|  |  |
| --- | --- |
|  | **DEPARTMENT OF COMPUTER ENGINEERING** |

**Experiment No. 1**

|  |  |
| --- | --- |
| Semester | S.E-Semester IV – Computer Engineering |
| Subject | Operating System |
| Subject Professor In-charge | Prof. Pankaj Vanwari |
| Assisting Teachers | Prof. Pankaj Vanwari |

|  |
| --- |
| Student Name – Deep Salunkhe |
| Roll Number – 21102A0014 |
| Division and Batch – Division A, Batch 1 |
| Date of Implementation – |
| Experiment Title: FCFS |
| **Theory:** |
|  |

|  |
| --- |
| #include <iostream>  #include <list>  #include <vector>  #include <queue>  #include <algorithm>  using namespace std;  struct ProcessControlBlock  {      int pid;      int BT;      int AT;      int CT;      int waitingtime;      int TAT;  };  vector<ProcessControlBlock> PCB;  vector<ProcessControlBlock> Active;  int sortByarrival(ProcessControlBlock a, ProcessControlBlock b)  {      if (a.AT < b.AT)          return 1;      else          return 0;  }  int sortByburst(ProcessControlBlock a, ProcessControlBlock b)  {      if (a.BT < b.BT)          return 1;      else          return 0;  }  int main()  {      int clock = 0, pcbindex = 0;      int noOfProcess;      cout << "ENTER THE NO. PROCESS =>"           << "\t";      cin >> noOfProcess;      cout << endl;      int n = noOfProcess;      for (int i = 0; i < n; i++)      {          ProcessControlBlock dummy;          cout << "Enter the pid          " << i + 1 << "\t";          cin >> dummy.pid;          cout << "Enter the Burst Time   " << i + 1 << "\t";          cin >> dummy.BT;          cout << "Enter the Arrival Time " << i + 1 << "\t";          cin >> dummy.AT;          cout << endl;          PCB.push\_back(dummy);      }  *//--------------------------------------------Logic for FCFS----------------------------------------//*      sort(PCB.begin(), PCB.end(), sortByarrival);      int envActiveFor1;      int time1 = 0;      int currentpcb1 = 0;      int qfront1 = 0;      float wating1 = 0;      float TurnAround = 0;      cout << "For FCFS for how much time the environment must be active "           << "\t";      cin >> envActiveFor1;      while (envActiveFor1--)      {          while (time1 >= PCB[currentpcb1].AT)          {              Active.push\_back(PCB[currentpcb1]);              currentpcb1 = currentpcb1 + 1;          }          if (Active[0].BT == 0)          {              PCB[qfront1].waitingtime = time1 - PCB[qfront1].BT - PCB[qfront1].AT;              PCB[qfront1].CT = time1;              PCB[qfront1].TAT = time1 - PCB[qfront1].AT;              qfront1 = qfront1 + 1;              Active.erase(Active.begin() + 0);          }          time1++;          Active[0].BT--;      }      cout << "\t\tFCFS" << endl;      cout << "PID"           << "\t"           << "BT"           << "\t"           << "AT"           << "\t"           << "CT"           << "\t"           << "TAT"           << "\t"           << "WT" << endl;      for (int i = 0; i < n; i++)      {          cout << PCB[i].pid << " \t" << PCB[i].BT << "  \t" << PCB[i].AT << "  \t" << PCB[i].CT << "\t" << PCB[i].TAT << "\t" << PCB[i].waitingtime << endl;      }      for (int i = 0; i < n; i++)      {          wating1 = wating1 + PCB[i].waitingtime;          TurnAround = TurnAround + PCB[i].TAT;      }      cout << "Avg Waiting for FCFS:" << (float)(wating1 / n) << endl;      cout << "Avg TAT for FCFS:    " << (float)(TurnAround / n) << endl;      return 0;  } |
| Output: |