HW 1

Real-Time Systems Task Generation Program

2020310083 손형준

Instruction Overview

Assignment Requirements:

Create a Python program that generates 100 sets of real-time tasks with utilization values.

Takes three command-line arguments:

- **n**: Number of tasks
- **U**: Utilization of the task set (<1)
- **v**: Deadline type (0 for implicit, 1 for constrained)
- **Output**: Task sets saved in a specified format in the `./output` directory.

Input Validation Process

Function:

`validate_user_input()`

Purpose:

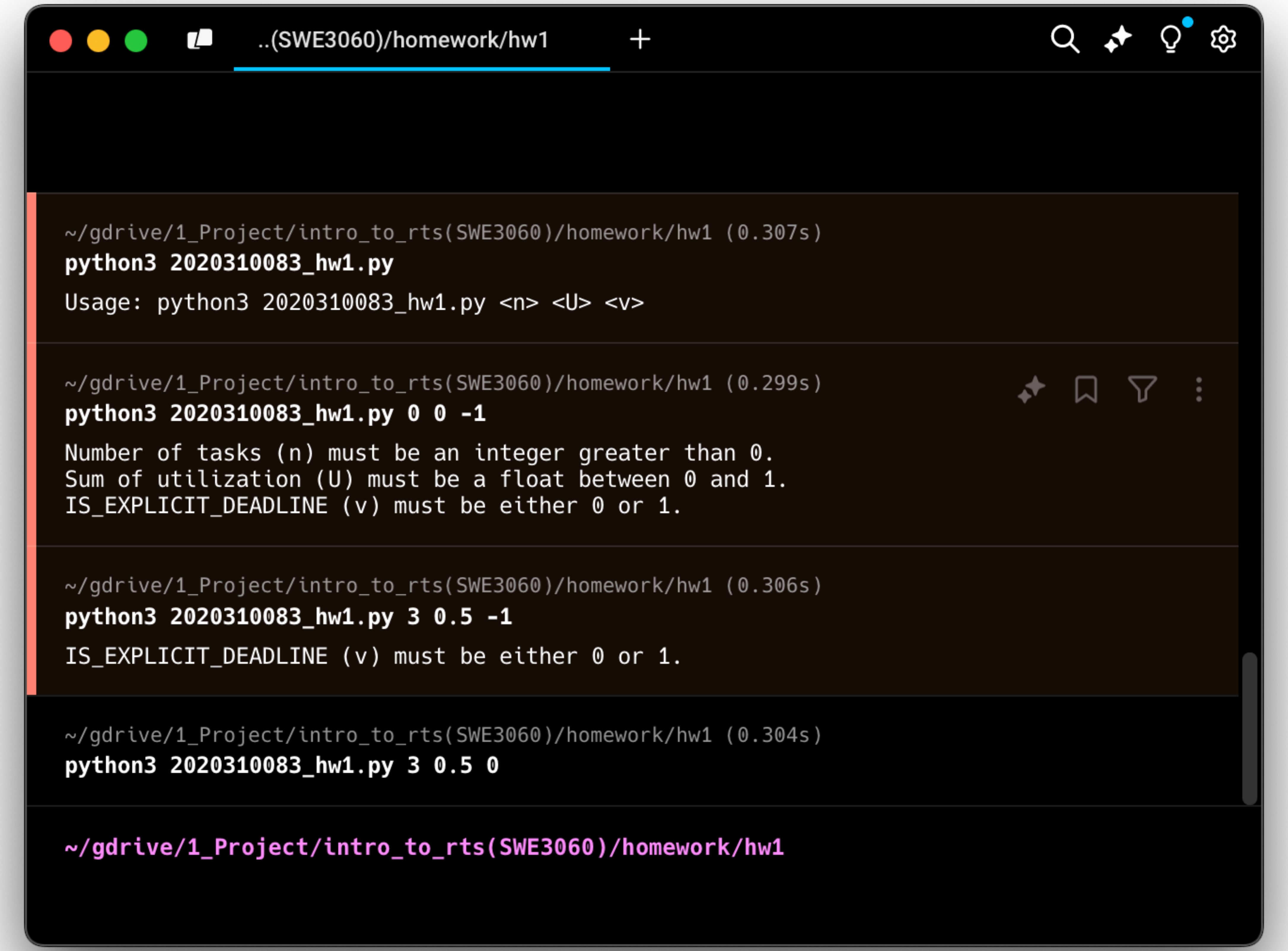
Ensures valid input for n, U, and v based on specific conditions:

- **n** > 0 and integer
- **U** is a float between 0 and 1
- **v** is either 0 or 1

Error Handling:

Any invalid input results in printed error messages, and the program exits.

- If the particular error occurred, append the error message to `error_msg array`
- If `error_msg array` is not empty, print all error messages and exit



```
~/gdrive/1_Project/intro_to_rts(SWE3060)/homework/hw1 (0.307s)
python3 2020310083_hw1.py
Usage: python3 2020310083_hw1.py <n> <U> <v>

~/gdrive/1_Project/intro_to_rts(SWE3060)/homework/hw1 (0.299s)
python3 2020310083_hw1.py 0 0 -1
Number of tasks (n) must be an integer greater than 0.
Sum of utilization (U) must be a float between 0 and 1.
IS_EXPLICIT_DEADLINE (v) must be either 0 or 1.

~/gdrive/1_Project/intro_to_rts(SWE3060)/homework/hw1 (0.306s)
python3 2020310083_hw1.py 3 0.5 -1
IS_EXPLICIT_DEADLINE (v) must be either 0 or 1.

~/gdrive/1_Project/intro_to_rts(SWE3060)/homework/hw1 (0.304s)
python3 2020310083_hw1.py 3 0.5 0

~/gdrive/1_Project/intro_to_rts(SWE3060)/homework/hw1
```

UUniFast Algorithm

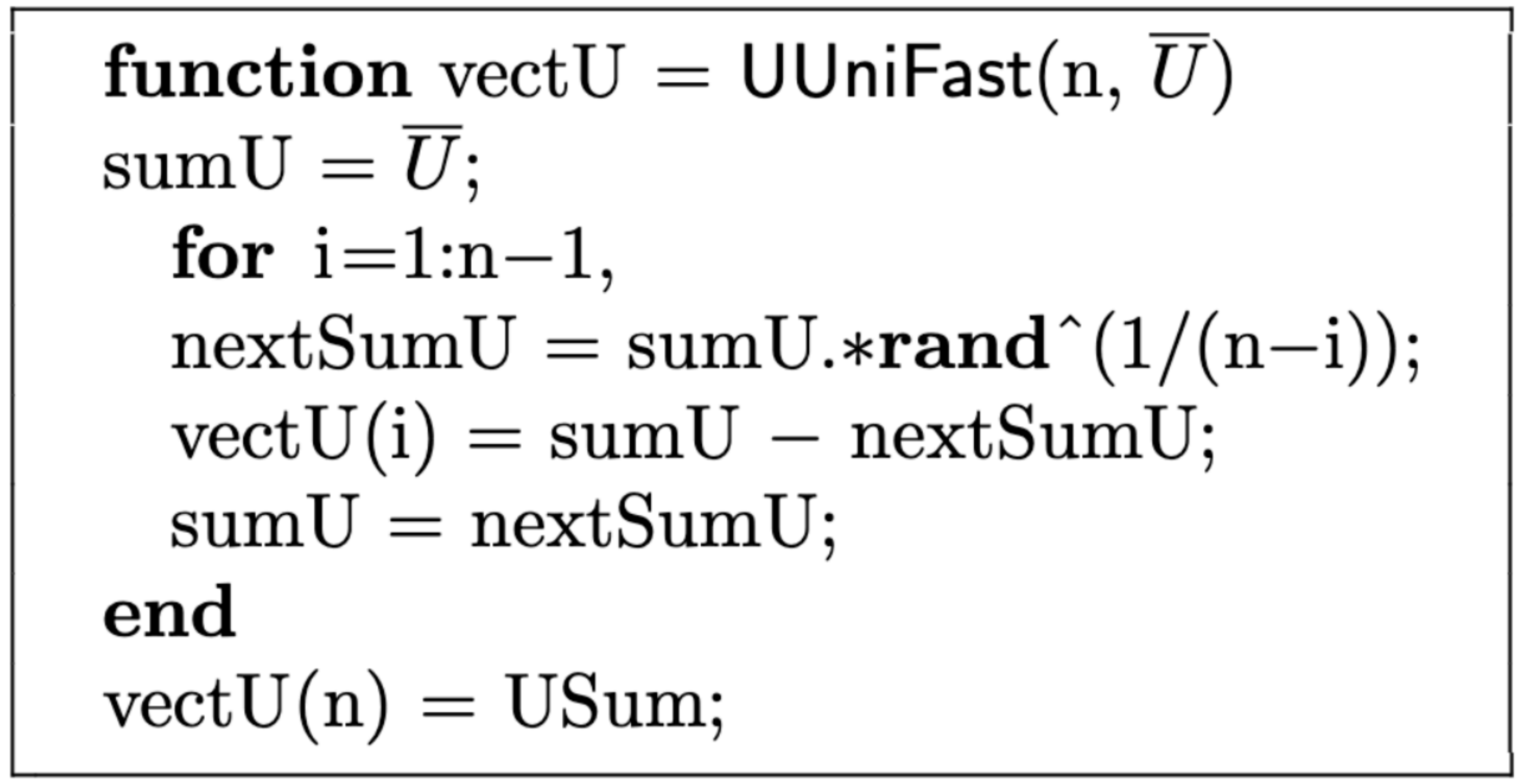
E. Bini and G. Buttazzo. 2005. Measuring the performance of schedulability tests

Purpose:

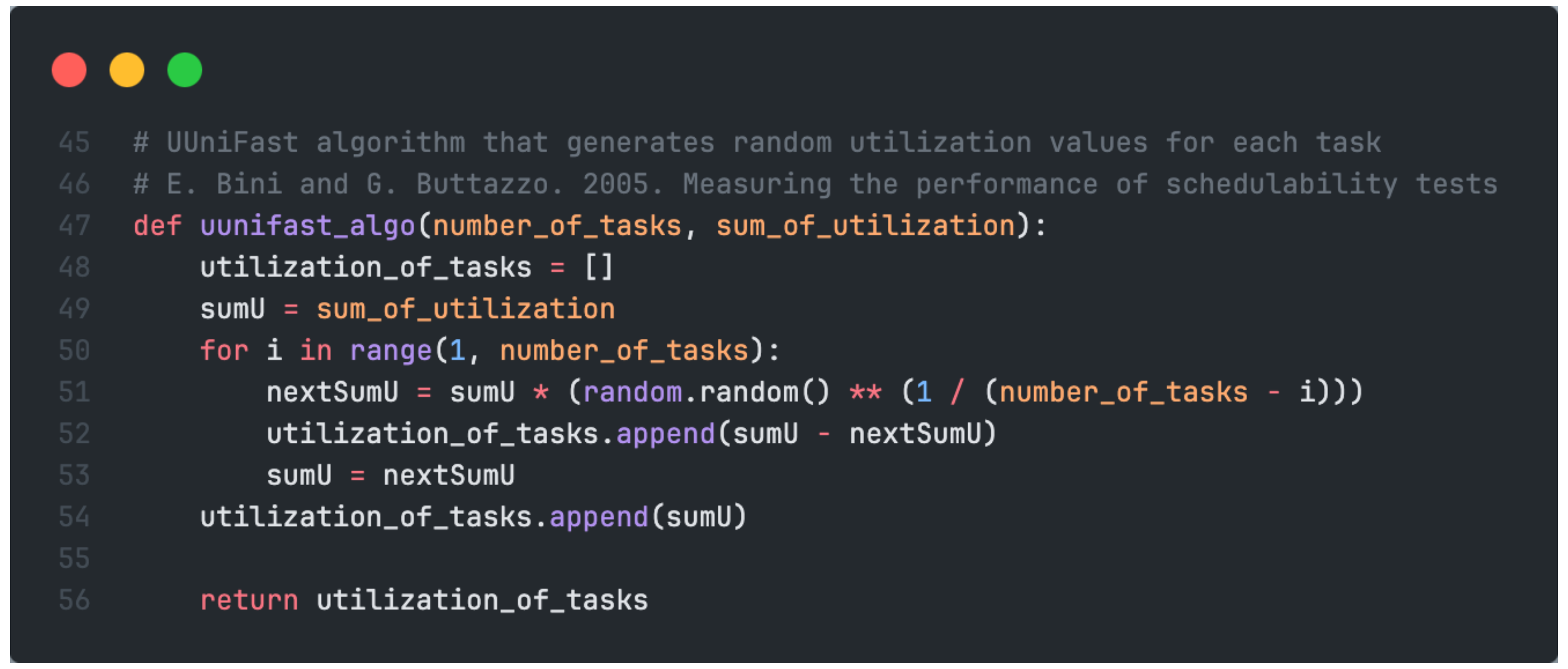
Generates random utilization values that sum up to U.

Steps:

1. Initialize sumU as the target utilization (U).
2. Iteratively divide sumU to allocate portions to each task.
3. Ensures total utilization constraint is met accurately.



```
**function** vectU = UUniFast(n, $\overline{U}$)
sumU = $\overline{U}$;
   **for** i=1:n−1,
   nextSumU = sumU.∗**rand**ˆ(1/(n−i));
   vectU(i) = sumU − nextSumU;
   sumU = nextSumU;
**end**
vectU(n) = USum;
```



```python
# UUniFast algorithm that generates random utilization values for each task
# E. Bini and G. Buttazzo. 2005. Measuring the performance of schedulability tests
def uunifast_algo(number_of_tasks, sum_of_utilization):
    utilization_of_tasks = []
    sumU = sum_of_utilization
    for i in range(1, number_of_tasks):
        nextSumU = sumU * (random.random() ** (1 / (number_of_tasks - i)))
        utilization_of_tasks.append(sumU - nextSumU)
        sumU = nextSumU
    utilization_of_tasks.append(sumU)

    return utilization_of_tasks
```

Task Generation Methodology

Function:

`generate_tasks()`

Inputs:

Number of tasks (n), utilization (U), deadline type (v)

Output:

List of tasks with randomly generated parameters:

- **Period (Ti)**: Random integer between 100 and 1000.
- **WCET (Ci)**: Calculated using Ti and utilization value that got from **`uunifast_algo`**

  (rounded up and ensure a minimum value of 1).
- **Deadline (Di)**: Set to Ti **if v=0 or a random integer between Ci and Ti if v=1**.

Writing Output to File

Function:

`main()`

Directory and File Creation:

- Creates output directory if it doesn't exist.

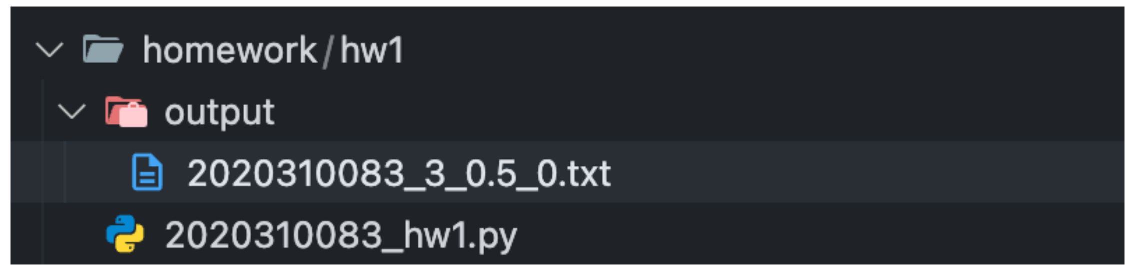
  (**`exist_ok=True`**)

- Constructs filename using format **`2020310083_{n}_{U}_{v}.txt`**.

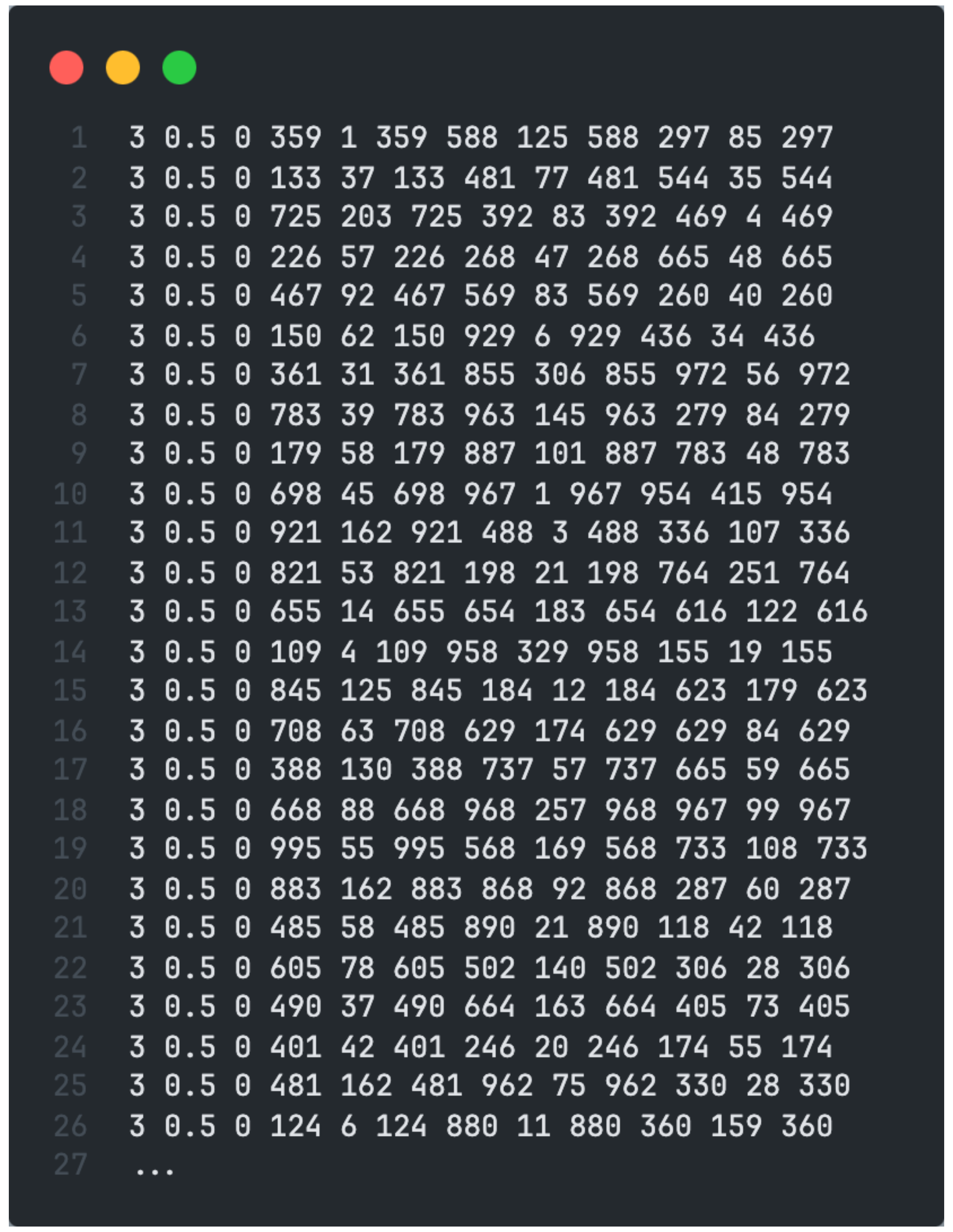
  (**`file mode`** to **`'w'`** for overwriting file when same parameter inputs are given)

File Output:

Each line represents a task set: **`{n} {U} {v} T1 C1 D1 T2 C2 D2 … Tn Cn Dn`** for each task.



```
homework / hw1
  output
    2020310083_3_0.5_0.txt
  2020310083_hw1.py
```



```
3 0.5 0 359 1 359 588 125 588 297 85 297
3 0.5 0 133 37 133 481 77 481 544 35 544
3 0.5 0 725 203 725 392 83 392 469 4 469
3 0.5 0 226 57 226 268 47 268 665 48 665
3 0.5 0 467 92 467 569 83 569 260 40 260
3 0.5 0 150 62 150 929 6 929 436 34 436
3 0.5 0 361 31 361 855 306 855 972 56 972
3 0.5 0 783 39 783 963 145 963 279 84 279
3 0.5 0 179 58 179 887 101 887 783 48 783
3 0.5 0 698 45 698 967 1 967 954 415 954
3 0.5 0 921 162 921 488 3 488 336 107 336
3 0.5 0 821 53 821 198 21 198 764 251 764
3 0.5 0 655 14 655 654 183 654 616 122 616
3 0.5 0 109 4 109 958 329 958 155 19 155
3 0.5 0 845 125 845 184 12 184 623 179 623
3 0.5 0 708 63 708 629 174 629 629 84 629
3 0.5 0 388 130 388 737 57 737 665 59 665
3 0.5 0 668 88 668 968 257 968 967 99 967
3 0.5 0 995 55 995 568 169 568 733 108 733
3 0.5 0 883 162 883 868 92 868 287 60 287
3 0.5 0 485 58 485 890 21 890 118 42 118
3 0.5 0 605 78 605 502 140 502 306 28 306
3 0.5 0 490 37 490 664 163 664 405 73 405
3 0.5 0 401 42 401 246 20 246 174 55 174
3 0.5 0 481 162 481 962 75 962 330 28 330
3 0.5 0 124 6 124 880 11 880 360 159 360
...
```