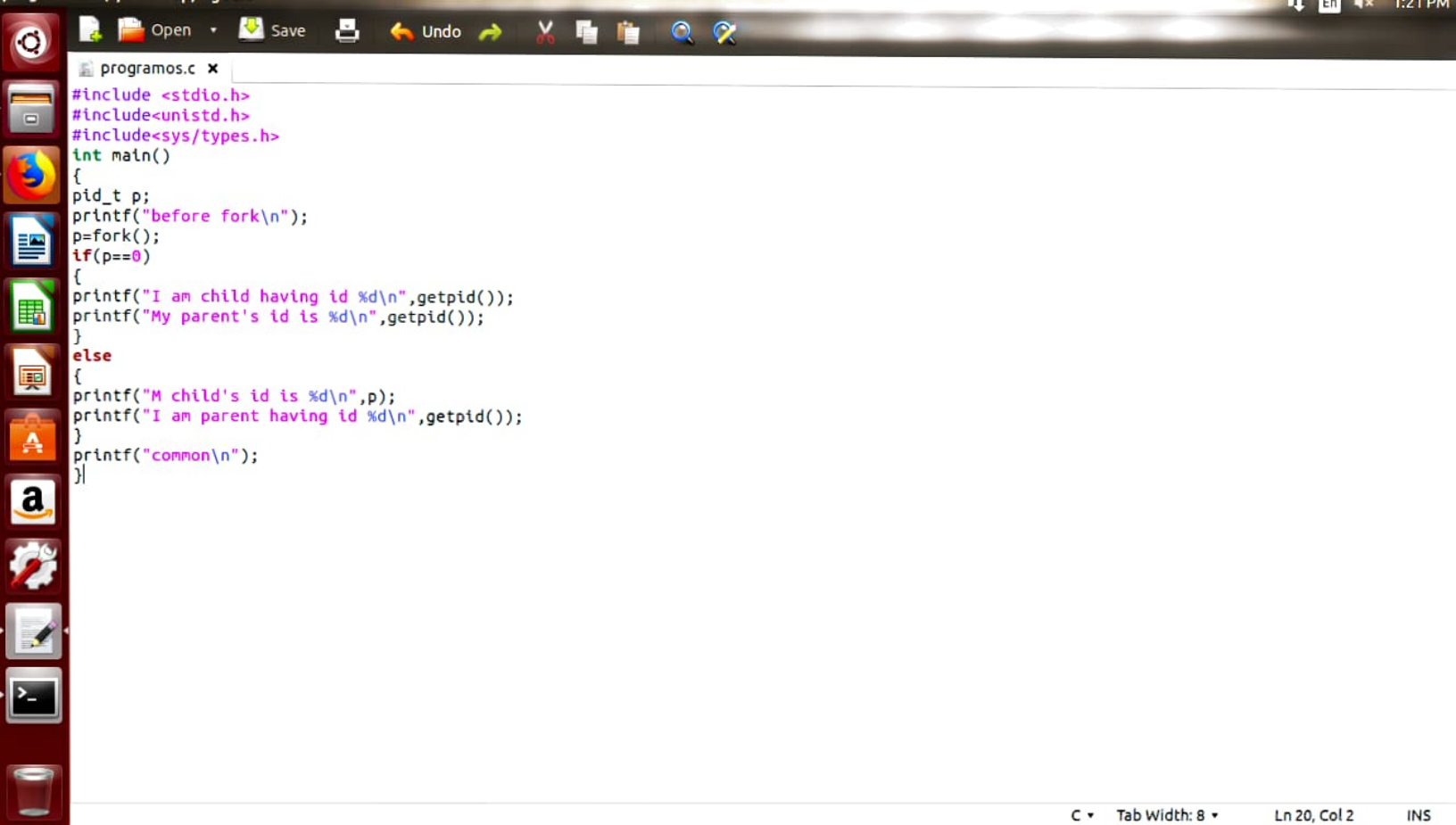
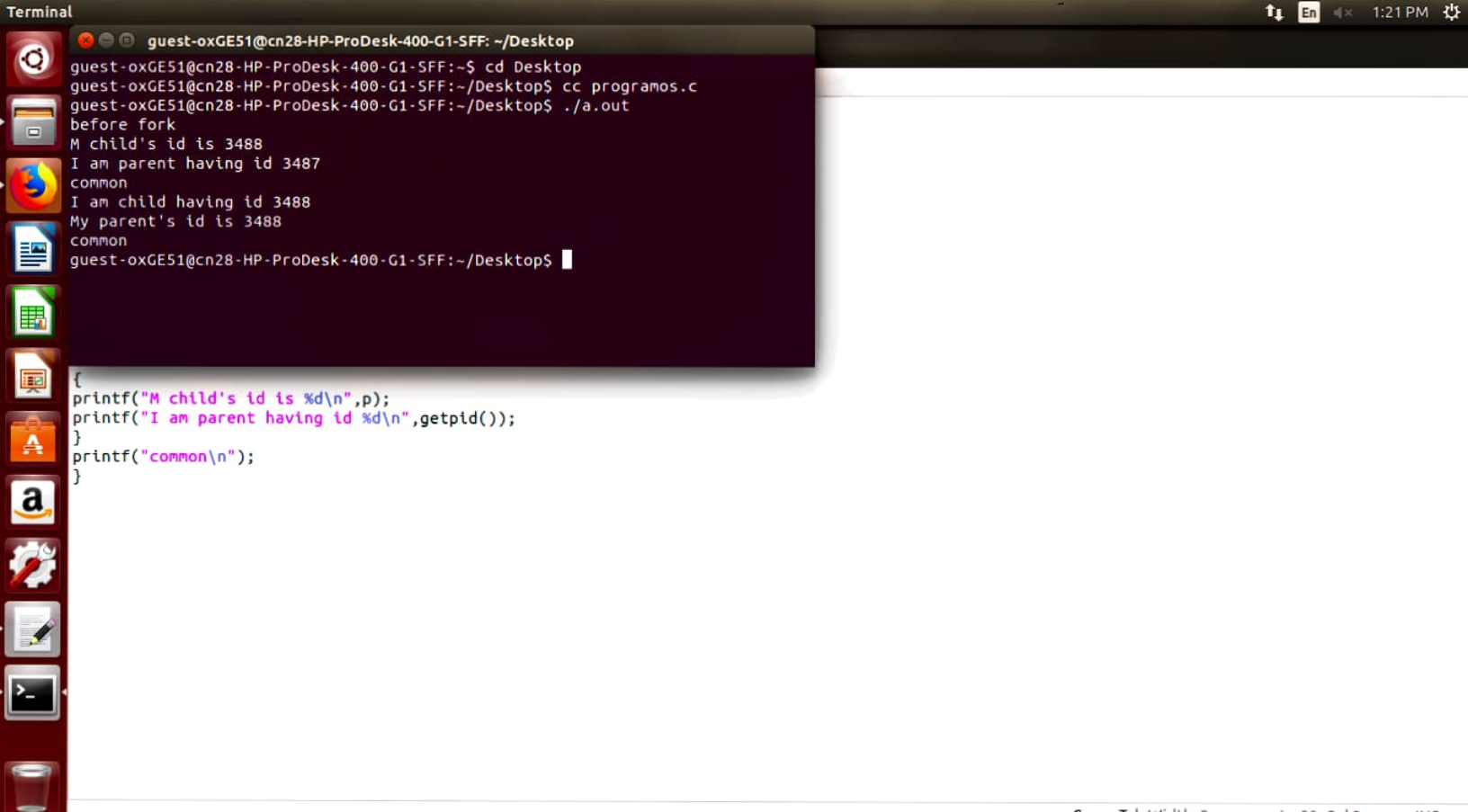
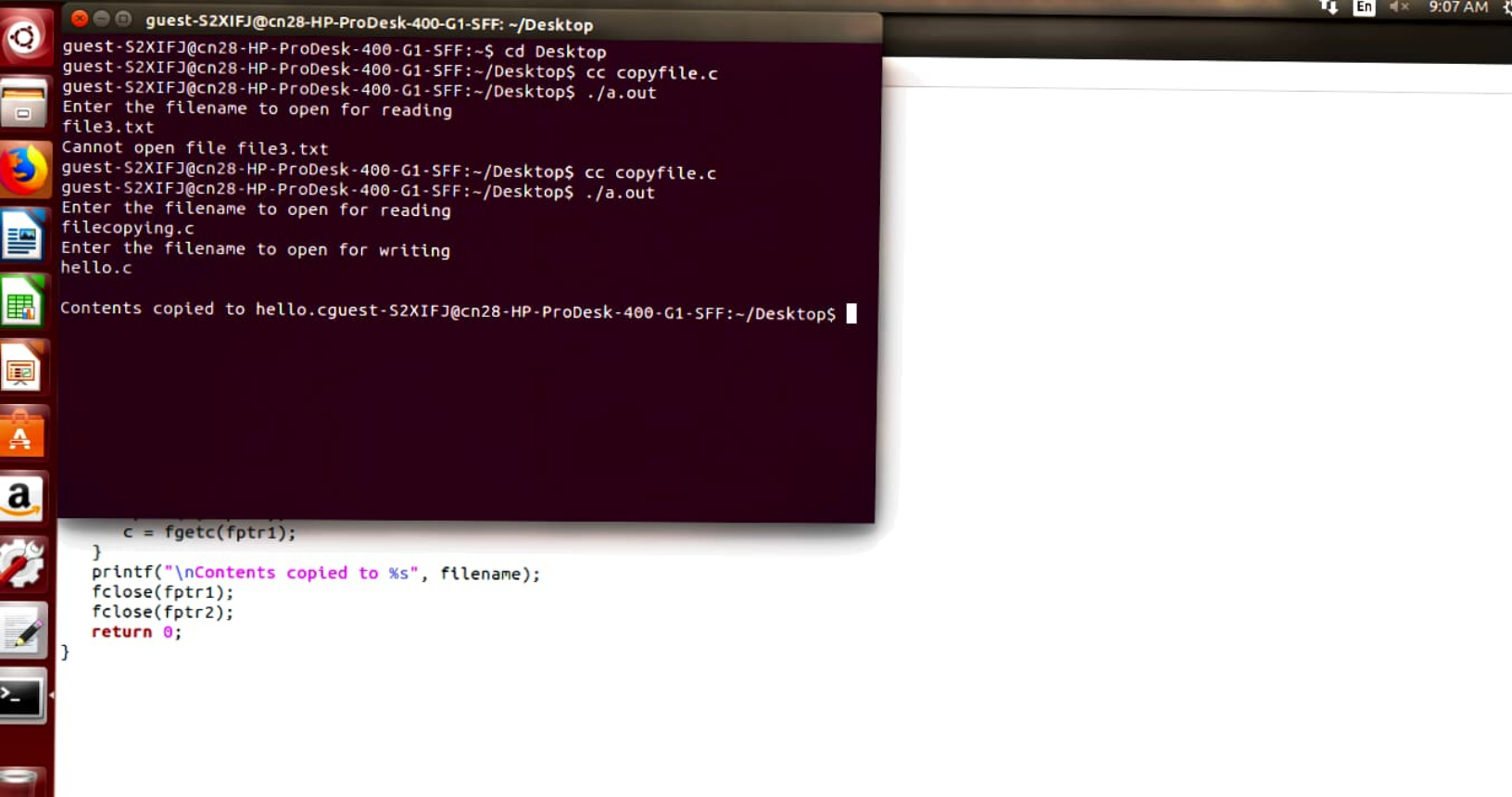

Open
Save
Undo
programos.c
#include <stdio.h>
#include<unistd.h>
#include<sys/types.h>
int main()
{
pid_t p;
printf("before fork\n");
p=fork();
if(p==0)
{
printf("I am child having id %d\n",getpid());
printf("My parent's id is %d\n",getpid());
}
else
{
printf("M child's id is %d\n",p);
printf("I am parent having id %d\n",getpid());
}
printf("common\n");
}
C
Tab Width: 8
Ln 20, Col 2
INS


Terminal
1:21 PM
guest-oxGE51@cn28-HP-ProDesk-400-G1-SFF: ~/Desktop
guest-oxGE51@cn28-HP-ProDesk-400-G1-SFF:~$ cd Desktop
guest-oxGE51@cn28-HP-ProDesk-400-G1-SFF:~/Desktop$ cc programos.c
guest-oxGE51@cn28-HP-ProDesk-400-G1-SFF:~/Desktop$ ./a.out
before fork
M child's id is 3488
I am parent having id 3487
common
I am child having id 3488
My parent's id is 3488
common
guest-oxGE51@cn28-HP-ProDesk-400-G1-SFF:~/Desktop$
{
printf("M child's id is %d\n",p);
printf("I am parent having id %d\n",getpid());
}
printf("common\n");
}

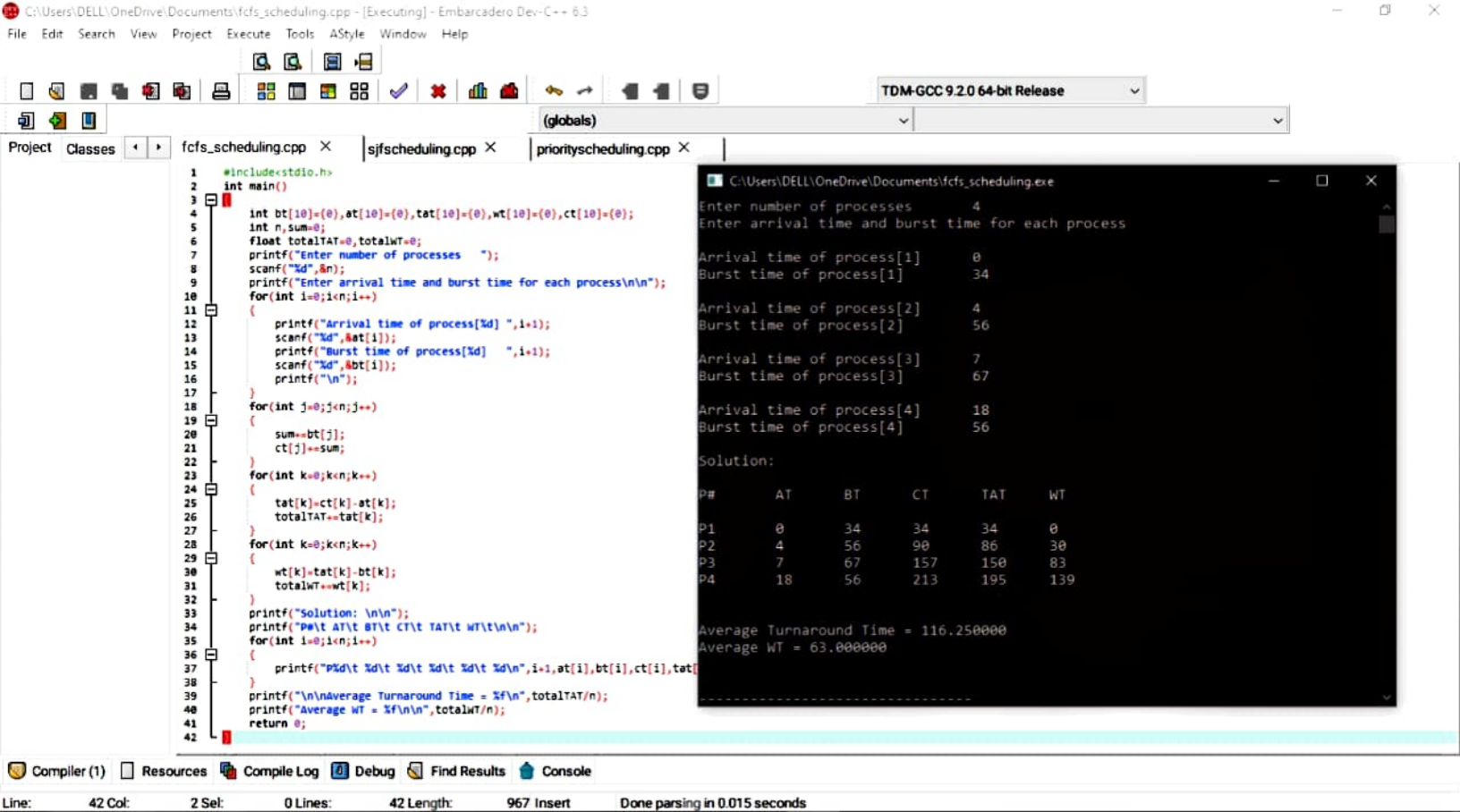
Open Save Undo

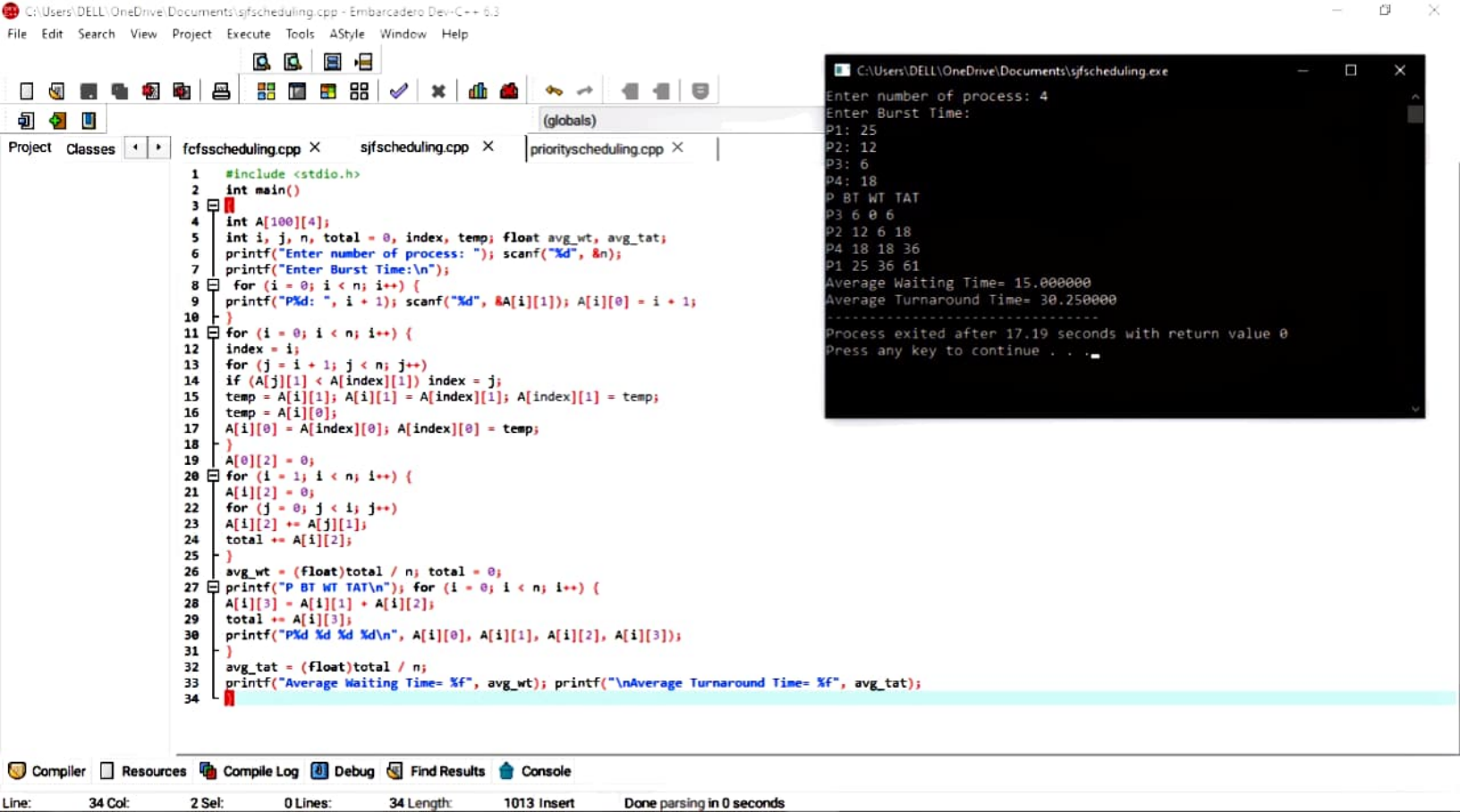
filecopying.c × copyfile.c × hello.c ×

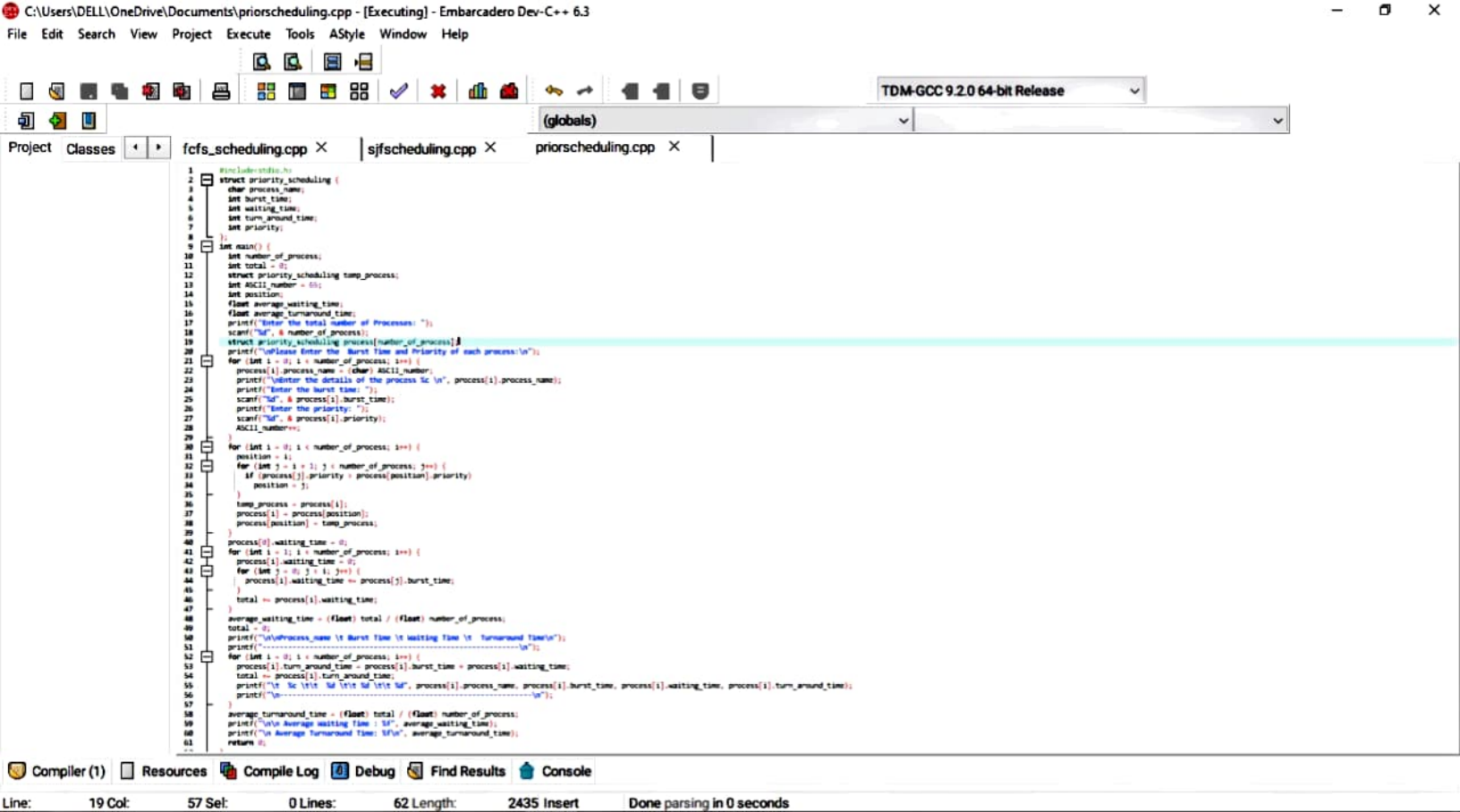
```c
#include <stdio.h>
#include <stdlib.h>
int main(){
   FILE *fptr1, *fptr2;
   char filename[100], c;
   printf("Enter the filename to open for reading \n");
   scanf("%s",filename);
   fptr1 = fopen(filename, "r");
   if (fptr1 == NULL){
      printf("Cannot open file %s \n", filename);
      exit(0);
   }
   printf("Enter the filename to open for writing \n");
   scanf("%s", filename);
   fptr2 = fopen(filename, "w");
   if (fptr2 == NULL){
      printf("Cannot open file %s \n", filename);
      exit(0);
   }
   c = fgetc(fptr1);
   while (c != EOF){
      fputc(c, fptr2);
      c = fgetc(fptr1);
   }
   printf("\nContents copied to %s", filename);
   fclose(fptr1);
   fclose(fptr2);
   return 0;
}
```




```
    c = fgetc(fptr1);
  }
  printf("\nContents copied to %s", filename);
  fclose(fptr1);
  fclose(fptr2);
  return 0;
}
```


C:\Users\DELL\OneDrive\Documents\fcfs_scheduling.cpp - [Executing] - Embarcadero Dev-C++ 6.3
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 9.2.0 64-bit Release
(globals)
Project Classes
fcfs_scheduling.cpp
sjfscheduling.cpp
priorityscheduling.cpp
#include<stdio.h>
int main()
{
int bt[10]={0},at[10]={0},tat[10]={0},wt[10]={0},ct[10]={0};
int n,sum=0;
float totalTAT=0,totalWT=0;
printf("Enter number of processes ");
scanf("%d",&n);
printf("Enter arrival time and burst time for each process\n\n");
for(int i=0;i<n;i++)
{
printf("Arrival time of process[%d] ",i+1);
scanf("%d",&at[i]);
printf("Burst time of process[%d] ",i+1);
scanf("%d",&bt[i]);
printf("\n");
}
for(int j=0;j<n;j++)
{
sum+=bt[j];
ct[j]+=sum;
}
for(int k=0;k<n;k++)
{
tat[k]=ct[k]-at[k];
totalTAT+=tat[k];
}
for(int k=0;k<n;k++)
{
wt[k]=tat[k]-bt[k];
totalWT+=wt[k];
}
printf("Solution: \n\n");
printf("P#\t AT\t BT\t CT\t TAT\t WT\t\n\n");
for(int i=0;i<n;i++)
{
printf("P%d\t %d\t %d\t %d\t %d\t %d\n",i+1,at[i],bt[i],ct[i],tat[
}
printf("\n\nAverage Turnaround Time = %f\n",totalTAT/n);
printf("Average WT = %f\n\n",totalWT/n);
return 0;
C:\Users\DELL\OneDrive\Documents\fcfs_scheduling.exe
Enter number of processes 4
Enter arrival time and burst time for each process
Arrival time of process[1] 0
Burst time of process[1] 34
Arrival time of process[2] 4
Burst time of process[2] 56
Arrival time of process[3] 7
Burst time of process[3] 67
Arrival time of process[4] 18
Burst time of process[4] 56
Solution:
P# AT BT CT TAT WT
P1 0 34 34 34 0
P2 4 56 90 86 30
P3 7 67 157 150 83
P4 18 56 213 195 139
Average Turnaround Time = 116.250000
Average WT = 63.000000
Compiler (1) Resources Compile Log Debug Find Results Console
Line: 42 Col: 2 Sel: 0 Lines: 42 Length: 967 Insert Done parsing in 0.015 seconds


C:\Users\DELL\OneDrive\Documents\sjfscheduling.cpp - Embarcadero Dev-C++ 6.3
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Classes
fcfsscheduling.cpp
sjfscheduling.cpp
priorityscheduling.cpp
#include <stdio.h>
int main()
{
int A[100][4];
int i, j, n, total = 0, index, temp; float avg_wt, avg_tat;
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", &A[i][1]); A[i][0] = i + 1;
}
for (i = 0; i < n; i++) {
index = i;
for (j = i + 1; j < n; j++)
if (A[j][1] < A[index][1]) index = j;
temp = A[i][1]; A[i][1] = A[index][1]; A[index][1] = temp;
temp = A[i][0];
A[i][0] = A[index][0]; A[index][0] = temp;
}
A[0][2] = 0;
for (i = 1; i < n; i++) {
A[i][2] = 0;
for (j = 0; j < i; j++)
A[i][2] += A[j][1];
total += A[i][2];
}
avg_wt = (float)total / n; total = 0;
printf("P BT WT TAT\n"); for (i = 0; i < n; i++) {
A[i][3] = A[i][1] + A[i][2];
total += A[i][3];
printf("P%d %d %d %d\n", A[i][0], A[i][1], A[i][2], A[i][3]);
}
avg_tat = (float)total / n;
printf("Average Waiting Time= %f", avg_wt); printf("\nAverage Turnaround Time= %f", avg_tat);
}
C:\Users\DELL\OneDrive\Documents\sjfscheduling.exe
Enter number of process: 4
Enter Burst Time:
P1: 25
P2: 12
P3: 6
P4: 18
P BT WT TAT
P3 6 0 6
P2 12 6 18
P4 18 18 36
P1 25 36 61
Average Waiting Time= 15.000000
Average Turnaround Time= 30.250000
Process exited after 17.19 seconds with return value 0
Press any key to continue . . .
Compiler Resources Compile Log Debug Find Results Console
Line: 34 Col: 2 Sel: 0 Lines: 34 Length: 1013 Insert Done parsing in 0 seconds


C:\Users\DELL\OneDrive\Documents\priorscheduling.cpp - [Executing] - Embarcadero Dev-C++ 6.3
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 9.2.0 64-bit Release
(globals)
Project Classes
fcfs_scheduling.cpp
sjfscheduling.cpp
priorscheduling.cpp
#include<stdio.h>
struct priority_scheduling {
char process_name;
int burst_time;
int waiting_time;
int turn_around_time;
int priority;
};
int main() {
int number_of_process;
int total = 0;
struct priority_scheduling temp_process;
int ASCII_number = 65;
int position;
float average_waiting_time;
float average_turnaround_time;
printf("Enter the total number of Processes: ");
scanf("%d", &number_of_process);
struct priority_scheduling process[number_of_process];
printf("\nPlease Enter the Burst Time and Priority of each process:\n");
for (int i = 0; i < number_of_process; i++) {
process[i].process_name = (char) ASCII_number;
printf("\nEnter the details of the process %c \n", process[i].process_name);
printf("Enter the burst time: ");
scanf("%d", &process[i].burst_time);
printf("Enter the priority: ");
scanf("%d", &process[i].priority);
ASCII_number++;
}
for (int i = 0; i < number_of_process; i++) {
position = i;
for (int j = i + 1; j < number_of_process; j++) {
if (process[j].priority < process[position].priority)
position = j;
}
temp_process = process[i];
process[i] = process[position];
process[position] = temp_process;
}
process[0].waiting_time = 0;
for (int i = 1; i < number_of_process; i++) {
process[i].waiting_time = 0;
for (int j = 0; j < i; j++) {
process[i].waiting_time += process[j].burst_time;
}
total += process[i].waiting_time;
}
average_waiting_time = (float) total / (float) number_of_process;
total = 0;
printf("\n\nProcess_name \t Burst Time \t Waiting Time \t Turnaround Time\n");
printf("---\n");
for (int i = 0; i < number_of_process; i++) {
process[i].turn_around_time = process[i].burst_time + process[i].waiting_time;
total += process[i].turn_around_time;
printf("\t %c \t\t %d \t\t %d \t\t %d", process[i].process_name, process[i].burst_time, process[i].waiting_time, process[i].turn_around_time);
printf("\n---\n");
}
average_turnaround_time = (float) total / (float) number_of_process;
printf("\n\n Average Waiting Time : %f", average_waiting_time);
printf("\n Average Turnaround Time: %f\n", average_turnaround_time);
return 0;
}
Compiler (1)
Resources
Compile Log
Debug
Find Results
Console
Line: 19 Col: 57 Sel: 0 Lines: 62 Length: 2435 Insert
Done parsing in 0 seconds

C:\Users\DELL\OneDrive\Documents\priorscheduling.exe

```
Enter the burst time: 15
Enter the priority: 3

Enter the details of the process C
Enter the burst time: 12
Enter the priority: 1

Enter the details of the process D
Enter the burst time: 13
Enter the priority: 4


Process_name     Burst Time      Waiting Time     Turnaround Time
-----------------------------------------------------------------
        D               13              0               13
-----------------------------------------------------------------
        B               15              13              28
-----------------------------------------------------------------
        A               3               28              31
-----------------------------------------------------------------
        C               12              31              43
-----------------------------------------------------------------


 Average Waiting Time : 18.000000
 Average Turnaround Time: 28.750000

--------------------------------
Process exited after 56.22 seconds with return value 0
Press any key to continue . . .
```