

**CS-303 Operation System** Lab Project Report

**Submitted by:**

Jibran Bilal Khan (20-SE-040)

Daniyal Tabassum (20-SE-025)

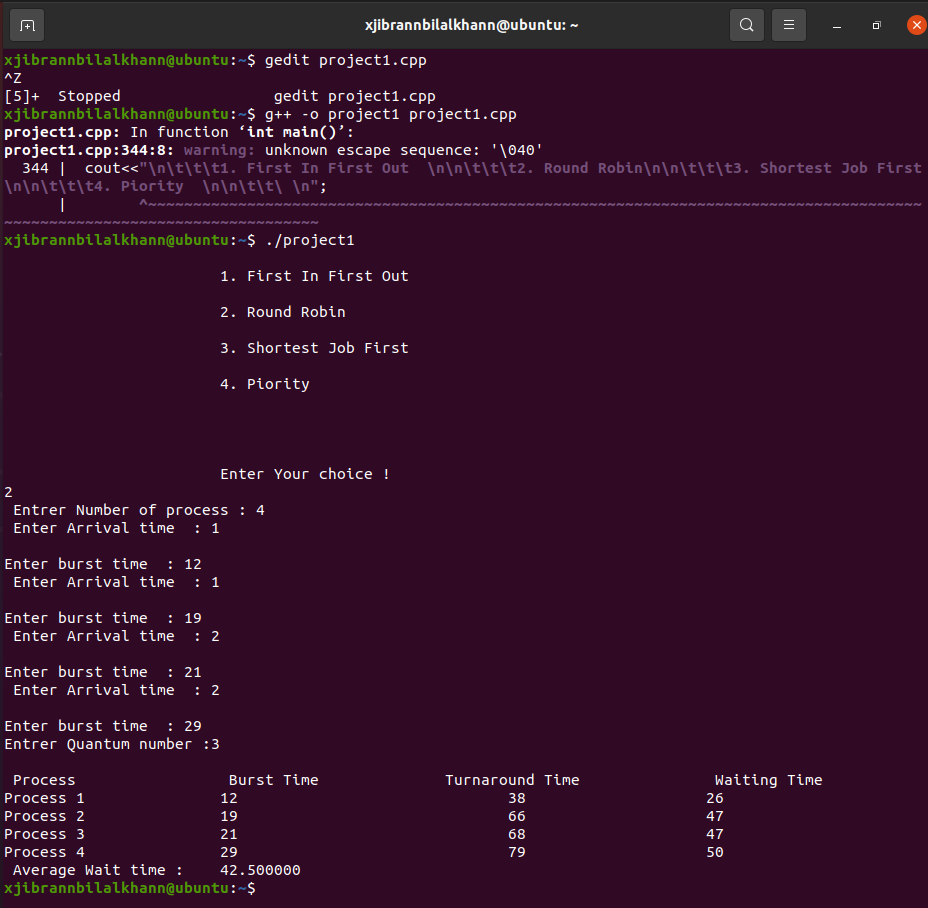
**Submitted to:**

Sir. Umer Aftab

**Department Of Software Engineering**

**HITEC UNIVERSITY, TAXILA**

|  |
| --- |
| *#include <iostream>*  *using namespace std;*  *class cpusch{*  *private:*  *int n,x,total;*  *int processes[15];*  *int arrival[15];*  *int bursttime[15];*  *int priority[15];*  *int waiting[15];*  *int tat[15];*  *int avtat;*  *int avwt;*  *public:*  *void first(){*  *cout<<"Enter the Number of Processes : ";*  *cin>>n;*  *//int processes[n];*  *for(int i=0 ; i<n ; i++){*  *processes[i]=i+1;*  *}*  *for(int i=0 ; i<n; i++){*  *cout<<"\n\nProccess Details";*  *cout<<"\nEnter the Arrival Time for Proccess "<<i+1<<" : ";*  *cin>>arrival[i];*  *cout<<"\nEnter the Burst Time for Proccess "<<i+1<<" : ";*  *cin>>bursttime[i];*  *cout<<"\nEnter the Priority for Proccess "<<i+1<<" : ";*  *cin>>priority[i];*  *}*  *}*  *void display(){*  *for(int i=0 ; i<n ; i++){*  *cout<<"\n\nPROCESSES "<<i+1;*  *cout<<"\n----------";*  *cout<<"\n\nArrival Time : "<<arrival[i];*  *cout<<"\n\nBurst Time : "<<bursttime[i];*  *cout<<"\n\nPriority Time : "<<priority[i];*  *}*    *}*  *void fcfs(){*  *cout<<"\n\n\_\_";*  *cout<<"\n\nFCFS CPU Scheduling";*  *cout<<"\n---------------------";*  *waiting[0]=0; //waiting time for first process is 0*    *//calculating waiting time*  *for(int i=1;i<n;i++)*  *{*  *waiting[i]=0;*  *for(int j=0;j<i;j++)*  *waiting[i]+=bursttime[j];*  *}*    *cout<<"\nProcess\t\tBurst Time\tWaiting Time\tTurnaround Time";*    *//calculating turnaround time*  *for(int i=0;i<n;i++)*  *{*  *tat[i]=bursttime[i]+waiting[i];*  *avwt+=waiting[i];*  *avtat+=tat[i];*  *cout<<"\nP["<<i+1<<"]"<<"\t\t"<<bursttime[i]<<"\t\t"<<waiting[i]<<"\t\t"<<tat[i];*  *}*      *avwt/=n;*  *avtat/=n;*  *cout<<"\n\nAverage Waiting Time:"<<avwt;*  *cout<<"\nAverage Turnaround Time:"<<avtat;*    *}*    *//Priority Scheduling*  *void prioritysch(){*  *int temp;*  *//sorting burst time, priority and process number in ascending order using selection sort*  *for(int i=0;i<n;i++)*  *{*  *int pos=i;*  *for(int j=i+1;j<n;j++)*  *{*  *if(priority[j]<priority[pos])*  *pos=j;*  *}*    *temp=processes[i];*  *priority[i]=priority[pos];*  *priority[pos]=temp;*    *temp=bursttime[i];*  *bursttime[i]=bursttime[pos];*  *bursttime[pos]=temp;*    *temp=processes[i];*  *processes[i]=processes[pos];*  *processes[pos]=temp;*  *}*    *waiting[0]=0; //waiting time for first process is zero*    *//calculate waiting time*  *for(int i=1;i<n;i++)*  *{*  *waiting[i]=0;*  *for(int j=0;j<i;j++)*  *waiting[i]+=bursttime[j];*    *total+=waiting[i];*  *}*    *avwt=total/n; //average waiting time*  *total=0;*    *cout<<"\nProcess\t Burst Time \tWaiting Time\tTurnaround Time";*  *for(int i=0;i<n;i++)*  *{*  *tat[i]=bursttime[i]+waiting[i]; //calculate turnaround time*  *total+=tat[i];*  *cout<<"\nP["<<processes[i]<<"]\t\t "<<bursttime[i]<<"\t\t "<<waiting[i]<<"\t\t\t"<<tat[i];*  *}*    *avtat=total/n; //average turnaround time*  *cout<<"\n\nAverage Waiting Time="<<avwt;*  *cout<<"\nAverage Turnaround Time="<<avtat;*  *}*    *void RR(void)*  *{*  *// initialisation des variables*  *int i, NOP, sum=0,count=0, y, quant, wt=0, tat=0, at[10], bt[10], temp[10];*  *float avg\_wt, avg\_tat;*  *cout<<" Entrer Number of process : ";*  *cin>>NOP;*  *y = NOP; // Assigner le nombre du processus a la variable y*  *for(i=0; i<NOP; i++)*  *{*  *cout<<" Enter Arrival time : "; // temps d'arrive*  *cin>>at[i];*  *cout<<" \nEnter burst time : "; // temps d'execution*  *cin >>bt[i];*  *temp[i] = bt[i]; // enregistrer temps d'execution dans l'array*  *}*  *// Quantum*  *cout<<"Entrer Quantum number :";*  *cin>>quant;*  *// Affichage de :the process No, burst time, Turn Around Time and the waiting time*  *cout<<"\n Process \t\t Burst Time \t\t Turnaround Time \t\t Waiting Time ";*  *for(sum=0, i = 0; y!=0; )*  *{*  *if(temp[i] <= quant && temp[i] > 0) // definir les conditions*  *{*  *sum = sum + temp[i];*  *temp[i] = 0;*  *count=1;*  *}*  *else if(temp[i] > 0)*  *{*  *temp[i] = temp[i] - quant;*  *sum = sum + quant;*  *}*  *if(temp[i]==0 && count==1)*  *{*  *y--; //decrementer le numero du processus*  *cout<<"\nProcess "<<i+1<<" \t\t" << bt[i]<<"\t\t\t\t" << sum-at[i]<<"\t\t\t"<< sum-at[i]-bt[i];*  *wt = wt+sum-at[i]-bt[i];*  *tat = tat+sum-at[i];*  *count =0;*  *}*  *if(i==NOP-1)*  *{*  *i=0;*  *}*  *else if(at[i+1]<=sum)*  *{*  *i++;*  *}*  *else*  *{*  *i=0;*  *}*  *}*  *//Moyenne de Temps d'Attente et Moyenne de Temps de Rotation*  *avg\_wt = wt \* 1.0/NOP;*  *avg\_tat = tat \* 1.0/NOP;*  *printf("\n Average Wait time : \t%f", avg\_wt);*  *printf("\n Average TurnAround Time : \t%f", avg\_tat);*  *}*    *};*  *void RR(void)*  *{*  *// initialisation des variables*  *int i, NOP, sum=0,count=0, y, quant, wt=0, tat=0, at[10], bt[10], temp[10];*  *float avg\_wt, avg\_tat;*  *cout<<" Entrer Number of process : ";*  *cin>>NOP;*  *y = NOP; // Assigner le nombre du processus a la variable y*  *for(i=0; i<NOP; i++)*  *{*  *cout<<" Enter Arrival time : "; // temps d'arrive*  *cin>>at[i];*  *cout<<" \nEnter burst time : "; // temps d'execution*  *cin >>bt[i];*  *temp[i] = bt[i]; // enregistrer temps d'execution dans l'array*  *}*  *// Quantum*  *cout<<"Entrer Quantum number :";*  *cin>>quant;*  *// Affichage de :the process No, burst time, Turn Around Time and the waiting time*  *cout<<"\n Process \t\t Burst Time \t\t Turnaround Time \t\t Waiting Time ";*  *for(sum=0, i = 0; y!=0; )*  *{*  *if(temp[i] <= quant && temp[i] > 0) // definir les conditions*  *{*  *sum = sum + temp[i];*  *temp[i] = 0;*  *count=1;*  *}*  *else if(temp[i] > 0)*  *{*  *temp[i] = temp[i] - quant;*  *sum = sum + quant;*  *}*  *if(temp[i]==0 && count==1)*  *{*  *y--; //decrementer le numero du processus*  *cout<<"\nProcess "<<i+1<<" \t\t" << bt[i]<<"\t\t\t\t" << sum-at[i]<<"\t\t\t"<< sum-at[i]-bt[i];*  *wt = wt+sum-at[i]-bt[i];*  *tat = tat+sum-at[i];*  *count =0;*  *}*  *if(i==NOP-1)*  *{*  *i=0;*  *}*  *else if(at[i+1]<=sum)*  *{*  *i++;*  *}*  *else*  *{*  *i=0;*  *}*  *}*  *//Moyenne de Temps d'Attente et Moyenne de Temps de Rotation*  *avg\_wt = wt \* 1.0/NOP;*  *avg\_tat = tat \* 1.0/NOP;*  *printf("\n Average Wait time : \t%f", avg\_wt);*  *printf("\n Average TurnAround Time : \t%f", avg\_tat);*  *}*  *void sjf(){*  *{*  *int n,temp,tt=0,min,d,i,j;*  *float atat=0,awt=0,stat=0,swt=0;*  *cout<<"enter no of process"<<endl;*  *cin>>n;*  *int a[n],b[n],e[n],tat[n],wt[n];*    *for(i=0;i<n;i++)*  *{*  *cout<<"enter arival time "; //input*  *cin>>a[i];*  *}*  *for(i=0;i<n;i++)*  *{*  *cout<<"enter brust time "; //input*  *cin>>b[i];*  *}*  *for(i=0;i<n;i++)*  *{*  *for(j=i+1;j<n;j++)*  *{*  *if(b[i]>b[j])*  *{*  *temp=a[i];*  *a[i]=a[j];*  *a[j]=temp;*  *temp=b[i];*  *b[i]=b[j];*  *b[j]=temp;*  *}*  *}*  *}*  *min=a[0];*  *for(i=0;i<n;i++)*  *{*  *if(min>a[i])*  *{*  *min=a[i];*  *d=i;*  *}*  *}*  *tt=min;*  *e[d]=tt+b[d];*  *tt=e[d];*  *for(i=0;i<n;i++)*  *{*  *if(a[i]!=min)*  *{*  *e[i]=b[i]+tt;*  *tt=e[i];*  *}*  *}*  *for(i=0;i<n;i++)*  *{*  *tat[i]=e[i]-a[i];*  *stat=stat+tat[i];*  *wt[i]=tat[i]-b[i];*  *swt=swt+wt[i];*  *}*  *atat=stat/n;*  *awt=swt/n;*  *cout<<"Process Arrival-time(s) Burst-time(s) Waiting-time(s) Turnaround-time(s)\n";*  *for(i=0;i<n;i++)*  *{*  *cout<<"P"<<i+1<<" "<<a[i]<<" "<<b[i]<<" "<<wt[i]<<" "<<tat[i]<<endl;*  *}*  *cout<<"awt="<<awt<<" atat="<<atat; //average waiting time and turn around time*  *}*  *}*  *int main(){*  *cpusch a;*  *cout<<"\n\t\t\t1. First In First Out \n\n\t\t\t2. Round Robin\n\n\t\t\t3. Shortest Job First\n\n\t\t\t4. Piority \n\n\t\t\ \n";*  *cout<<"\n\n\t\t\tEnter Your choice !\n";*  *int u;*  *cin>>u;*  *switch(u) {*  *case 1:*  *a.first();*  *a.fcfs();*  *break;*  *case 2:*  *RR();*  *break;*  *case 3:*  *sjf();*  *break;*  *case 4:*  *a.first();*  *a.prioritysch();*  *break;*  *default:*  *cout<<"Invalid Entry ";*  *}*  *}* |



Question 2:

|  |
| --- |
| #include <stdio.h>  #include <string.h>  #include <fcntl.h>  #include <unistd.h>  #define MAX 256  int main() {  FILE \*fptr1, \*fptr2;  int lno, linectr = 0;  char str[MAX],fname[MAX];  char newln[MAX], temp[] = "temp.txt";  static char savefile[MAX];    printf("\n\n REPLACE A WRONGLY WRITTEN INFORMATION IN TEXT FILE WITH A NEW TEXT:\n");  printf("-------------------------------------------------------------\n");  printf(" INPUT THE FILE NAME YOU WANT TO BE OPENED : ");  fgets(fname, MAX, stdin);  fname[strlen(fname) - 1] = '\0';  fptr1 = fopen(fname, "r");  int fd = open(fname, O\_RDONLY);  if (!fptr1) {  printf("ERROR !!!!!!------UNABLE TO OPEN THE INPUT FILE!!\n");  return 0;  }  if(fd != -1) {    printf("-----------------------------EXIXTING DATA FROM FILE WITH WRONG INFORMATION :--------------------------------\n");  printf("\n");  read(fd, savefile, 256);    printf("%s", savefile);  }  fptr2 = fopen(temp, "w");  if (!fptr2) {  printf("ERROR!!!!!-------UNABLE TO OPEN A TEMPORARY FILE TO WRITE !!\n");  fclose(fptr1);  return 0;  }  /\* get the new line from the user \*/  printf(" INPUT THE CORRECT THE CORRECT INFORMATION IN NEW LINE : ");  fgets(newln, MAX, stdin);  /\* get the line number to delete the specific line \*/  printf(" INPUT THE LINE NO. YOU WANT OT REPLACE :");  scanf("%d", &lno);  lno++;  // copy all contents to the temporary file other except specific line  while (!feof(fptr1)) {  strcpy(str, "\0");  fgets(str, MAX, fptr1);  if (!feof(fptr1))  {  linectr++;  if (linectr != lno) {  fprintf(fptr2, "%s", str);  }  else {  fprintf(fptr2, "%s", newln);  }  }  }  fclose(fptr1);  fclose(fptr2);  remove(fname);  rename(temp, fname);  printf("---------DATA REPLACE SUCCESSFULLY---------- \n");  return 0;  } |

Output

