Noakhali Science and Technology University
Noakhali-3814.

Online Examination Answer scripts

Year: ..03.... Term: ...01..... Examination, 20..21.........

Course Title: ....Operating System Lab........

Course Code: ....ICE-3106........

Department:........ICE........

Roll no.: ..ASH-1811056M........ Session: ....2017-18........

Total number of pages	:	
Signature of the student	:	Abdul Awal Nadim
Date of examination	:	
Signature of invigilator	:	

1

throughput, avg. WT
avg. TAT

SJF

```
#include<bits/stdc++.h>
using namespace std;
struct os {
        int bt;
        int at;
        int id;
};
bool compare (os a, os b) {
        if (a.bt == b.bt) return a.at < b.at;
        return a.bt < b.bt;
}
int main() {
    int n, at[20], bt[20], ct[20], wt[20], tat[20],
        btt[20], i, j;
    float avwt=0, avtat=0;
    printf ("Enter total number of processes
            (maximum 20):");
    scanf ("%d", &n);

    printf ("\nEnter Process Burst
            Time:\n");
```

Scanned with CamScanner

```
for (i=0; i<n ; i++){
        printf ("p[%d]: ", i+1);
        scanf ("%d", &bt[i]);
        btt[i] = bt[i]

}
~~printf("\nEnter Process Arrival Time:\n");~~
~~for (int i=0;i<n;i++){~~
~~        printf ("p[%d]:", i+1);~~
~~        scanf ("%d", &bt[i]);~~
~~        btt~~
printf ("\nEnter Process Arrival Time:\n")
  for(i=0;i<n; i++){
        printf ("p[%d]: ", i+1);
        scanf ("%d", &at[i]);

  }
  int complete = 0
  for (int time = 0; complete<n;){
  vector <os> que;
  ~~for (in~~ for (int i=0 ; i<n ; i++){
          if (time>=at[i] and
                bt[i] != 0)
          que.push_back({bt[i],at[i]}}.
          }
        sort(que.begin(), que.end(), compare);
        if(que.empty()){ time++ ; continue; }
        int pro=que.front().id;
        bt[pro]--;
        time++;
        if (bt[pro]==0) {
            ct[pro]=time;
            complete++;
        }
    }
printf("\nProcess \t Arrival T \t BT \t CT \tTAT
        ~~\t~~ \t WT\n");
int total_ct=0
 for(i=0; i<n; i++){
     tat[i]=ct[i] - at[i]
     wt[i]= tat[i] - btt[i];
     total_ct +=ct[i];
     avwt += wt[i];
     avtat += tat[i];
     print ("\n P[%d] \t %d \t%d \t %d
          \t %d\t %d", at[i], btt[i],
 }        ct[i], tat[i], wt[i]);
```

```
avwt/=(n*1.0)
avtat/=(n*1.0)
float Throghput =(n/(total_ct*1.0));

cout<<" throput ="<< throghput<<endl;
cout<<"Averge WT ="<<avwt <<endl;
cout<< "Averge TAT ="<< avtat<<endl;

return 0;

}
```

Round Robin

```
#include<bits/stdc++.h>
using namespace std;
struct os {
        int at;
        int id;
};
bool compare(os a, os b){
        if(a.at==b.at) a.id<b.id;
        else return a.at<b.at;
}
int main() {
    int i, j, n, time, remain, flag=0, time_quantum;
    i~~nt ww wt=0, tat=0~~
    int wait_time=0, turnaround_time=0, at[10],
    bt[10], btt[10], ct[10];
    printf("Enter The Number of Total Process:\t");
    cin>>n
    cout<<"Enter AT of Processes :");
    for (i=0 ; i<n ; i++){
        cout<<"P[%d] :", i+1);
        cin>>bt[i];
        btt[i]=bt[i];
    }
```

```
cout<<"Enter Time Quantum"<<endl;
cin>>time__quantum;

bool ses[n+2];
int complete=0; last=-1;
float ava_wait=0, ava_turn=0
queue<int>ready_queue;
for(im=0; complete<n;){
        vector<os>que;
        for(int i=0; i<n; i++)
            if(time>at[i]  and  ses[i]==false){
              que.push_back({at[i], i});

            }
            }else{  time+=bt[pro];
                    bt[pro]=0;  ct[pro]=time;
        }

}
cout<<"Process \t AT \t BT  \t CT, \t TAT
        \t WT"<<endl;
```

```
for (i=0; i<n; i++) {
    tat = ct[i] - at[i]
    wt = ~~Rean~~ tat - btt[i]
    ava_tat ~~+~~ = tat ;
    ava_wt + = wt
   cout << "~~P[tat] \t tat~~ i+1 <<at[i]<< btt[i]
       <<ct[i] << wt [i] ;

}
float troput= n/(time*1.0) ;
cout << "Throghput =" << throghput<<endl;
cout << "Average WT = ", << ava_~~wt~~ <<endl;
cout << Average TAT = " << ava_~~tat~~ <<endl
cout << Av~~erage~~

return 0;

}
```

Priority

```
#include <bits/stdc++.h>
using namespace std;
struct os {
        int bt;
        int at;
        int id;
};
bool compare (os a, os b) {
        if (a.bt == b.bt) return a.at < b.at
        return a.bt < b.bt;
}
int main() {
    int n, at[20], bt[20], ct[20], wt[20], tat[20],
        btt[20], i, j;
    float avwt=0, avtat=0;
    printf ("Enter total number of processes
            (maximum 20):");
    scanf ("%d", &n);

    printf ("\nEnter Process Burst
            Time:\n");
        sort(que.begin(), que.end(), compare);
        if(que.empty()) { time++ ; continue; }
        int pro = que.front().id;
        bt[pro]--;
        time++;
        if (bt[pro]==0) {
            ct[pro] = time;
            complete++;
        }
    }
printf("\nProcess \t Arrival T \t BT \t CT \tTAT
      ~~\t~~ \t WT\n");
int total_ct=0
for(i=0; i<n; i++) {
    tat[i] = ct[i] - at[i]
    wt[i] = tat[i] - btt[i];
    total_ct += ct[i];
    avwt += wt[i];
    avtat += tat [i];
    print ("\n P[%d] \t %d \t%d \t %d
        \t %d\t %d", at[i], btt[i],
        ct[i], tat[i], wt[i]);
}
```

at[i]]];

```
avwt/=(n*1.0)
avtat/=(n*1.0)
float Throghput =(n/(total_ct*1.0));

cout<<"thropnt ="<< throghput<<endl;
cout<<"Averge WT ="<<avwt<<endl;
cout<< "Averge TAT ="<< avtat<<endl;

return 0;

}
```

Scanned with CamScanner

.id ;

Optimal Replacment

```
for (int time ; complete < n ;
{
    vector<os> que ;
    for (int i=0 ; i<n ; i++) {        ue ;
```