LAB ASSIGNMENT 11

NON PREEMPTIVE SCHEDULING ALGORITHMS

P.PRIYADHARSHINI

2019103562

19-04-2021

ALGORITHMS:

- FIRST COME FIRST SERVE(FCFS)
- SHORTEST JOB FIRST SCHEDULING ALGORITHM
- PRIORITY SCHEDULING ALGORITHM

1.FIRST COME FIRST SERVE SCHEDULING ALGORITHM(FCFS)

```
[s2019103562@centos8-linux Mon Apr 19 09:02 AM lab11]$ ./fcfs
Enter the number of processes:
3
Enter the arrival time of process 1:
0
Enter the burst time of process 1:
9
Enter the arrival time of process 2:
1
Enter the burst time of process 2:
4
Enter the arrival time of process 3:
2
Enter the burst time of process 3:
9
PROCESS  ARRIVAL TIME    BURST TIME      WAITING TIME    TURNAROUND TIME
1               0               9               0               9
2               1               4               8               12
3               2               9               11              20
AVERAGE WAITING TIME: 6.333333
AVERAGE TURNAROUND TIME: 13.666667
[s2019103562@centos8-linux Mon Apr 19 09:02 AM lab11]$
```

```
[s2019103562@centos8-linux Mon Apr 19 09:02 AM lab11]$ cat fcfs.c
#include<stdio.h>
#define MAX 30
int main(){
        int bt[MAX],at[MAX],temp[MAX],wt[MAX],tat[MAX],n;
        float awt=0,atat=0;
        printf("Enter the number of processes:\n");
        scanf("%d",&n);
        for(int i=0;i<n;i++){
                printf("Enter the arrival time of process %d:\n",i+1);
                scanf("%d",&at[i]);
                printf("Enter the burst time of process %d:\n",i+1);
                scanf("%d",&bt[i]);
        }
        printf("PROCESS\t ARRIVAL TIME\t BURST TIME\t WAITING TIME\t TURNAROUND TIME\n");
        temp[0]=0;
        for(int i=0;i<n;i++){
                wt[i]=0;
                tat[i]=0;
                temp[i+1]=temp[i]+bt[i];
                wt[i]=temp[i]-at[i];
                tat[i]=wt[i]+bt[i];
                awt=awt+wt[i];
                atat=atat+tat[i];
                printf("%d\t\t %d\t\t %d\t\t %d\t\t %d\n",i+1,at[i],bt[i],wt[i],tat[i]);
        }
        awt=awt/n;
        atat=atat/n;
        printf("AVERAGE WAITING TIME: %f\n",awt);
        printf("AVERAGE TURNAROUND TIME: %f\n",atat);
        return 0;
}

[s2019103562@centos8-linux Mon Apr 19 09:04 AM lab11]$
```

2.SHORTEST JOB FIRST SCHEDULING ALGORITHM(SJF)

```
[s2019103562@centos8-linux Mon Apr 19 09:34 AM lab11]$ vim sjf.c
[s2019103562@centos8-linux Mon Apr 19 09:36 AM lab11]$ gcc sjf.c -o sjf
[s2019103562@centos8-linux Mon Apr 19 09:36 AM lab11]$ ./sjf
Enter the number of processes:
5
Enter the process ID:
1
Enter the arrival time of process 1:
0
Enter the burst time of process 1:
4
Enter the process ID:
2
Enter the arrival time of process 2:
1
Enter the burst time of process 2:
3
Enter the process ID:
3
Enter the arrival time of process 3:
2
Enter the burst time of process 3:
1
Enter the process ID:
4
Enter the arrival time of process 4:
3
Enter the burst time of process 4:
2
Enter the process ID:
5
Enter the arrival time of process 5:
4
Enter the burst time of process 5:
6
```

```
PROCESS ID       ARRIVAL TIME    BURST TIME      WAITING TIME    TURNAROUND TIME
1                0               4               0               4
3                2               1               2               3
4                3               2               2               4
2                1               3               6               9
5                4               6               6               12
AVERAGE WAITING TIME: 3.200000
AVERAGE TURNAROUND TIME:32.000000
[s2019103562@centos8-linux Mon Apr 19 09:37 AM lab11]$
```

```
[s2019103562@centos8-linux Mon Apr 19 09:37 AM lab11]$ cat sjf.c
#include<stdio.h>
int mat[10][6];
void swap(int* a,int* b){
        int temp=*a;
        *a=*b;
        *b=temp;
}
void arrangeArrival(int n,int mat[][6]){
        for(int i=0;i<n;i++){
                for(int j=0;j<n-i-1;j++){
                        if(mat[1][j]>mat[1][j+1]){
                                for(int k=0;k<5;k++){
                                        swap(&mat[k][j],&mat[k][j+1]);
                                }
                        }
                }
        }
}
void completionTime(int n,int mat[][6]){
        int temp,val;
        mat[3][0]=mat[1][0]+mat[2][0];
        mat[5][0]=mat[3][0]-mat[1][0];
        mat[4][0]=mat[5][0]-mat[2][0];
        for(int i=1;i<n;i++){
                temp=mat[3][i-1];
                int low=mat[2][i];
                for(int j=i;j<n;j++){
                        if(temp>=mat[1][j] && low>=mat[2][j]){
                                low=mat[2][j];
                                val=j;
                        }
                }
```

```
                mat[3][val]=temp+mat[2][val];
                mat[5][val]=mat[3][val]-mat[1][val];
                mat[4][val]=mat[5][val]-mat[2][val];
                for(int k=0;k<6;k++){
                        swap(&mat[k][val],&mat[k][i]);
                }
        }
}

int main(){
        float twt=0,tat=0;
        int n;
        printf("Enter the number of processes:\n");
        scanf("%d",&n);
        for(int i=0;i<n;i++){
                printf("Enter the process ID:\n");
                scanf("%d",&mat[0][i]);
                printf("Enter the arrival time of process %d:\n",i+1);
                scanf("%d",&mat[1][i]);
                printf("Enter the burst time of process %d:\n",i+1);
                scanf("%d",&mat[2][i]);
        }
                arrangeArrival(n,mat);
                completionTime(n,mat);
                printf("PROCESS ID\t ARRIVAL TIME\t BURST TIME\t WAITING TIME\t TURNAROUND TIME\n");
                for(int i=0;i<n;i++){
                        printf("%d\t\t %d\t\t %d\t\t %d\t\t %d\n",mat[0][i],mat[1][i],mat[2][i],mat[4][i],mat[5][i]);
                }
                for(int i=0;i<n;i++){
                        twt=twt+mat[4][i];
                        tat=tat+mat[5][i];
                }
                twt=twt/n;
                tat/n;
                printf("AVERAGE WAITING TIME: %f\n",twt);
                printf("AVERAGE TURNAROUND TIME:%f\n",tat);
                return 0;
        }

[s2019103562@centos8-linux Mon Apr 19 09:39 AM lab11]$
```

3.PRIORITY SCHEDULING ALGORITHM

```
[s2019103562@centos8-linux Mon Apr 19 10:05 AM lab11]$ vim pri.c
[s2019103562@centos8-linux Mon Apr 19 10:06 AM lab11]$ gcc pri.c -o pri
[s2019103562@centos8-linux Mon Apr 19 10:06 AM lab11]$ ./pri
Enter the number of processes
7
Enter the process name:
p1
Enter the arrival time for process 1:
0
Enter the burst time for process 1:
3
Enter the priority for process 1:
2
Enter the process name:
p2
Enter the arrival time for process 2:
2
Enter the burst time for process 2:
5
Enter the priority for process 2:
6
Enter the process name:
p3
Enter the arrival time for process 3:
1
Enter the burst time for process 3:
4
Enter the priority for process 3:
3
```

```
Enter the process name:
p4
Enter the arrival time for process 4:
4
Enter the burst time for process 4:
2
Enter the priority for process 4:
5
Enter the process name:
p5
Enter the arrival time for process 5:
6
Enter the burst time for process 5:
9
Enter the priority for process 5:
7
Enter the process name:
p6
Enter the arrival time for process 6:
5
Enter the burst time for process 6:
4
Enter the priority for process 6:
4
Enter the process name:
p7
Enter the arrival time for process 7:
7
Enter the burst time for process 7:
10
Enter the priority for process 7:
10
```

```
PROCESS ID      ARRIVAL TIME    BURST TIME      PRIORITY        WAITING TIME    TURNAROUND TIME
p1              0               3               2               0               3
p3              1               4               3               2               6
p6              5               4               4               2               6
p4              4               2               5               7               9
p2              2               5               6               11              16
p5              6               9               7               12              21
p7              7               10              10              20              30
AVERAGE WAITING TIME: 7.714286
AVERAGE TURNAROUND TIME:13.000000
[s2019103562@centos8-linux Mon Apr 19 10:07 AM lab11]$ cat pri.c
```

```
[s2019103562@centos8-linux Mon Apr 19 10:07 AM lab11]$ cat pri.c
#include<stdio.h>
#include<string.h>
#define MAX 30
int main(){
        int bt[MAX],at[MAX],n,i,j,temp,p[MAX],st[MAX],ft[MAX],wt[MAX],ta[MAX];
        float twt=0,tat=0;
//      float awt=0,atat=0;
        char pn[MAX][MAX],t[MAX];
        printf("Enter the number of processes\n");
        scanf("%d",&n);
        for(i=0;i<n;i++){
                printf("Enter the process name:\n");
                scanf("%s",pn[i]);
                printf("Enter the arrival time for process %d:\n",i+1);
                scanf("%d",&at[i]);
                printf("Enter the burst time for process %d:\n",i+1);
                scanf("%d",&bt[i]);
                printf("Enter the priority for process %d:\n",i+1);
                scanf("%d",&p[i]);
        }
        for(i=0;i<n;i++){
                for(j=0;j<n;j++){
                        if(p[i]<p[j]){
                                temp=p[i];
                                p[i]=p[j];
                                p[j]=temp;
                                temp=at[i];
                                at[i]=at[j];
                                at[j]=temp;
                                temp=bt[i];
                                bt[i]=bt[j];
                                bt[j]=temp;
                                strcpy(t,pn[i]);
                                strcpy(pn[i],pn[j]);
                                strcpy(pn[j],t);
                        }
                }
        }
        for(i=0;i<n;i++){
                if(i==0){
                        st[i]=at[i];
                        wt[i]=st[i]-at[i];
                        ft[i]=st[i]+bt[i];
                        ta[i]=ft[i]-at[i];
                }
                else{
                        st[i]=ft[i-1];
                        wt[i]=st[i]-at[i];
                        ft[i]=st[i]+bt[i];
                        ta[i]=ft[i]-at[i];
                }
                twt=twt+wt[i];
                tat=tat+ta[i];
        }
        twt=twt/n;
        tat=tat/n;
        printf("PROCESS ID\t ARRIVAL TIME\t BURST TIME\t PRIORITY\t WAITING TIME\t TURNAROUND TIME\n");
        for(i=0;i<n;i++)
                printf("%s\t\t %d\t\t %d\t\t %d\t\t %d\t\t %d\n",pn[i],at[i],bt[i],p[i],wt[i],ta[i]);
        printf("AVERAGE WAITING TIME: %f\n",twt);
        printf("AVERAGE TURNAROUND TIME:%f\n",tat);
        return 0;
}
[s2019103562@centos8-linux Mon Apr 19 10:08 AM lab11]$
```