

### OS LAB-3

Q) Write a C program to simulate the following CPU scheduling algorithm to find turnaround time and waiting time. ☒ Priority (Non-pre-emptive) ☒ Round Robin (Experiment with different quantum sizes for RR algorithm)

CODE-

OS LAB-3 28/6/13

1. Write a C program to simulate the following CPU scheduling algorithm to find turnaround time and waiting time.

- 1) Priority (non-pre-emptive)
- 2) Round Robin

```
#include <stdio.h>
#include <conio.h>

int main()
{
    int n, i, t[100], p[100], at[100], j, temp;
    int ct[100], tat[100], sum=0, avgat=0, avgct=0;

    printf("Enter num of processes: ");
    scanf("%d", &n);

    for(i=0; i<n; i++)
    {
        printf("Enter arrival time for p[%d]: ", i+1);
        scanf("%d", &t[i]);
        printf("Enter burst time for p[%d]: ", i+1);
        scanf("%d", &ct[i]);
    }

    for(i=0; i<n-1; i++)
    {
        for(j=i+1; j<n; j++)
        {
            if (priority[j] < priority[i])
            {
                temp = priority[j];
                priority[j] = priority[i+1];
                priority[i+1] = temp;
            }
        }
    }
}
```

```
temp = b[rj];
b[rj] = b[rj+1];
b[rj+1] = temp;
```

```
temp = a[rj];
a[rj] = a[rj+1];
a[rj+1] = temp;
```

y

y

```
printf("Waiting time");
```

```
for (i=0; i<n; i++)
```

{

```
wt[i] = sum - a[i];
```

```
fat[i] = wt[i] + b[i];
```

```
printf("%d %d\n", wt[i], fat[i]);
```

```
avgwt += wt[i];
```

```
avgfat += fat[i];
```

```
sum += b[i];
```

y

```
float avgwtf = (float) avgwt / n;
```

```
float avgfat = (float) avgfat / n;
```

```
printf("Average waiting time: %.2f, avgwtf");
```

```
printf("Average Turnaround time: %.2f, avgfatf");
```

y



```
2. #include <stdio.h>
```

```
#include <stdlib.h>
```

```
int main()
```

```
{ int n, i, b[10], r[10], q[10] = 0, tot[10], cot[10], ct[10];
```

```
float atk = 0, and = 0;
```

```
printf("Enter the number of processes: ");
```

```
scanf("%d", &n);
```

```
printf("Enter the burst time of the processes: ");
```

```
for (i = 0; i < n; i++)
```

```
{
```

```
for (i = 0; i < n; i++) {
```

```
scanf("%d", &b[i]);
```

```
atk[i] = b[i];
```

```
}
```

```
printf("Enter the arrival time of the processes: ");
```

```
for (i = 0; i < n; i++) {
```

```
scanf("%d", &at[i]);
```

```
}
```

```
printf("Enter the time Quantum (n): ");
```

```
scanf("%d", &q);
```

```
while(1)
```

```
{ int done = 1;
```

```
for (i = 0; i < n; i++) {
```

```
if (r[i] > 0) {
```

```
done = 0;
```

```
if (r[i] > q)
```

```
{ r[i] = r[i] - q;
```

```
ct[i] = ct[i] + q;
```



## OUTPUT-

### Priority(non-pre emptive)

```
Enter number of processors: 4
Enter arrival time for p[1]: 0
Enter burst time for p[1]: 4
Enter priority of p[1]: 3
Enter arrival time for p[2]: 1
Enter burst time for p[2]: 3
Enter priority of p[2]: 4
Enter arrival time for p[3]: 2
Enter burst time for p[3]: 3
Enter priority of p[3]: 6
Enter arrival time for p[4]: 3
Enter burst time for p[4]: 5
Enter priority of p[4]: 5
waiting time
0
3
4
10

Total average waiting time: 4.250000
Total average turnaround time: 8.000000|
```

### Round robin

```
Enter the number of processes: 5
Enter the burst time of the processes: 5 3 1 2 3
Enter the arrival time of the processes: 0 1 2 3 4
Enter the time quantum: 2

Process Burst Time      Turnaround Time Waiting Time
1         5             14              9
2         3             11              8
3         1              3              2
4         2              4              2
5         3              9              6

Average Turnaround Time: 8.20
Average Waiting Time: 5.40

Process returned 0 (0x0)   execution time : 19.804 s
Press any key to continue.
```

