Updated on 2023/5/18

RTS Homework Requirements

Story

In game programming, performance is key. Game logic typically includes an update loop, failing to meet deadline in the update loop will result in perceivable lag, this is an example of **Periodic Task.** Some modern games incrementally save your progress without the need of explicit savepoints, this can be considered an example of **Aperiodic Task.** When a user clicks a button, they expect the game to complete the respective action within a short deadline, this is an example of **Sporadic Task.** You are a group of programmers working on the game engine Unidoteal, to improve the performance, you are tasked to prototype a new **Scheduler**, handling **TaskGroups** which consists of 3 scheduling class: **Periodic**, **Aperiodic**, **Sporadic.**

The input is given in the **JSON format**, it is recommended that you use a JSON parsing library to save time.

Input

The input consist of an vector of **TaskGroups**, each TaskGroup is an instance of tasks you need to schedule. TaskGroup consist of vector of **Periodic**, **Aperiodic**, **Sporadic tasks. Periodic tasks** consist of Period **P** and Cost **C**(Worst Case Execution Time), for simplicity, as long as periodic tasks complete once per period, it is fine. **Aperiodic tasks**, **Sporadic tasks** consist of arrival time **A** and Cost **C**(Worst Case Execution Time), the deadline D is equal to A+C. These can only be scheduled after A, note that Periodic and Sporadic tasks have hard deadlines and thus have greater priority over Aperiodic tasks(soft deadlines).
All time units are of type **Int64.** The total Utilization **U** is **100**, that is, you only need to schedule **from 0 to 100.**

Note:
Priority within the same kind of task is given by its order (First element highest, Last lowest), reject task if it isn’t feasible to schedule within U = 100
Reject Aperiodic, Sporadic tasks that cannot be scheduled

Example:

```
[
    {
        "Periodic": [
            {
                "P": 50,
                "C": 2
            },
            {
                "P": 33,
                "C": 200
            }
        ],
        "Aperiodic": [
            {
                "A": 5,
                "C": 1
            },
            {
                "A": 10,
                “C": 20
            }
        ],
        "Sporadic": [
            {
                "A": 6,
                "C": 19
            },
            {
                "A": 70,
                "C": 25
            }
        ]
    }
]
```

Output

You will need to print the schedule your scheduler produces in the following format

```
#Schedules: number of Schedules, same as length of input vector
#TaskGroup: index of TaskGroup, starting from 0
Schedule for that TaskGroup:
     Task Info: consist of Type # Start End Status
     Type: P, A ,S denoting Periodic, Aperiodic, Sporadic
respectively
```

```
      # : index of task of a specific type denoted, starting from 0
      Start: integer denoting when task start to run
      End: integer denoting when task completes running
      Status: Complete or Reject
Statistics: Rejection Rate of P, A ,S in Float64
-1 at the end of Output
```

Note:
Start and End is -1 if reject
Start time should be monotonous increasing

Example:

```
1
0
P 1 -1 -1 Reject
A 1 -1 -1 Reject
P 0 0 2 Complete
A 0 5 6 Complete
S 0 6 25 Complete
S 1 70 95 Complete
0.5 0.5 0.0
-1
```

Grading

```
10% Report(PDF):
      2% Group Info
      2% Document your progress, thought process
      2% Explain the logic of your scheduler
      2% Things you learned
      2% Correct Format
20% Demo:
      15% Presentation Clarity
      5% Followup Questions
70% Code
      10% Functional:
            5% compiles/run without problem(MakeFile,Dependency,…etc)
            5% standard conforming
            (no implementation defined, undefined behavior)
      15% Accept/Reject tasks based on feasibility:
            Scheduler outputs feasible Schedule
            Execute Tasks when possible
      15% Correct Format
      30% Schedule Tasks:
            15% Check Feasibility
            15% Valid Schedule
```

10% Bonus(something special you did)

Clarifications

Added in 5/18

Some of you aren’t familiar with the JSON format input, so here is a more general example.

Also, do consider using a library to save your time.

Another Input Example:

```
[
    {
        "Periodic": [
            {
                "P": 4,
                "C": 5
            }
        ],
        "Aperiodic": [
            {
                "A": 90,
                "C": 5
            },
            {
                "A": 19,
                "C": 3
            },
            {
                "A": 96,
                "C": 3
            },
            {
                "A": 1,
                "C": 1
            },
            {
                "A": 92,
                "C": 1
            },
            {
                "A": 3,
                "C": 4
            }
        ],
        "Sporadic": [
            {
```

```
                "A": 97,
                "C": 1
            },
            {
                "A": 45,
                "C": 3
            },
            {
                "A": 6,
                "C": 6
            },
            {
                "A": 52,
                "C": 1
            },
            {
                "A": 87,
                "C": 7
            },
            {
                "A": 39,
                "C": 9
            },
            {
                "A": 77,
                "C": 1
            }
        ]
    },
    {
        "Periodic": [
            {
                "P": 2,
                "C": 10
            },
            {
                "P": 2,
                "C": 4
            },
            {
                "P": 3,
                "C": 2
            }
        ],
        "Aperiodic": [
            {
                "A": 25,
                "C": 4
            },
```

```
        {
            "A": 77,
            "C": 9
        },
        {
            "A": 12,
            "C": 6
        },
        {
            "A": 75,
            "C": 8
        },
        {
            "A": 51,
            "C": 3
        },
        {
            "A": 90,
            "C": 5
        }
    ],
    "Sporadic": [
        {
            "A": 45,
            "C": 2
        },
        {
            "A": 69,
            "C": 1
        },
        {
            "A": 18,
            "C": 2
        },
        {
            "A": 96,
            "C": 7
        },
        {
            "A": 31,
            "C": 2
        },
        {
            "A": 94,
            "C": 3
        },
        {
            "A": 95,
            "C": 3
```

```
            },
            {
                "A": 66,
                "C": 3
            },
            {
                "A": 3,
                "C": 6
            },
            {
                "A": 14,
                "C": 3
            }
        ]
    }
]
```