Waterloo Engineering Competition

Programming Challenge

Program Design

1. Read in data from CSV files

```python
import csv

def indexFunc(e):
    return e[2]

# read in all of the server data from the server csv
servers = []
tot_params = [0, 0] # cores, ram
with open('./servers1.csv') as f:
    data = csv.DictReader(f)
    for line in data:
        servers.append([int(line['Sever Number']), int(line['Number of Cores']), int(line['Number of Watts']), int(line['Total RAM']), []])
        if(int(line['Number of Cores']) > max_params[0]):
            max_params[0] = int(line['Number of Cores'])
        if(int(line['Total RAM']) > max_params[1]):
            max_params[1] = int(line['Total RAM'])
        tot_params[0] += int(line['Number of Cores'])
        tot_params[1] += int(line['Total RAM'])
servers.sort(key=indexFunc)

# read in all of the tasks from the task csv
all_tasks = []
tot_task_params = [0, 0]
with open('./tasks1.csv') as f:
    data = csv.DictReader(f)
    for line in data:
        all_tasks.append([int(line['Number of Cores']), int(line['Number of Turns']), int(line['RAM']), int(line['Complete in Turns']), 0, 0, 0, 0])
        tot_task_params[0] += int(line['Number of Cores'])
        tot_task_params[1] += int(line['RAM'])
```

Ideal server

```
# preprocessing
# start deciding which server will run each task: if there is a task that can only possibly be completed by one server, assign that server to the task
for task in all_tasks:
    for i in range(0, len(servers)):
        if(servers[i][1] >= task[0] and servers[i][3] >= task[2]):
            task[6] = i
            break
        else:
            task[6] = -1
```

Code File Edit Selection View Go Run Terminal Window Help Sun Nov 24 3:26 PM

wec_curr

simple program to read from csv Untitled-1 | driver.py U | output.csv U | tasks1.csv U | servers1.csv U

driver.py > ...

```python
counter = 0 # counter of finished tasks (both completed or failed)
turn = 0 #counter to go through tasks array
actualturn = 1 # turn number
whoswhere = [len(servers)]
print("loop start")
curr = [] #array of tasks currently on servers
while(counter<len(all_tasks)):
    if(all_tasks[turn][6]<0): # does not have an ideal server (RAM exceeds max server RAM)
        remove = [0,0,0,0,0]
        remove[0] = actualturn
        remove [1] = i
        remove[3] = servers[whoswhere[i]][2] * all_tasks[turn][1]
        with open("output.csv", mode='a', newline='') as file:
            writer = csv.writer(file)
            writer.writerow(remove)
        counter+=1
    else: # has an ideal server that exists
        curr.append(all_tasks[turn]) # add to array of tasks currently running
        for i in range(all_tasks[turn][6], len(servers)):#adding to a server
            if((servers[i][3]-servers[i][4])>all_tasks[turn][2]): #check which is the most ideal server is that can hold the task based on available RAM
                servers[i][4] += all_tasks[turn][2]
                #whoswhere[i] = i
                break
            else: # no server available, failed
                remove = [0,0,0,0,0]
                remove[0] = actualturn
                remove [1] = i
                #remove[3] = servers[whoswhere[i]][2] * all_tasks[turn][1]
                with open("output.csv", mode='a', newline='') as file: # write to output.csv
                    writer = csv.writer(file)
                    writer.writerow(remove)
                counter+=1 # add one to counter of finished tasks
        min = curr[0][3]-curr[0][1] #setting minimum difference between Complete in Turns and Number of Turns to first task
        min_index = 0 # current index 0
        for i in range(len(curr)): # checking for minimum difference
            if (len(curr)==1):
                min_index = i # if only one task currently, min_index is 0
            elif((curr[i][3] - curr[i][1])<min):
                min_index = i # replace min index if found a smaller difference
                min = curr[i][3] - curr[i][1]
        curr[min_index][4]-=1 # delete from turn of task with the minimum difference
        numdel = 0 # if a number is deleted (fixes an error)
        for i in range(len(curr)):
            if(curr[i-numdel][4]==0):#completed, turn is 0
                done = [0,0,1,0,0]
                done[0] = actualturn
```

main* ⊗ 0 ⚠ 0 0 Ln 55, Col 67 Tab Size: 4 UTF-8 LF {} Python 3.9.6 64-bit

Code File Edit Selection View Go Run Terminal Window Help Sun Nov 24 3:26 PM

wec_curr

simple program to read from csv Untitled-1 | driver.py U | output.csv U | tasks1.csv U | servers1.csv U

driver.py > ...

```python
counter = 0 # counter of finished tasks (both completed or failed)
turn = 0 #counter to go through tasks array
actualturn = 1 # turn number
whoswhere = [len(servers)]
print("loop start")
curr = [] #array of tasks currently on servers
while(counter<len(all_tasks)):
    if(all_tasks[turn][6]<0): # does not have an ideal server (RAM exceeds max server RAM)
        remove = [0,0,0,0,0]
        remove[0] = actualturn
        remove [1] = i
        remove[3] = servers[whoswhere[i]][2] * all_tasks[turn][1]
        with open("output.csv", mode='a', newline='') as file:
            writer = csv.writer(file)
            writer.writerow(remove)
        counter+=1
    else: # has an ideal server that exists
        curr.append(all_tasks[turn]) # add to array of tasks currently running
        for i in range(all_tasks[turn][6], len(servers)):#adding to a server
            if((servers[i][3]-servers[i][4])>all_tasks[turn][2]): #check which is the most ideal server is that can hold the task based on available RAM
                servers[i][4] += all_tasks[turn][2]
                #whoswhere[i] = i
                break
            else: # no server available, failed
                remove = [0,0,0,0,0]
                remove[0] = actualturn
                remove [1] = i
                #remove[3] = servers[whoswhere[i]][2] * all_tasks[turn][1]
                with open("output.csv", mode='a', newline='') as file: # write to output.csv
                    writer = csv.writer(file)
                    writer.writerow(remove)
                counter+=1 # add one to counter of finished tasks
        min = curr[0][3]-curr[0][1] #setting minimum difference between Complete in Turns and Number of Turns to first task
        min_index = 0 # current index 0
        for i in range(len(curr)): # checking for minimum difference
            if (len(curr)==1):
                min_index = i # if only one task currently, min_index is 0
            elif((curr[i][3] - curr[i][1])<min):
                min_index = i # replace min index if found a smaller difference
                min = curr[i][3] - curr[i][1]
        curr[min_index][4]-=1 # delete from turn of task with the minimum difference
        numdel = 0 # if a number is deleted (fixes an error)
        for i in range(len(curr)):
            if(curr[i-numdel][4]==0):#completed, turn is 0
                done = [0,0,1,0,0]
                done[0] = actualturn
```

main* ⊗ 0 ⚠ 0 0 Ln 55, Col 67 Tab Size: 4 UTF-8 LF {} Python 3.9.6 64-bit