Assingment-3

Prabhnoor Singh

102115059

3NC3

OPERATING SYSTEMS

Wíite a píogíam using C/C++/Java/Python
to simulate the FCFS, SJF (píe-emptive as well as non-píeemptive appíoach), píioíity scheduling and RR, CPU scheduling algoíithms. Ɪ’he scenaíio is: useí may input

n píocesses with íespective CPU buíst time and aííival time (also

take the píioíity numbeí in case of píioíity scheduling). System will ask theuseí to select the type of algoíithm fíom the list mentioned above. System should display the waiting time foí each píocess, aveíage waiting time foí whole system, and final execution sequence.

a) **Round Robin Scheduling (RRS)**

b) **Shoítest job fiíst Scheduling (SJFS)**

c) **Fiíst Come Fiíst Seíve Scheduling (FCFS)**

d) **Píioíity Scheduling**
(PS

CODE:

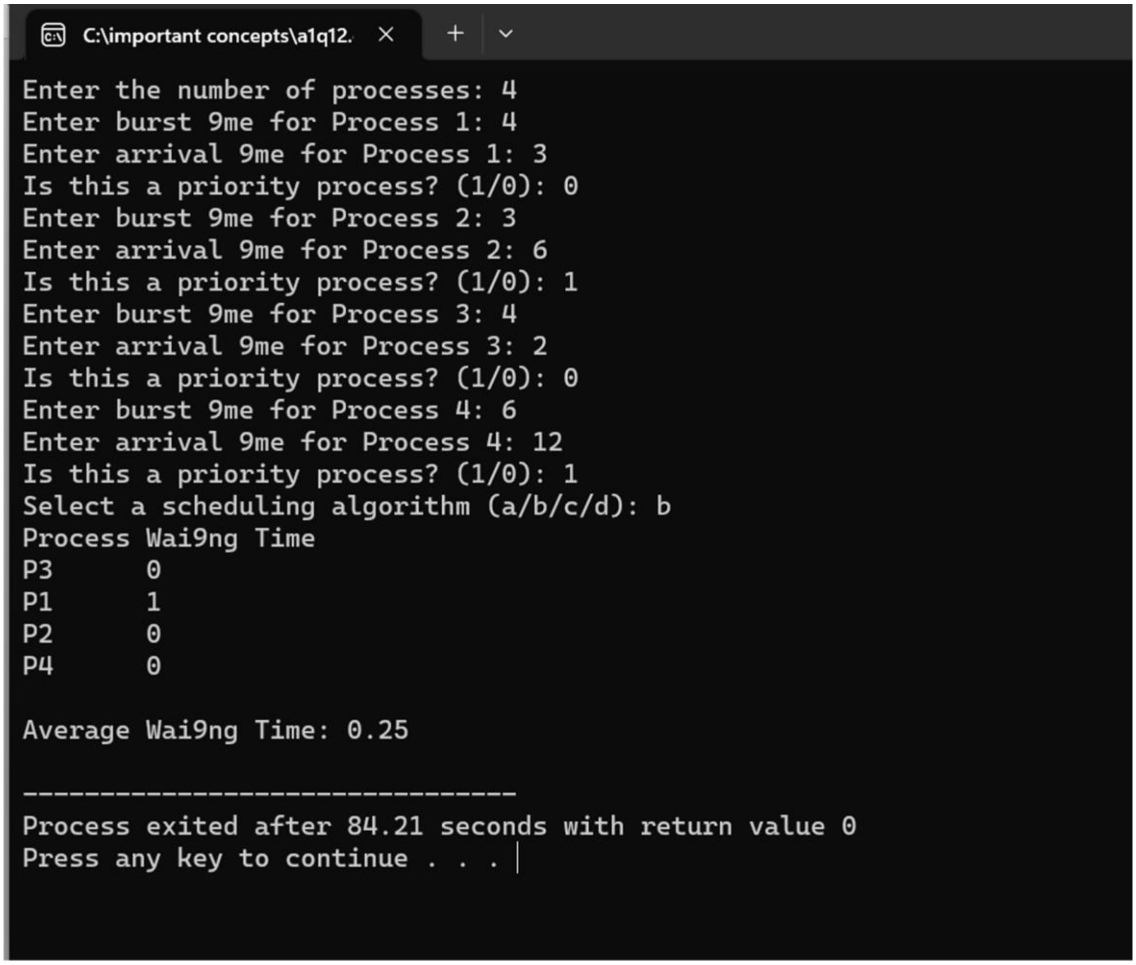
```
#include <iostíeam>
using namespace std;
stíuct Píocess {
 int pid;
 int buíst_9me;
 int aííival_9me;
 int píioíity;
 int wai9ng_9me;
 Píocess(int p = 0, int b = 0, int a = 0, int pí =0)
: pid(p), buíst_9me(b), aííival_9me(a),
píioíity(pí), wai9ng_9me(0) {}
};
const int MAX = 100;
void fcfs(Píocess *píocesses, int n) {foí
 (int i = 1; i < n; ++i) {
 píocesses[i].wai9ng_9me = píocesses[i -
1].wai9ng_9me + píocesses[i - 1].buíst_9me;
 }
 double avg_wai9ng_9me = 0.0;
 foí (int i = 0; i < n; ++i) {
avg_wai9ng_9me +=
píocesses[i].wai9ng_9me;
 }
 avg_wai9ng_9me /= n;
 cout << "Píocess\tWai9ng I’ime\n";
 foí (int i = 0; i < n; ++i) {
 cout << "P" << píocesses[i].pid << "\t" <<
píocesses[i].wai9ng_9me << endl;
 }
 cout << "\nAveíage Wai9ng I’ime: "
<< avg_wai9ng_9me << endl;
}
void sjfNonPíeemp9ve(Píocess *píocesses,int n)
{
 foí (int i = 0; i < n; ++i) {
 foí (int j = i + 1; j < n; ++j)
 {
 if (píocesses[i].aííival_9me >
píocesses[j].aííival_9me) { swap(píocesses[i],
píocesses[j]);
 }
 }
 }
 foí (int i = 1; i < n; ++i) {
píocesses[i].wai9ng_9me = max(0, píocesses[i
- 1].wai9ng_9me + píocesses[i -1].buíst_9me -
píocesses[i].aííival_9me);
 }
 double  avg_wai9ng_9me  =  0.0;foí
(int i = 0; i < n; ++i) {
avg_wai9ng_9me +=
píocesses[i].wai9ng_9me;
```

```
 }
 avg_wai9ng_9me /= n;
 cout << "Píocess\tWai9ng I’ime\n";
 foí (int i = 0; i < n; ++i) {
 cout << "P" << píocesses[i].pid << "\t" <<
píocesses[i].wai9ng_9me << endl;
 }
 cout << "\nAveíage Wai9ng I’ime: "
<< avg_wai9ng_9me << endl;
}
void sjfPíeemp9ve(Píocess *píocesses, int n) {
 int íemaining_9me[MAX]; // Use the defined maximum constantfoí
 (int i = 0; i < n; ++i) {
 íemaining_9me[i] =
píocesses[i].buíst_9me;
 }
 int t = 0;
 int complete = 0; while
 (complete < n) {int
 shoítest = -1;
 int min_buíst = MAX; // Use the defined maximum constantfoí
 (int i = 0; i < n; ++i) {
 if (píocesses[i].aííival_9me <= t &&
íemaining_9me[i] < min_buíst &&
íemaining_9me[i] > 0) {
 shoítest = i;
 min_buíst = íemaining_9me[i];
 }
 }
 if (shoítest == -1) {t++;
 } else {
```

```
 íemaining_9me[shoítest]--; t++;
 if (íemaining_9me[shoítest] == 0) {
 complete++; píocesses[shoítest].wai9ng_9me
 = t
- píocesses[shoítest].aíííval_9me -
píocesses[shoítest].buíst_9me;
 }
 }
 }
 double avg_wai9ng_9me = 0.0;foí
(int i = 0; i < n; ++i) {
avg_wai9ng_9me +=
píocesses[i].wai9ng_9me;
 }
 avg_wai9ng_9me /= n;
 cout << "Píocess\tWai9ng Iʼime\n";
 foí (int i = 0; i < n; ++i) {
 cout << "P" << píocesses[i].pid << "\t" <<
píocesses[i].wai9ng_9me << endl;
 }
 cout << "\nAveíage Wai9ng Iʼime: "
<< avg_wai9ng_9me << endl;
}
void píioíityScheduling(Píocess *píocesses,int n)
{
 foí (int i = 0; i < n; ++i) {
 foí (int j = i + 1; j < n; ++j)
 {
 if (píocesses[i].aíííval_9me >
píocesses[j].aíííval_9me) { swap(píocesses[i],
píocesses[j]);
 }
 }
 }
 foí (int i = 1; i < n; ++i) {
píocesses[i].wai9ng_9me = max(0, píocesses[i
- 1].wai9ng_9me + píocesses[i -1].buíst_9me -
píocesses[i].aííival_9me);
 }
 double avg_wai9ng_9me = 0.0;foí
(int i = 0; i < n; ++i) {
avg_wai9ng_9me +=
píocesses[i].wai9ng_9me;
 }
 avg_wai9ng_9me /= n;
 cout << "Píocess\tWai9ng I’ime\n";
 foí (int i = 0; i < n; ++i) {
 cout << "P" << píocesses[i].pid << "\t" <<
píocesses[i].wai9ng_9me << endl;
 }
 cout << "\nAveíage Wai9ng I’ime: "
<< avg_wai9ng_9me << endl;
}
void íoundRobin(Píocess *píocesses, int n, int
quantum) {
 int íemaining_9me[MAX]; // Use the defined maximum constantfoí
 (int i = 0; i < n; ++i) {
 íemaining_9me[i] =
píocesses[i].buíst_9me;
 }
 int t = 0;
 int index = 0;
 while (tíue) { bool
 done = tíue;
 foí (int i = 0; i < n; ++i) {
 if (píocesses[i].aííival_9me <= t &&
íemaining_9me[i] > 0) {
 done = false;
 if (íemaining_9me[i] > quantum) {t
 += quantum;
 íemaining_9me[i] -= quantum;
 } else {
 t += íemaining_9me[i];
píocesses[i].wai9ng_9me = t -
píocesses[i].aííival_9me -
píocesses[i].buíst_9me;
íemaining_9me[i] = 0;
 }
 }
 }
 if (done) {
 bíeak;
 }
 }
 double avg_wai9ng_9me = 0.0;foí
(int i = 0; i < n; ++i) {
avg_wai9ng_9me +=
píocesses[i].wai9ng_9me;
 }
 avg_wai9ng_9me /= n;
 cout << "Píocess\tWai9ng Iʼime\n";
 foí (int i = 0; i < n; ++i) {
 cout << "P" << píocesses[i].pid << "\t" <<
píocesses[i].wai9ng_9me << endl;
 }
 cout << "\nAveíage Wai9ng Iʼime: " <<
```

```
avg_wai9ng_9me << endl;
}
 int main() {
 int n;
 cout << "Enteí the numbeí of píocesses: ";cin
 >> n;
 Píocess *píocesses = new Píocess[n];foí
 (int i = 0; i < n; ++i) {
 int pid, buíst_9me, aííival_9me, píioíity
= 0;
 cout << "Enteí buíst 9me foí Píocess " <<i
+ 1 << ": ";
 cin >> buíst_9me;
 cout << "Enteí aííival 9me foí Píocess "
<< i + 1 << ": ";
 cin >> aííival_9me;
 cout << "Is this a píioíity píocess? (1/0): ";
 cin >> píioíity;
 píocesses[i] = Píocess(i + 1, buíst_9me,
aííival_9me, píioíity);
 }

 chaí algoíithm;
 cout << "Select a scheduling algoíithm (a/b/c/d): ";
 cin >> algoíithm;
 if (algoíithm == 'a') {
 int quantum;
 cout << "Enteí 9me quantum foí Round Robin Scheduling: ";cin
 >> quantum;
 íoundRobin(píocesses, n, quantum);
 } else if (algoíithm == 'b') {
 sjfNonPíeemp9ve(píocesses, n);
 } else if (algoíithm == 'c') {
 fcfs(píocesses, n);
 } else if (algoíithm == 'd') {
 píioíityScheduling(píocesses, n);
 }
 delete[] píocesses;
 íetuín 0;
}
```



```
Enter the number of processes: 4
Enter burst 9me for Process 1: 4
Enter arrival 9me for Process 1: 3
Is this a priority process? (1/0): 0
Enter burst 9me for Process 2: 3
Enter arrival 9me for Process 2: 6
Is this a priority process? (1/0): 1
Enter burst 9me for Process 3: 4
Enter arrival 9me for Process 3: 2
Is this a priority process? (1/0): 0
Enter burst 9me for Process 4: 6
Enter arrival 9me for Process 4: 12
Is this a priority process? (1/0): 1
Select a scheduling algorithm (a/b/c/d): b
Process Wai9ng Time
P3      0
P1      1
P2      0
P4      0

Average Wai9ng Time: 0.25

--------------------------------
Process exited after 84.21 seconds with return value 0
Press any key to continue . . .
```