NAME: Kaustubh Gajanan Indulkar
TE-IT-A 25027

ASSIGNMENT 3

//Shortest Job First (Preemptive)

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

struct Process
{
	int pid;
	int at;
	int bt;
	int ct, tt, wt, rt, st;
};

int main()
{
	int size = 0;
	printf("Enter number of processes: ");
	scanf("%d", &size);
	struct Process ps[size];

	printf("\nEnter process Details: \n");
	for (int i = 0; i < size; ++i)
	{
		printf("Enter %dth process details: \n", i + 1);
		ps[i].pid = i + 1;

		printf("\tEnter Arrival Time: ");
		scanf("%d", &ps[i].at);

		printf("\tEnter Burst Time: ");
		scanf("%d", &ps[i].bt);
	}
	int n = size;
	int completed = 0;
	int currentTime = 0;
	int burstTimeR[4];
	bool iscompleted[4] = {false};
```

NAME: Kaustubh Gajanan Indulkar
TE-IT-A 25027

```
float avgWT = 0, avgTT = 0, avgRT = 0;

for (int i = 0; i < n; i++)
{
        burstTimeR[i] = ps[i].bt;
}

while (completed != n)
{

        int minimum = 99999;
        int miniI = -1;
        for (int i = 0; i < n; i++)
        {
                if ((ps[i].at <= currentTime) && (iscompleted[i] == false))
                {
                        if (burstTimeR[i] < minimum)
                        {
                                minimum = burstTimeR[i];
                                miniI = i;
                        }
                        if (burstTimeR[i] == minimum)
                        {
                                if (ps[i].at < ps[miniI].at)
                                {
                                        minimum = burstTimeR[i];
                                        miniI = i;
                                }
                        }
                }
        }

        if (miniI == -1)
        {
                currentTime++;
        }
        else
        {

                if (burstTimeR[miniI] == ps[miniI].bt)
                {
                        ps[miniI].st = currentTime;
                }
```

NAME: Kaustubh Gajanan Indulkar
TE-IT-A 25027

```
                    burstTimeR[minil] -= 1;
                    currentTime++;

                    if (burstTimeR[minil] == 0)

{

                        ps[minil].ct = currentTime;
                        ps[minil].tt = ps[minil].ct - ps[minil].at;
                        ps[minil].wt = ps[minil].tt - ps[minil].bt;
                        ps[minil].rt = ps[minil].st - ps[minil].at;

                        avgWT += ps[minil].wt;
                        avgTT += ps[minil].tt;
                        avgRT += ps[minil].rt;

                        completed++;
                        iscompleted[minil] = true;
                    }
                }
        }

printf("\n\n=============================================================
========================\n");
        printf("PID \t AT \t BT \t CT \t TAT \t WT \t RT \t\n");
        for (int i = 0; i < n; i++)
        {

                printf("%d \t %d \t %d \t %d \t %d \t %d \t %d \t\n", ps[i].pid, ps[i].at, ps[i].bt,
ps[i].ct, ps[i].tt, ps[i].wt, ps[i].rt);
        }

printf("\n\n=============================================================
========================\n");

        printf("\n\n AVG WT: %f", avgWT / n);
        printf("\n\n AVG TAT: %f", avgTT / n);
        printf("\n\n AVG RT: %f", avgRT / n);

printf("\n\n=============================================================
========================\n");
}
```

NAME: Kaustubh Gajanan Indulkar
TE-IT-A 25027

```
ubuntu@ubuntu:~$ cd Desktop
ubuntu@ubuntu:~/Desktop$ gcc 27OSass3sjf.c
ubuntu@ubuntu:~/Desktop$ ./a.out
Enter number of processes: 2

Enter process Details:
Enter 1th process details:
        Enter Arrival Time: 2
        Enter Burst Time: 5
Enter 2th process details:
        Enter Arrival Time: 1
        Enter Burst Time: 6


================================================================================
PID      AT      BT      CT      TAT     WT      RT
1        2       5       12      10      5       5
2        1       6       7       6       0       0


================================================================================


 AVG WT: 2.500000

 AVG TAT: 8.000000

 AVG RT: 2.500000
```

##Round Robin

```
#include <stdio.h>
// #include <limits.h>
#include <stdbool.h>
#include <stdlib.h>  //for qsort

struct process_struct
{
  int pid;
  int at;
  int bt;
  int ct, wt, tat, rt, start_time;
  int bt_remaining;
} ps[100];
```

NAME: Kaustubh Gajanan Indulkar
TE-IT-A 25027

```
int findmax(int a, int b)
{
  return a > b ? a : b;
}

int comparatorAT(const void *a, const void *b)
{
  int x = ((struct process_struct *)a)->at;
  int y = ((struct process_struct *)b)->at;
  if (x < y)
    return -1;
  else if (x >= y)
    return 1;
}

int comparatorPID(const void *a, const void *b)
{
  int x = ((struct process_struct *)a)->pid;
  int y = ((struct process_struct *)b)->pid;
  if (x < y)
    return -1;
  else if (x >= y)
    return 1;
}

int main()
{

  int n, index;
  int cpu_utilization;
  bool visited[100] = {false}, is_first_process = true;
  int current_time = 0, max_completion_time;
  int completed = 0, tq, total_idle_time = 0, length_cycle;
  printf("Enter total number of processes: ");
  scanf("%d", &n);
  int queue[100], front = -1, rear = -1;
  float sum_tat = 0, sum_wt = 0, sum_rt = 0;

  printf("\nEnter process Details: \n");
        for (int i = 0; i < n; ++i)
        {
                printf("Enter %dth process details: \n", i + 1);
                ps[i].pid = i + 1;

                printf("\tEnter Arrival Time: ");
```

NAME: Kaustubh Gajanan Indulkar
TE-IT-A 25027

```
            scanf("%d", &ps[i].at);
            ps[i].pid = i;

            printf("\tEnter Burst Time: ");
            scanf("%d", &ps[i].bt);
            ps[i].bt_remaining = ps[i].bt;
        }
 printf("\nEnter time quanta: ");
 scanf("%d", &tq);
 qsort((void *)ps, n, sizeof(struct process_struct), comparatorAT);
 front = rear = 0;
 queue[rear] = 0;
 visited[0] = true;

 while (completed != n)
 {
   index = queue[front];
   front++;

   if (ps[index].bt_remaining == ps[index].bt)
   {
     ps[index].start_time = findmax(current_time, ps[index].at);
     total_idle_time += (is_first_process == true) ? 0 : ps[index].start_time - current_time;
     current_time = ps[index].start_time;
     is_first_process = false;
   }

   if (ps[index].bt_remaining - tq > 0)
   {
     ps[index].bt_remaining -= tq;
     current_time += tq;
   }
   else
   {
     current_time += ps[index].bt_remaining;
     ps[index].bt_remaining = 0;
     completed++;

     ps[index].ct = current_time;
     ps[index].tat = ps[index].ct - ps[index].at;
     ps[index].wt = ps[index].tat - ps[index].bt;
     ps[index].rt = ps[index].start_time - ps[index].at;

     sum_tat += ps[index].tat;
     sum_wt += ps[index].wt;
```

NAME: Kaustubh Gajanan Indulkar
TE-IT-A 25027

```
      sum_rt += ps[index].rt;
    }
    for (int i = 1; i < n; i++)
    {
      if (ps[i].bt_remaining > 0 && ps[i].at <= current_time && visited[i] == false)
      {
        queue[++rear] = i;
        visited[i] = true;
      }
    }
    if (ps[index].bt_remaining > 0)
      queue[++rear] = index;
    if (front > rear)
    {
      for (int i = 1; i < n; i++)
      {
        if (ps[i].bt_remaining > 0)
        {
          queue[rear++] = i;
          visited[i] = true;
          break;
        }
      }
    }
  }
  max_completion_time = 1e-9;
  for (int i = 0; i < n; i++)
    max_completion_time = findmax(max_completion_time, ps[i].ct);
  length_cycle = max_completion_time - ps[0].at;
  cpu_utilization = (float)(length_cycle - total_idle_time) / length_cycle;
  qsort((void *)ps, n, sizeof(struct process_struct), comparatorPID);

printf("\n\n=============================================================
========================\n");
  printf("\nProcess No.\tAT\tBT\tStart Time\tCT\tTAT\tWT\tRT\n");
  for (int i = 0; i < n; i++)
    printf("%d\t\t%d\t%d\t\t%d\t%d\t%d\t%d\t%d\n",i+1, ps[i].at, ps[i].bt, ps[i].start_time,
ps[i].ct, ps[i].tat, ps[i].wt, ps[i].rt);
  printf("\n");

printf("\n\n=============================================================
========================\n");
  printf("\nAverage Turn Around time= %.2f", (float)sum_tat / n);
  printf("\nAverage Waiting Time= %.2f", (float)sum_wt / n);
  printf("\nAverage Response Time= %.2f\n", (float)sum_rt / n);
```

NAME: Kaustubh Gajanan Indulkar
TE-IT-A 25027

```
  return 0;
}
```

```
ubuntu@ubuntu:~/Desktop$ gcc 27OSass3rr.c
ubuntu@ubuntu:~/Desktop$ ./a.out
Enter total number of processes: 2

Enter process Details:
Enter 1th process details:
        Enter Arrival Time: 4
        Enter Burst Time: 6
Enter 2th process details:
        Enter Arrival Time: 3
        Enter Burst Time: 7

Enter time quanta: 2


================================================================================

Process No.     AT      BT      Start Time      CT      TAT     WT      RT
1               4       6               5       15      11      5       1
2               3       7               3       16      13      6       0



================================================================================

Average Turn Around time= 12.00
Average Waiting Time= 5.50
Average Response Time= 0.50
```