A dictionary containing the jobs available at the current
clock time

    Example Response:
        "data": [
            {
            "job_id": 1,
            "session_id": 13,
            "arrival_time": 1989,
            "priority": 2
            }
        ]
    """
    route = f"http://profgriffin.com:8000/job?client_id=
{client_id}&session_id={session_id}&clock_time={clock_time}"
    r = requests.get(route)
    if r.status_code == 200:
        response = r.json()
        return response
```

```
    else:
        print(f"Error: {r.status_code}")
        return None

def getBurst(client_id, session_id, job_id):
    """
    Description:
        This function will get the burst for a job
    Args:
        client_id (str): The client_id
        session_id (int): The session_id
        job_id (int): The job_id
    Returns:
        dict: A dictionary containing the burst for the job
    Example Response:
        "data": {
            "burst_id": 1,
            "burst_type": "CPU",  # CPU, IO, EXIT
            "duration": 11
        }
    """
    route = f"http://profgriffin.com:8000/burst?client_id=
{client_id}&session_id={session_id}&job_id={job_id}"
    r = requests.get(route)
    if r.status_code == 200:
        response = r.json()
        return response
    else:
        print(f"Error: {r.status_code}")
        return None

def getBurstsLeft(client_id, session_id, job_id):
    """
    Description:
        This function will get the number of bursts left for a job
    Args:
        client_id (str): The client_id
        session_id (int): The session_id
        job_id (int): The job_id
    Returns:
        int: Simply an integer with count of bursts left zero otherwise
    Example Response:
        3
    """
    route = f"http://profgriffin.com:8000/burstsLeft?client_id=
{client_id}&session_id={session_id}&job_id={job_id}"
    r = requests.get(route)
    if r.status_code == 200:
        response = r.json()
        return response
    else:
        print(f"Error: {r.status_code}")
        return None
```

```
def getJobsLeft(client_id, session_id):
    """
    Description:
        This function will get the number of jobs left for a session
    Args:
        client_id (str): The client_id
        session_id (int): The session_id
    Returns:
        int: Simply an integer with count of jobs left zero otherwise
    Example Response:
        11
    """
    route = f"http://profgriffin.com:8000/jobsLeft?client_id=
{client_id}&session_id={session_id}"
    r = requests.get(route)
    if r.status_code == 200:
        response = r.json()
        return response
    else:
        print(f"Error: {r.status_code}")
        return None


if __name__ == '__main__':
    do_init = False;
    do_job = False;
    do_burst = False;

    jobs = {}

    client_id = "sgtrock"
    config = getConfig(client_id)
    base_url = 'http://profgriffin.com:8000/'
    response = init(config)

    start_clock = response['start_clock']
    session_id = response['session_id']

    clock = start_clock

    while(clock):
        #print(f"Clock: {clock}")
        jobsLeft = getJobsLeft(client_id, session_id)
        if not jobsLeft:
            break
        response = getJob(client_id,session_id,clock)
        if response and response['success']:
            if response['data']:
                for data in response['data']:
                    job_id = data['job_id']
                    print(f"Job {job_id} received at {clock}...")
                    if job_id not in jobs:
                        jobs[job_id] = {'data':data,'bursts':{}}
```

```
        print(jobs)

        for job_id in jobs:
            #print(f"cid: {client_id}, sid: {session_id}, jid: {job_id}")
            burstsLeft = getBurstsLeft(client_id, session_id, job_id)
            if not burstsLeft:
                print(f"No bursts left for job {job_id} at {clock}")
                continue
            bresp = getBurst(client_id, session_id, job_id)
            if isinstance(bresp, dict) and 'success' in bresp and
bresp['success']:
                burst = bresp['data']
                bid = burst['burst_id']
                print(f"Burst {bid} received ...")
                jobs[job_id]['bursts'][bid] = burst


        clock += 1
```