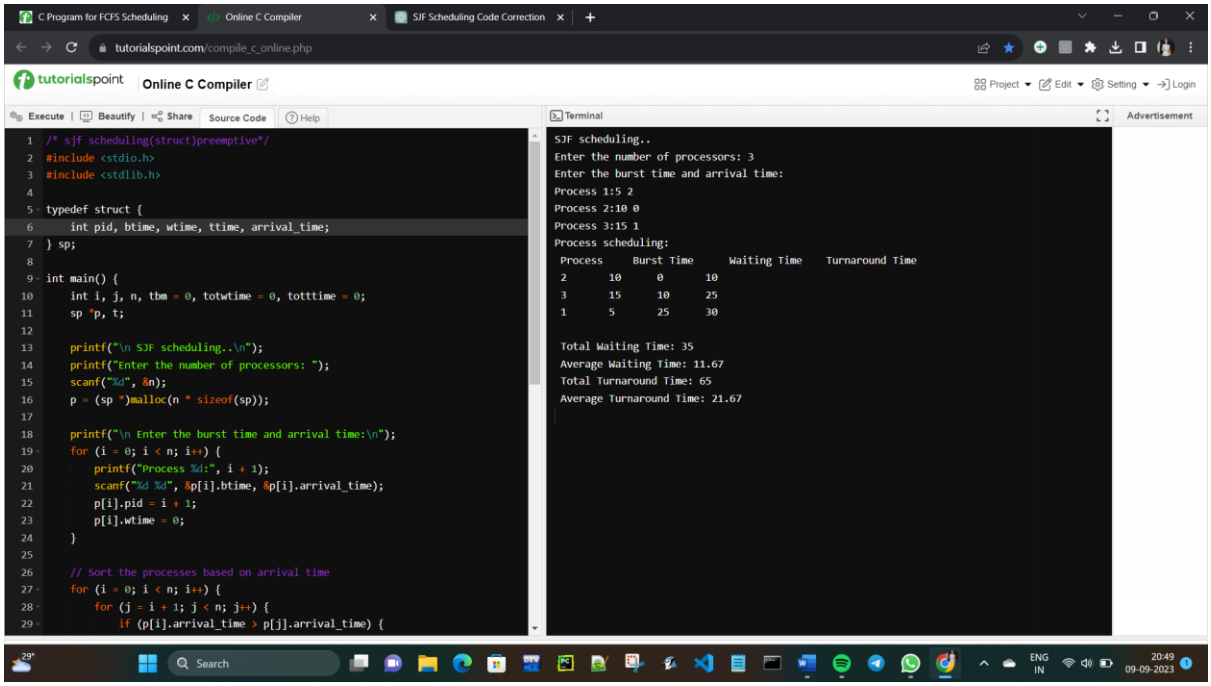
/* sjf scheduling(struct)preemptive*/

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

typedef struct {
   int pid, btime, wtime, ttime, arrival_time;
} sp;

int main() {
   int i, j, n, tbm = 0, totwtime = 0, totttime = 0;
   sp *p, t;

   printf("\n SJF scheduling..\n");
   printf("Enter the number of processors: ");
   scanf("%d", &n);
   p = (sp *)malloc(n * sizeof(sp));

   printf("\n Enter the burst time and arrival time:\n");
   for (i = 0; i < n; i++) {
      printf("Process %d:", i + 1);
      scanf("%d %d", &p[i].btime, &p[i].arrival_time);
      p[i].pid = i + 1;
      p[i].wtime = 0;
   }

   // Sort the processes based on arrival time
   for (i = 0; i < n; i++) {
      for (j = i + 1; j < n; j++) {
         if (p[i].arrival_time > p[j].arrival_time) {
            t = p[i];
            p[i] = p[j];
```

```
            p[j] = t;
        }
    }
}

printf("\n Process scheduling:\n");
printf(" Process \t Burst Time \t Waiting Time \t Turnaround Time\n");
for (i = 0; i < n; i++) {
    tbm += p[i].btime;
    p[i].ttime = tbm;
    p[i].wtime = tbm - p[i].btime;

    totwtime += p[i].wtime;
    totttime += p[i].ttime;

    printf(" %d\t\t %d\t\t %d\t\t %d\n", p[i].pid, p[i].btime, p[i].wtime, p[i].ttime);
}

printf("\n Total Waiting Time: %d\n", totwtime);
printf(" Average Waiting Time: %.2f\n", (float)totwtime / n);
printf(" Total Turnaround Time: %d\n", totttime);
printf(" Average Turnaround Time: %.2f\n", (float)totttime / n);

free(p);
return 0;
}
```


C Program for FCFS Scheduling
Online C Compiler
SJF Scheduling Code Correction
tutorialspoint.com/compile_c_online.php
tutorialspoint
Online C Compiler
Project
Edit
Setting
Login
Execute | Beautify | Share
Source Code
Help
Terminal
Advertisement
/* sjf scheduling(struct)preemptive*/
#include <stdio.h>
#include <stdlib.h>
typedef struct {
int pid, btime, wtime, ttime, arrival_time;
} sp;
int main() {
int i, j, n, tbm = 0, totwtime = 0, totttime = 0;
sp *p, t;
printf("\n SJF scheduling..\n");
printf("Enter the number of processors: ");
scanf("%d", &n);
p = (sp *)malloc(n * sizeof(sp));
printf("\n Enter the burst time and arrival time:\n");
for (i = 0; i < n; i++) {
printf("Process %d:", i + 1);
scanf("%d %d", &p[i].btime, &p[i].arrival_time);
p[i].pid = i + 1;
p[i].wtime = 0;
}
// Sort the processes based on arrival time
for (i = 0; i < n; i++) {
for (j = i + 1; j < n; j++) {
if (p[i].arrival_time > p[j].arrival_time) {
SJF scheduling..
Enter the number of processors: 3
Enter the burst time and arrival time:
Process 1:5 2
Process 2:10 0
Process 3:15 1
Process scheduling:
Process Burst Time Waiting Time Turnaround Time
2 10 0 10
3 15 10 25
1 5 25 30
Total Waiting Time: 35
Average Waiting Time: 11.67
Total Turnaround Time: 65
Average Turnaround Time: 21.67
29°
Search
ENG
IN
20:49
09-09-2023