Operating System Principles Prashanth.S

Assignment-2 (19MID0020)

1)C program to create Zombie Process

```c
#include<sys/types.h>
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>

int main()
{
pid_t p;
p=fork();

printf("\nValue of p : %d",p);

if(p>0)   // Parent process --> Sleeping
{
        printf("\nInside Parent's process");
        printf("\nProcess ID : %d",getpid());

        sleep(30);
        //exit(0);  // Parent doesn't exits and waits for the child to return back by leaving the child alive
}

if(p==0)  // Child process  -->  exiting while the parent is sleeping
{
        printf("\nInside Child's process");
        printf("\nProcess ID of child : %d",getpid());
        printf("\nProcess ID of parent : %d",getppid());
        //sleep(30)
        exit(0);
}
return 0;
}
```

Output

```
prashanth@prashanth-VirtualBox:~/Process$ ./a.out &
[1] 2465
prashanth@prashanth-VirtualBox:~/Process$ pstree -p 2465
a.out(2465)──a.out(2466)
prashanth@prashanth-VirtualBox:~/Process$ ps aux | grep 2466
prashan+    2466  0.0  0.0      0     0 pts/0    Z    08:30   0:00 [a.out] <defunct>
prashan+    2470  0.0  0.0  17532   664 pts/0    S+   08:30   0:00 grep --color=auto 2466
prashanth@prashanth-VirtualBox:~/Process$ ps aux | grep 2466
prashan+    2466  0.0  0.0      0     0 pts/0    Z    08:30   0:00 [a.out] <defunct>
prashan+    2472  0.0  0.0  17532   736 pts/0    S+   08:30   0:00 grep --color=auto 2466
prashanth@prashanth-VirtualBox:~/Process$ ps aux | grep 2466
prashan+    2466  0.0  0.0      0     0 pts/0    Z    08:30   0:00 [a.out] <defunct>
prashan+    2474  0.0  0.0  17532   736 pts/0    S+   08:31   0:00 grep --color=auto 2466
prashanth@prashanth-VirtualBox:~/Process$ ps aux | grep 2466
prashan+    2466  0.0  0.0      0     0 pts/0    Z    08:30   0:00 [a.out] <defunct>
prashan+    2476  0.0  0.0  17532   664 pts/0    S+   08:31   0:00 grep --color=auto 2466
prashanth@prashanth-VirtualBox:~/Process$ ps aux | grep 2466
prashan+    2478  0.0  0.0  17532   664 pts/0    S+   08:31   0:00 grep --color=auto 2466
[1]+  Done                    ./a.out
prashanth@prashanth-VirtualBox:~/Process$
```

2) C program to create orphan process

```c
#include<sys/types.h>
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>

int main()
{
pid_t p;
p=fork();

printf("\nValue of p : %d",p);

if(p>0)   // Parent process --> exiting while the child is sleeping
{
{
        printf("\nInside Parent's process");
        printf("\nProcess ID : %d",getpid());

        //sleep(30);
        exit(0);      // Parent exits and doesn't waits for the child to return back by leaving the child orphan
}


if(p==0)  // Child process  -->  Sleeping
        printf("\nInside Child's process");
        printf("\nProcess ID of child : %d",getpid());
        printf("\nProcess ID of parent : %d",getppid());
        sleep(30)
        //exit(0);
}
return 0;
}

```

Output

```
prashanth@prashanth-VirtualBox: ~
prashanth@prashanth-VirtualBox:~$ ps aux | grep 3477
prashan+    3480  0.0  0.0  17532   728 pts/1    S+   20:47   0:00 grep --color=auto 3477
prashanth@prashanth-VirtualBox:~$ pstree -p 3477
prashanth@prashanth-VirtualBox:~$ pstree -p 3478
a.out(3478)
prashanth@prashanth-VirtualBox:~$ ps -o ppid=3478
   3478
   2191
   3421
prashanth@prashanth-VirtualBox:~$ pstree -p 2191
gnome-terminal-(2191)─┬─bash(3421)───pstree(3488)
                      ├─bash(3457)
                      ├─{gnome-terminal-}(2192)
                      ├─{gnome-terminal-}(2193)
                      ├─{gnome-terminal-}(2194)
                      └─{gnome-terminal-}(2201)
```

```
prashanth@prashanth-VirtualBox: ~
prashanth@prashanth-VirtualBox:~$ gcc orphan.c
prashanth@prashanth-VirtualBox:~$ ./a.out &
[1] 3477
prashanth@prashanth-VirtualBox:~$ The process ID of parent is 3477
The parent process executed completely without any waiting...
But the Child process is still alive .So it will be adopted by a new parent
the process ID of child is 3478
the new parent process ID of child is  1305
Completed...
[1]+  Done                    ./a.out
prashanth@prashanth-VirtualBox:~$
```

3) First come First Serve Scheduling Algorithm

```c
#include<stdio.h>

int main()

{
    int n,time_burst[20],time_wait[20],turn_wait[20],avgtime_wait=0,avgturn_wait=0,i,j;
    printf("Enter total number of processes:");
    scanf("%d",&n);

    printf("Enter Process Burst Timen for \n");
    for(i=0;i<n;i++)
    {
        printf("P[%d]:",i+1);
        scanf("%d",&time_burst[i]);
    }

    time_wait[0]=0;

    for(i=1;i<n;i++)
    {
        time_wait[i]=0;
        for(j=0;j<i;j++)
            time_wait[i]+=time_burst[j];
    }

    printf("\nProcess\tBurst_Time\tWaiting_Time\tTurnaround_Time");

    for(i=0;i<n;i++)
    {
        turn_wait[i]=time_burst[i]+time_wait[i];
        avgtime_wait+=time_wait[i];
        avgturn_wait+=turn_wait[i];
        printf("\nP[%d]\t\t%d\t\t%d\t\t%d",i+1,time_burst[i],time_wait[i],turn_wait[i]);
    }

    avgtime_wait/=i;
    avgturn_wait/=i;
    printf("\nAverage Waiting Time:%d",avgtime_wait);
    printf("\nAverage Turnaround Time:%d\n",avgturn_wait);

    return 0;
}
```

Output

```
prashanth@prashanth-VirtualBox:~$ gcc fcfs.c
prashanth@prashanth-VirtualBox:~$ ./a.out
Enter total number of processes:8
Enter Process Burst Timen for
P[1]:9
P[2]:7
P[3]:5
P[4]:3
P[5]:2
P[6]:1
P[7]:6
P[8]:4

Process Burst_Time      Waiting_Time    Turnaround_Time
P[1]            9               0               9
P[2]            7               9               16
P[3]            5               16              21
P[4]            3               21              24
P[5]            2               24              26
P[6]            1               26              27
P[7]            6               27              33
P[8]            4               33              37
Average Waiting Time:19
Average Turnaround Time:24
prashanth@prashanth-VirtualBox:~$
```

4) Shortest Job First scheduling algorithm

Code:

```c
#include<stdio.h>
int main()
{
   int time_burst[20],p[20],wait[20],time_turn[20],i,j,n,total=0,pos,temp;
   float time_avgwait,time_avg_turn;
   printf("Enter number of process:");
   scanf("%d",&n);

   printf("Enter Burst Time:\n");
   for(i=0;i<n;i++)
   {
       printf("p%d:",i+1);
       scanf("%d",&time_burst[i]);
       p[i]=i+1;
   }

  //sorting of time_burst times
   for(i=0;i<n;i++)
   {
       pos=i;
       for(j=i+1;j<n;j++)
       {
           if(time_burst[j]<time_burst[pos])
               pos=j;
       }

        temp=time_burst[i];
        time_burst[i]=time_burst[pos];
        time_burst[pos]=temp;

        temp=p[i];
        p[i]=p[pos];
        p[pos]=temp;
    }

    wait[0]=0;


    for(i=1;i<n;i++)
    {
        wait[i]=0;
        for(j=0;j<i;j++)
            wait[i]+=time_burst[j];

        total+=wait[i];
    }

    time_avgwait=(float)total/n;
    total=0;

    printf("\nProcess\tBurst_Time\tWaiting_Time\tTurnaround_Time");
    for(i=0;i<n;i++)
    {
        time_turn[i]=time_burst[i]+wait[i];
        total+=time_turn[i];
        printf("\np%d\t\t%d\t\t%d\t\t%d",p[i],time_burst[i],wait[i],time_turn[i]);
    }

    time_avg_turn=(float)total/n;
    printf("\nAverage Waiting Time=%f",time_avgwait);
    printf("\nAverage Turnaround Time=%f\n",time_avg_turn);

    return 0;
```

Output:

```
prashanth@prashanth-VirtualBox:~$ gcc sjf.c
prashanth@prashanth-VirtualBox:~$ ./a.out
Enter number of process:5
Enter Burst Time:
p1:8
p2:6
p3:4
p4:2
p5:10

Process Burst_Time      Waiting_Time    Turnaround_Time
p4              2               0               2
p3              4               2               6
p2              6               6               12
p1              8               12              20
p5              10              20              30
Average Waiting Time=8.000000
Average Turnaround Time=14.000000
```

5) Round Robin Scheduling Algorithm

```c
#include<stdio.h>

int main()
{
      int i,n, total = 0, x, counter = 0, time_quantum;
      int wait = 0, turn = 0, arrival[10], burst[10], temp[10];
      float avgwait, avgturn;
      printf("Enter Total Number of Processes:");
      scanf("%d", &n);
      x = n;
      printf("Enter Details of Process :\n");
      for(i = 0; i <n; i++)
      {
            printf("Enter arrival Time for P%d:",i+1);
            scanf("%d", &arrival[i]);
            printf("Enter burst Time for P%d:",i+1);
            scanf("%d", &burst[i]);
            temp[i] = burst[i];
      }

     printf("Enter Time Quantum:");
     scanf("%d", &time_quantum);
     printf("\nProcess ID\tBurst_Time\tTurnaround_Time\tWaiting_Time");
     for(total = 0, i = 0; x != 0;)
     {
           if(temp[i] <= time_quantum && temp[i] > 0)
           {
                 total = total + temp[i];
                 temp[i] = 0;
                 counter = 1;
           }
           else if(temp[i] > 0)
           {
                 temp[i] = temp[i] - time_quantum;
                 total = total + time_quantum;
           }
           if(temp[i] == 0 && counter == 1)
           {
                 x--;
     printf("\nProcess[%d]\t\t%d\t\t %d\t\t %d", i + 1, burst[i], total - arrival[i], total - arrival[i] - burst[i]);
     wait =wait + total - arrival[i] - burst[i];
     turn = turn + total - arrival[i];
     counter = 0;
           }
           if(i == n - 1)
           {
                 i = 0;
           }
           else if(arrival[i + 1] <= total)
           {
                 i++;
           }
           else
           {
                 i = 0;
           }
     }

     avgwait = wait * 1.0 / n;
     avgturn = turn * 1.0 / n;
     printf("\nAverage Waiting Time:%f", avgwait);
     printf("\nAvg Turnaround Time:%f", avgturn);
     return 0;
}
```

Output

```
prashanth@prashanth-VirtualBox:~$ gcc roundrobin.c
prashanth@prashanth-VirtualBox:~$ ./a.out
Enter Total Number of Processes:5
Enter Details of Process :
Enter arrival Time for P1:2
Enter burst Time for P1:4
Enter arrival Time for P2:1
Enter burst Time for P2:6
Enter arrival Time for P3:0
Enter burst Time for P3:4
Enter arrival Time for P4:6
Enter burst Time for P4:2
Enter arrival Time for P5:9
Enter burst Time for P5:5
Enter Time Quantum:4

Process ID      Burst_Time      Turnaround_Time Waiting_Time
Process[1]              4               2               -2
Process[3]              4               12              8
Process[4]              2               8               6
Process[2]              6               19              13
Process[5]              5               12              7
Average Waiting Time:6.400000
Avg Turnaround Time:10.600000prashanth@prashanth-VirtualBox:~$
```

6) Priority Scheduling Algorithm

```c
#include<stdio.h>
int main()
{
    int time_burst[20],p[20],time_wait[20],time_turn[20],pr[20],i,j,n,total=0,pos,temp,avgtime_wait,avgtime_turn;
    printf("Enter Total Number of Process:");
    scanf("%d",&n);

    printf("Enter Burst Time and Priority\n");
    for(i=0;i<n;i++)
    {
        printf("P[%d]\n",i+1);
        printf("Burst Time:");
        scanf("%d",&time_burst[i]);
        printf("Priority:");
        scanf("%d",&pr[i]);
        p[i]=i+1;
     }

    for(i=0;i<n;i++)
    {
        pos=i;
        for(j=i+1;j<n;j++)
        {
            if(pr[j]<pr[pos])
                pos=j;
        }

        temp=pr[i];
        pr[i]=pr[pos];
        pr[pos]=temp;
        temp=time_burst[i];
        time_burst[i]=time_burst[pos];
        time_burst[pos]=temp;
        temp=p[i];
        p[i]=p[pos];
        p[pos]=temp;
     }

     time_wait[0]=0;
     for(i=1;i<n;i++)
     {
        time_wait[i]=0;
        for(j=0;j<i;j++)
            time_wait[i]+=time_burst[j];

        total+=time_wait[i];
     }

    avgtime_wait=total/n;
    total=0;

    printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time");
    for(i=0;i<n;i++)
    {
        time_turn[i]=time_burst[i]+time_wait[i];
        total+=time_turn[i];
        printf("\nP[%d]\t\t%d\t\t%d\t\t\t%d",p[i],time_burst[i],time_wait[i],time_turn[i]);
    }

    avgtime_turn=total/n;
    printf("\nAverage Waiting Time=%d",avgtime_wait);
    printf("\nAverage Turnaround Time=%d\n",avgtime_turn);

        return 0;
}
```

Output

```
prashanth@prashanth-VirtualBox:~$ gcc psa.c
prashanth@prashanth-VirtualBox:~$ ./a.out
Enter Total Number of Process:5
Enter Burst Time and Priority
P[1]
Burst Time:10
Priority:2
P[2]
Burst Time:4
Priority:1
P[3]
Burst Time:6
Priority:4
P[4]
Burst Time:7
Priority:3
P[5]
Burst Time:8
Priority:5

Process Burst Time      Waiting Time    Turnaround Time
P[2]            4               0                       4
P[1]            10              4                       14
P[4]            7               14                      21
P[3]            6               21                      27
P[5]            8               27                      35
Average Waiting Time=13
Average Turnaround Time=20
prashanth@prashanth-VirtualBox:~$
```