(globals)

os 6.cpp

```c
int wait_time=0,turnaround_time=0,at[10],bt[10],rt[10];
printf("Enter Total Process:\t ");
scanf("%d",&n);
remain=n;
for(count=0;count<n;count++)
{
  printf("Enter Arrival Time and Burst Time for Process Process Number %d :",count+1);
   scanf("%d",&at[count]);
   scanf("%d",&bt[count]);
   rt[count]=bt[count];
}
printf("Enter Time Quantum:\t");
scanf("%d",&time_quantum);
printf("\n\nProcess\t|Turnaround Time|Waiting Time\n\n");
for(time=0,count=0;remain!=0;)
{
  if(rt[count]<=time_quantum && rt[count]>0)
  {
    time+=rt[count];
    rt[count]=0;
    flag=1;
  }
  else if(rt[count]>0)
  {
    rt[count]-=time_quantum;
    time+=time_quantum;
  }
  if(rt[count]==0 && flag==1)
  {
    remain--;
    printf("P[%d]\t|\t%d\t|\t%d\n",count+1,time-at[count],time-at[count]-bt[count]);
    wait_time+=time-at[count]-bt[count];
    turnaround_time+=time-at[count];
    flag=0;
  }
}
```

```
Enter Total Process:      4
Enter Arrival Time and Burst Time for Process Process Number 1 :0
4
Enter Arrival Time and Burst Time for Process Process Number 2 :1
5
Enter Arrival Time and Burst Time for Process Process Number 3 :2
6
Enter Arrival Time and Burst Time for Process Process Number 4 :3
7
Enter Time Quantum:      5


Process |Turnaround Time|Waiting Time

P[1]    |       4       |       0
P[2]    |       8       |       3
P[3]    |       18      |       12
P[4]    |       19      |       12

Average Waiting Time= 6.750000
Avg Turnaround Time = 12.250000
--------------------------------
Process exited after 41.12 seconds with return value 0
Press any key to continue . . .
```

Programiz
C Online Compiler

main.c

Run

```c
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/shm.h>
#include<string.h>
int main()
{
int i;
void *shared_memory;
char buff[100];
int shmid;
shmid=shmget((key_t)2345, 1024, 0666|IPC_CREAT);

printf("Key of shared memory is %d\n",shmid);
shared_memory=shmat(shmid,NULL,0);

printf("Process attached at %p\n",shared_memory);

printf("Enter some data to write to shared memory\n");
read(0,buff,100);
strcpy(shared_memory,buff);
printf("You wrote : %s\n",(char *)shared_memory);
}
```

Output   Clear

```
/tmp/ZZE1aiKTgQ.o
Key of shared memory is 0
Process attached at 0x7f80dc079000
Enter some data to write to shared memory
hello world
You wrote : hello world
```

File   Edit   Search   View   Project   Execute   Tools   AStyle   Window   Help

(globals)

TDM-GCC 4.9.2 64-bit Release

Project   Classes   Debug   os 6.cpp   os 10.cpp   os 8.cpp

```c
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <unistd.h>
4   #include <pthread.h>
5
6   void *myThreadFun(void *vargp)
7   {
8       sleep(1);
9       printf("Printing GeeksQuiz from Thread \n");
10      return NULL;
11  }
12
13  int main()
14  {
15      pthread_t thread_id;
16      printf("Before Thread\n");
17      pthread_create(&thread_id, NULL, myThreadFun, NULL);
18      pthread_join(thread_id, NULL);
19      printf("After Thread\n");
20      exit(0);
21  }
```
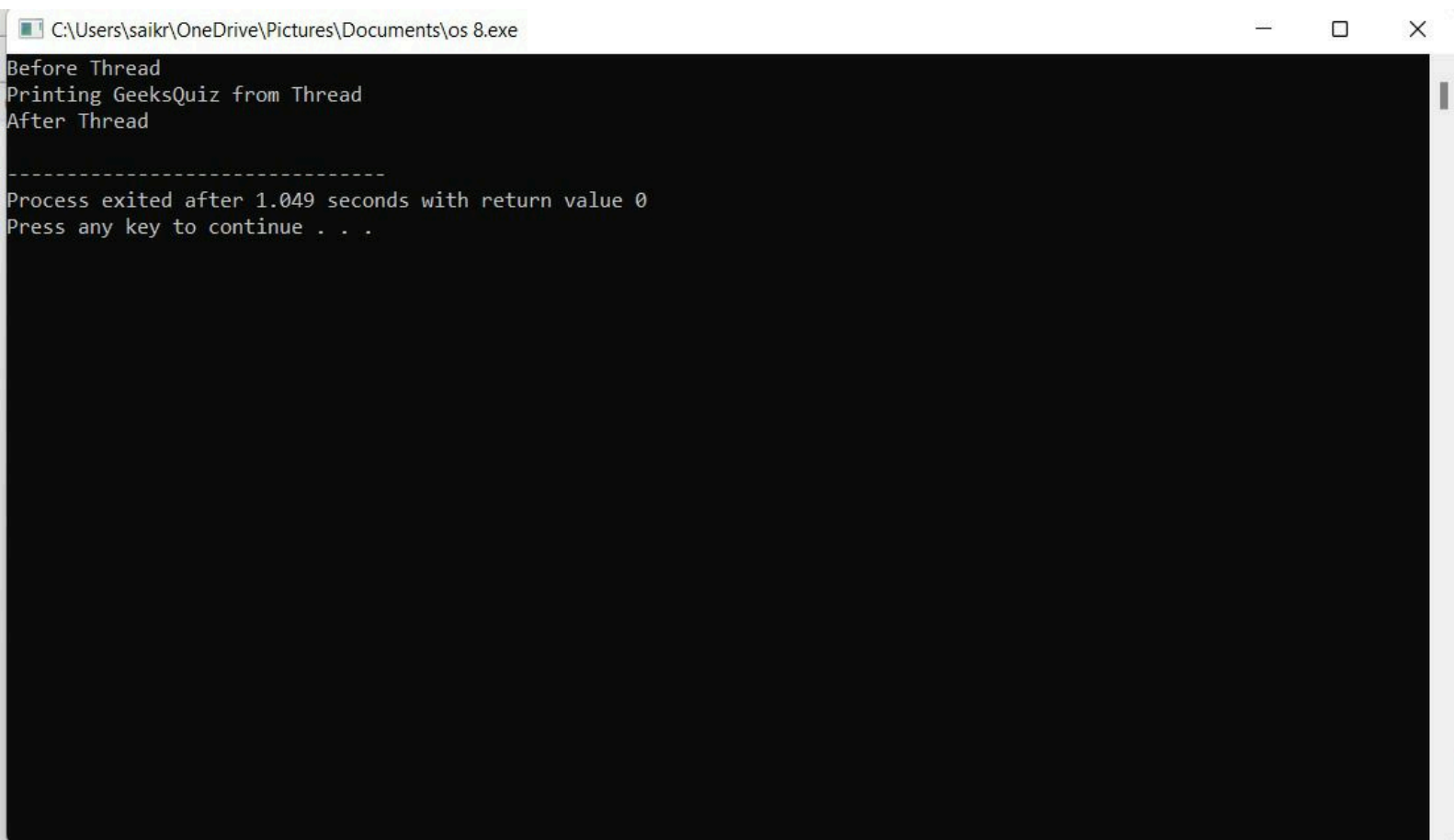
Line: 21   Col: 2   Sel: 0   Lines: 21   Length: 397   Insert   Done parsing in 0.063 seconds

```
Before Thread
Printing GeeksQuiz from Thread
After Thread

---------------------------------
Process exited after 1.049 seconds with return value 0
Press any key to continue . . .
```

TDM-GCC 4.9.2 64-bit Release

(globals)

Project Classes Debug   os 6.cpp os 10.cpp os 8.cpp os 9.cpp

```c
 1    #include<stdio.h>
 2
 3    #define n 4
 4
 5    int compltedPhilo = 0,i;
 6
 7    struct fork{
 8    int taken;
 9    }ForkAvil[n];
10
11    struct philosp{
12    int left;
13    int right;
14    }Philostatus[n];
15
16    void goForDinner(int philID){
17    if(Philostatus[philID].left==10 && Philostatus[philID].right==10)
18         printf("Philosopher %d completed his dinner\n",philID+1);
19    else if(Philostatus[philID].left==1 && Philostatus[philID].right==1){
20
21              printf("Philosopher %d completed his dinner\n",philID+1);
22
23              Philostatus[philID].left = Philostatus[philID].right = 10;
24              int otherFork = philID-1;
25
26              if(otherFork== -1)
27                  otherFork=(n-1);
28
29              ForkAvil[philID].taken = ForkAvil[otherFork].taken = 0;
30              printf("Philosopher %d released fork %d and fork %d\n",philID+1,philID+1,otherFork+1);
31              compltedPhilo++;
32         }
33              else if(Philostatus[philID].left==1 && Philostatus[philID].right==0){
34                  if(philID==(n-1)){
35                      if(ForkAvil[philID].taken==0){
```

Line: 87   Col: 2   Sel: 0   Lines: 87   Length: 3185   Insert   Done parsing in 0 seconds

93°F
Cloudy

ENG
IN

14:11
27-09-2022

```
Fork 1 taken by Philosopher 1
Fork 2 taken by Philosopher 2
Fork 3 taken by Philosopher 3
Philosopher 4 is waiting for fork 3

Till now num of philosophers completed dinner are 0

Fork 4 taken by Philosopher 1
Philosopher 2 is waiting for Fork 1
Philosopher 3 is waiting for Fork 2
Philosopher 4 is waiting for fork 3

Till now num of philosophers completed dinner are 0

Philosopher 1 completed his dinner
Philosopher 1 released fork 1 and fork 4
Fork 1 taken by Philosopher 2
Philosopher 3 is waiting for Fork 2
Philosopher 4 is waiting for fork 3

Till now num of philosophers completed dinner are 1

Philosopher 1 completed his dinner
Philosopher 2 completed his dinner
Philosopher 2 released fork 2 and fork 1
Fork 2 taken by Philosopher 3
Philosopher 4 is waiting for fork 3

Till now num of philosophers completed dinner are 2

Philosopher 1 completed his dinner
Philosopher 2 completed his dinner
Philosopher 3 completed his dinner
Philosopher 3 released fork 3 and fork 2
Fork 3 taken by philosopher 4

Till now num of philosophers completed dinner are 3

Philosopher 1 completed his dinner
Philosopher 2 completed his dinner
Philosopher 3 completed his dinner
Fork 4 taken by philosopher 4

Till now num of philosophers completed dinner are 3

Philosopher 1 completed his dinner
Philosopher 2 completed his dinner
Philosopher 3 completed his dinner
Philosopher 4 completed his dinner
```

```
1    #include<stdio.h>
2    int main( )
3    {
4        int bsize[10], psize[10], bno, pno, flags[10], allocation[10], i, j;
5        for(i = 0; i < 10; i++)
6        {
7            flags[i] = 0;
8            allocation[i] = -1;
9        }
10       printf("Enter no. of blocks: ");
11       scanf("%d", &bno);
12       printf("\nEnter size of each block: ");
13       for(i = 0; i < bno; i++)
14           scanf("%d", &bsize[i]);
15       printf("\nEnter no. of processes: ");
16       scanf("%d", &pno);
17       printf("\nEnter size of each process: ");
18       for(i = 0; i < pno; i++)
19           scanf("%d", &psize[i]);
20       for(i = 0; i < pno; i++)
21           for(j = 0; j < bno; j++)
22               if(flags[j] == 0 && bsize[j] >= psize[i])
23               {
24                   allocation[j] = i;
25                   flags[j] = 1;
26                   break;
27               }
28
29       printf("\nBlock no.\tsize\t\tprocess no.\t\tsize");
30       for(i = 0; i < bno; i++)
31       {
32           printf("\n%d\t\t%d\t\t", i+1, bsize[i]);
33           if(flags[i] == 1)
34               printf("%d\t\t%d",allocation[i]+1,psize[allocation[i]]);
35           else
```

Line: 39   Col: 1   Sel: 0   Lines: 39   Length: 929   Insert   Done parsing in 0 seconds

```
C:\Users\saikr\OneDrive\Pictures\Documents\os 10.exe                    —    □    ✕
Enter no. of blocks: 4

Enter size of each block: 3
4
5
6

Enter no. of processes: 4

Enter size of each process: 2
3
4
5

Block no.       size            process no.             size
1               3               1                       2
2               4               2                       3
3               5               3                       4
4               6               4                       5
--------------------------------
Process exited after 11.95 seconds with return value 0
Press any key to continue . . .
```