**Group Assignment**

Group Members: Jahanzaib and Hakim

Course: Operating System and System Programming

**Round Robin Scheduling**

**General Introduction:**

CPU scheduling is a complete process which allows one process to use the CPU resources while the execution of another process is paused due to unavailability of resources or based on priority of processes. The main aim of CPU scheduling is to make the system faster and more efficient.

There are certain algorithms which are used to schedule the execution of processes, which are either preemptive or non-preemptive. Preemptive scheduling is based on priority, where a low priority process will be put in the execution queue when a high priority process enters the into ready state. On the other hand, non-preemptive works on the First Come First Out (FIFO) principle. Once a process starts, it will not stop until it finished in allotted time.

Round Robin is one of the Preemptive process scheduling. Each process is provided with a fixed time to execute, which is known as quantum. Once a process is executed for a given period of time, it is paused, and other process is executed for the same period of time. When it has gone through all the process, it will come back to the first process. Same cycle will continue until all the processes achieve their execution time.

**Analysis of the Algorithm:**

Round Robin Scheduling is one the mostly used CPU scheduling algorithm. It gives better result in comparison to other scheduling algorithms, but it also has some disadvantages, which we will discuss along with the pros below.

*Advantages*: Each process, in this scheduling algorithm, is provided with a fixed quantum time by the CPU, so all the processes have same priority. No process is left behind based on its priority. Secondly, each process gets a chance to run and CPU resources are shared between all process due to which no starvation occurs. Its implementation can be simple using FIFO queue system and processes with the same priority are handled better using this algorithm.

*Disadvantages*: Using this scheduling algorithm, low priority processes may execute first as this process doesn’t depends on priority level. Moreover, if time quantum is shorter than needed, then the number of times that CPU switches from one process to another increases, which will eventually decrease the efficiency of CPU. And, if quantum time is longer, then it will behave the same way as FIFO does. In addition, longer processes will have to wait more which will cause starvation.

**Conclusions and Suggestion to enhance the performance:**

Round Robin Algorithm is based on first come first out (FIFO) with time quantum, that is why the response time is better, but sometimes it is the quantum time which make this Algorithm better or opposite. In this Algorithm, same quantum time is given to both Input Output bound and CPU bound processes which seems to be unjust. That is why it is recommended to use a combination of both Round Robin and a priority-based scheduling for better performance.

**Source Code with documentation:**

from prettytable import PrettyTable

def round\_robin():

numb = int(input("Enter the number of processes: "))

quant = int(input("Enter the Quantum range: "))

a=[]

for i in range(numb):

b=[]

name = str(input("Enter process name: "))

burst = int(input("Enter the burst time: "))

b.append(name)

b.append(burst)

a.append(b)

#print(a)

bt=[]

prc=[]

fbt=[]

for i in range(len(a)):

bt.append(a[i][1])

fbt.append(a[i][1])

prc.append(a[i][0])

#print("burst time: ",bt)

#print("The processes: ", prc)

print("\n")

#to print the time line of the process

line=[]

add=0

while any(bt[i]!=0 for i in range(len(bt))):

for j in range(len(bt)):

if (bt[j]<quant and bt[j]!=0):

evry=[]

add+=bt[j]

evry.append(prc[j])

evry.append(add)

line.append(evry)

bt[j]-=bt[j]

elif bt[j]>=quant:

evry=[]

add+=quant

bt[j]-=quant

evry.append(prc[j])

evry.append(add)

line.append(evry)

print(line)

#to associate the processes and their turn around time

dit={}

for i in range(len(line)):

if line[i][0] in dit.keys():

dit[line[i][0]]=line[i][1]

elif line[i][0] not in dit.keys():

dit.update({line[i][0]:line[i][1]})

#print("processes and tat", dit)

#to find the turn adound time from the dictionary above

tat=[]

for i in dit.keys():

tat.append(dit[i])

#print('turn around time: ', tat)

#to find witing time (wt=tat-bt)

wt=[]

for i in range(len(fbt)):

wt.append(tat[i]-fbt[i])

#print("waiting time: ",wt)

#to present all the data in the form of a tabular-data

t = PrettyTable(['Process', 'Turnaround Time', "Waiting Time"])

for i in range(len(prc)):

t.add\_row([prc[i],tat[i],wt[i]])

print(t)

#average turn around time

sum1=0

for i in tat:

sum1+=i

h1=(sum1/len(tat))

print("Average turn around time: ", h1)

#average waiting time

sum2=0

for i in wt:

sum2+=i

h2=sum2/len(wt)

print("Average waiting time: ", h2)

round\_robin()