```
~/storage $ proot-distro login ubuntu
root@localhost:~# date
Tue Jun 10 02:09:25 PM UTC 2025
root@localhost:~# cal
      June 2025
Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30

root@localhost:~# touch newfile.txt
root@localhost:~# ls
newfile.txt
root@localhost:~# mkdir vk
root@localhost:~# ls
newfile.txt  vk
root@localhost:~# cd vk
root@localhost:~/vk# ls
root@localhost:~/vk# uptime
 19:40:12 up 2 days,  6:52,  load average: 16.86, 19.17, 20.92
root@localhost:~/vk# uname
Linux
root@localhost:~/vk# hostname
localhost
root@localhost:~/vk# bc
bc 1.08.2
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006, 2008, 2012-2018
, 2024 Free Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
1+2*4
9
1/2+5-1
4
^C
(interrupt) Exiting bc.
root@localhost:~/vk# exit
logout
~/storage $ ls
dcim        movies  newfile.txt  shared
downloads   music   pictures
~/storage $
```

ESC	/	—	HOME	↑	END	PGUP
⇆	CTRL	ALT	←	↓	→	PGDN

```
linux@linux-Vostro-220-Series: ~
linux@linux-Vostro-220-Series:~$ touch vk.txt
linux@linux-Vostro-220-Series:~$ nano vk
vk       vk1.txt  vk2.txt  vk.txt   vk.txt~
linux@linux-Vostro-220-Series:~$ nano vk.txt
linux@linux-Vostro-220-Series:~$ nano vk.txt
linux@linux-Vostro-220-Series:~$ grep "is" vk.txt
linux is very powerfull language
linux is easy to learn
unix is easy to handle
unix is better than windows
unix is very reliable
linux is very safe and fast
linux@linux-Vostro-220-Series:~$ grep "unix" vk.txt
unix is easy to handle
unix is better than windows
unix is very reliable
linux@linux-Vostro-220-Series:~$ grep "linux" vk.txt
linux is very powerfull language
linux is easy to learn
linux has very much versions
linux is very safe and fast
linux@linux-Vostro-220-Series:~$ nano vk.txt
linux@linux-Vostro-220-Series:~$
```

```
linux@linux-Vostro-220-Series: ~
linux@linux-Vostro-220-Series:~$ touch sanket.txt
linux@linux-Vostro-220-Series:~$ touch vk.txt
linux@linux-Vostro-220-Series:~$ nano sanket.txt
linux@linux-Vostro-220-Series:~$ nano vk
linux@linux-Vostro-220-Series:~$ nano vk.txt
linux@linux-Vostro-220-Series:~$ nano sanket.txt
linux@linux-Vostro-220-Series:~$ cat sanket.txt vk.txt
this is sanket's file

name : sanket salunkhe
class : sy CSE

this is my file

name : vishal kadam
class : sy CSE
linux@linux-Vostro-220-Series:~$
```

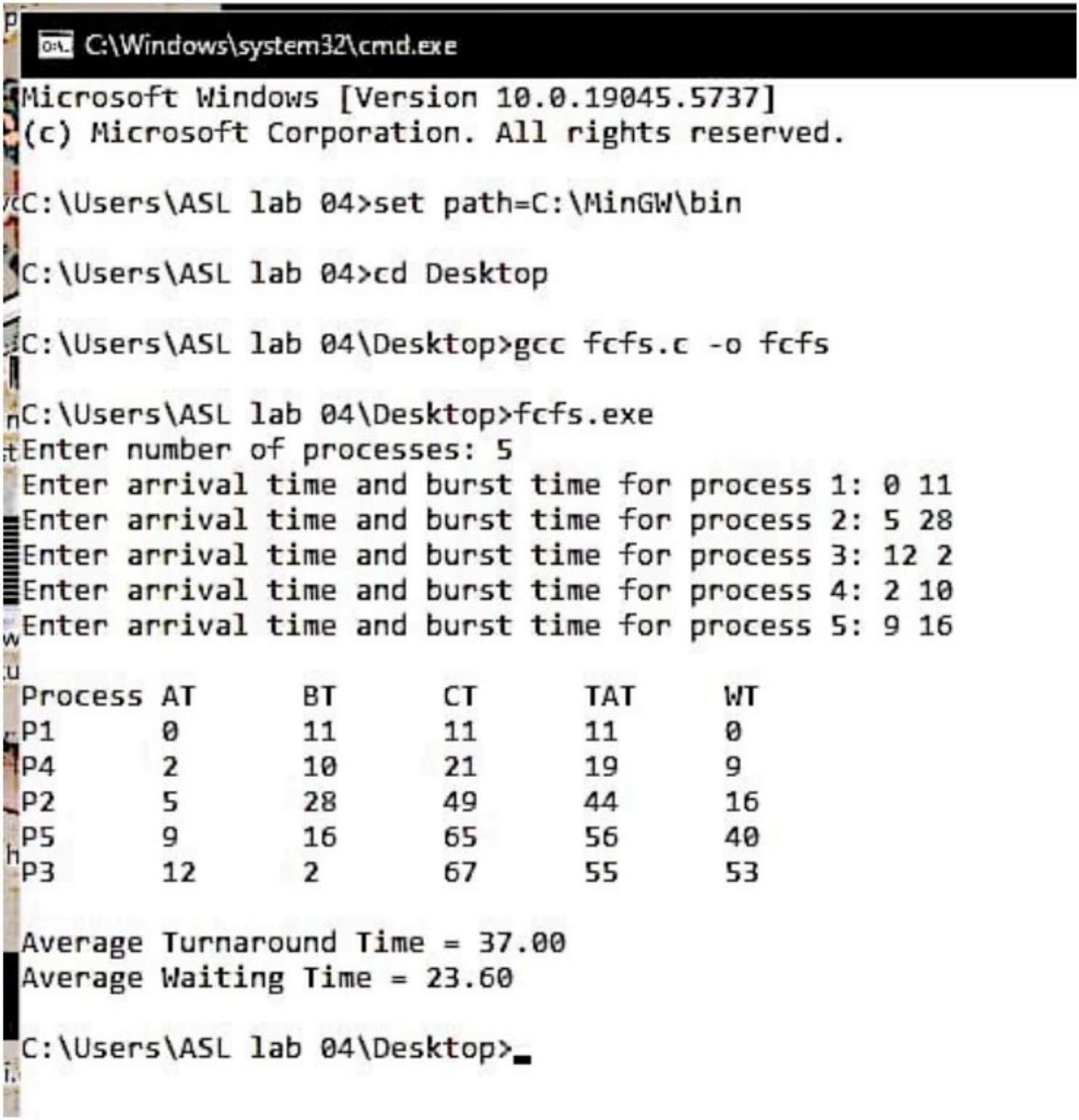
```
linux@linux-Vostro-220-Series: ~
 GNU nano 2.2.6

linux is very powerfull language
linux is easy to learn
unix is easy to handle
unix is better than windows
linux has very much versions
unix is very reliable
linux is very safe and fast
```

```
linux@linux-Vostro-220-Series: ~
linux@linux-Vostro-220-Series:~$ nano abc.txt
linux@linux-Vostro-220-Series:~$ awk '{print}' abc.txt
arvind manager account 45000
sunil clerk account 25000
samarth manager sales 50000
amit manager account 47000
vikas peon sales 15000
dipak clerk sales 23000
sunil peon sales 13000
surya director purchase 80000
sanket shipayi clg 150
linux@linux-Vostro-220-Series:~$ awk '\manager\{print}' abc.txt``
awk: 0: unexpected character '\'
awk: 1: unexpected character '\'
linux@linux-Vostro-220-Series:~$ awk '|manager|{print}' abc.txt``
awk: line 1: syntax error at or near |
linux@linux-Vostro-220-Series:~$ awk '/manager/{print}' abc.txt``
arvind manager account 45000
samarth manager sales 50000
amit manager account 47000
linux@linux-Vostro-220-Series:~$ awk '/clerk/{print}' abc.txt``
sunil clerk account 25000
dipak clerk sales 23000
linux@linux-Vostro-220-Series:~$ awk '/peon/{print}' abc.txt``
vikas peon sales 15000
sunil peon sales 13000
linux@linux-Vostro-220-Series:~$ awk '/shipayi/{print}' abc.txt``
sanket shipayi clg 150
linux@linux-Vostro-220-Series:~$
linux@linux-Vostro-220-Series:~$ awk '/account/{print}' abc.txt``
arvind manager account 45000
sunil clerk account 25000
```

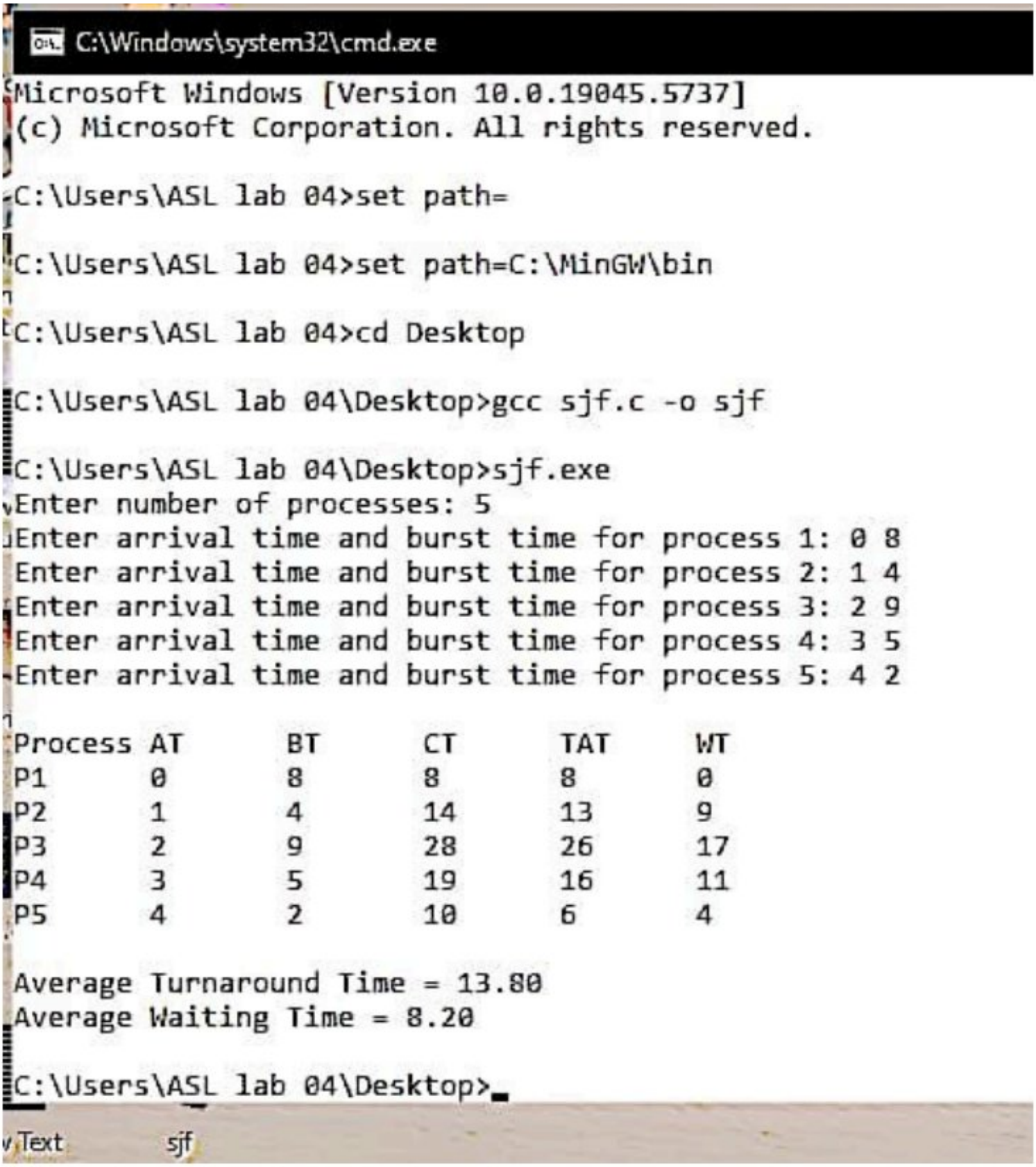
```
linux@linux-Vostro-220-Series:~$ awk '{print $1,$4}' abc.txt
arvind 45000
sunil 25000
samarth 50000
amit 47000
vikas 15000
dipak 23000
sunil 13000
surya 80000
sanket 150
```

```
#include <stdio.h>
struct P { int id, at, bt, ct, tat, wt; };

void sort(struct P p[], int n) {
   for (int i = 0; i < n - 1; i++)
      for (int j = i + 1; j < n; j++)
         if (p[i].at > p[j].at) {
            struct P t = p[i]; p[i] = p[j]; p[j] = t;
     }
}
void main() {
   int n; float tt = 0, tw = 0;
   printf("Enter number of processes: ");
   scanf("%d", &n);
   struct P p[n];
  for (int i = 0; i < n; i++) {
      p[i].id = i + 1;
      printf("Enter Arrival Time and Burst Time for Process %d: ", p[i].id);
      scanf("%d%d", &p[i].at, &p[i].bt);
   }
   sort(p, n);
   int t = 0;
   for (int i = 0; i < n; i++) {
      if (t < p[i].at) t = p[i].at;
      p[i].ct = t + p[i].bt;
      p[i].tat = p[i].ct - p[i].at;
      p[i].wt = p[i].tat - p[i].bt;
      t = p[i].ct;
      tt += p[i].tat;
      tw += p[i].wt;
   }
printf("\nProcess ID\tArrival Time\tBurst Time\tCompletion Time\tTurnaround Time\tWaiting Ti
me\n");
   for (int i = 0; i < n; i++)
      printf("P%d\t\t%d\t\t%d\t\t%d\t\t%d\t\t%d\n",
           p[i].id, p[i].at, p[i].bt, p[i].ct, p[i].tat, p[i].wt);

printf("\nAverage Turnaround Time = %.2f\n", tt / n);
printf("Average Waiting Time = %.2f\n", tw / n);
```


Scanned with OKEN Scanner



```
#include <stdio.h>
struct P { int id, at, bt, ct, tat, wt, d; };
int main() {
  int n, t = 0, c = 0;
  float tt = 0, tw = 0;
  printf("Enter number of processes: ");
  scanf("%d", &n);
  struct P p[n];
  for (int i = 0; i < n; i++) {
    p[i].id = i + 1;
    printf("Enter Arrival Time and Burst Time for Process %d: ", p[i].id);
    scanf("%d%d", &p[i].at, &p[i].bt);
    p[i].d = 0;
  }
  while (c != n) {
    int x = -1, mb = 1e9;
    for (int i = 0; i < n; i++)
      if (!p[i].d && p[i].at <= t && (p[i].bt < mb || (p[i].bt == mb && p[i].at < p[x].at))) {
        mb = p[i].bt;
        x = i;
      }
    if (x != -1) {
      t += p[x].bt;
      p[x].ct = t;
      p[x].tat = t - p[x].at;
      p[x].wt = p[x].tat - p[x].bt;
      p[x].d = 1; c++;
      tt += p[x].tat;
      tw += p[x].wt;
    } else t++;
  }
  printf("\nProcess ID\tArrival Time\tBurst Time\tCompletion Time\tTurnaround Time\tWaiting
Time\n");
  for (int i = 0; i < n; i++)
    printf("P%d\t\t%d\t\t%d\t\t%d\t\t%d\t\t%d\n", p[i].id, p[i].at, p[i].bt, p[i].ct, p[i].tat, p[i].wt);

  printf("\nAverage Turnaround Time = %.2f\nAverage Waiting Time = %.2f\n", tt/n, tw/n);
return 0;
}
```



```
#include <stdio.h>
struct P {
   int id, at, bt, pr, ct, wt, tat, d;
};
int main() {
   int n, t = 0, c = 0;
   float tw = 0, tt = 0;
   printf("Enter number of processes: ");
   scanf("%d", &n);

   struct P p[n];
   for (int i = 0; i < n; i++) {
      p[i].id = i + 1;
      printf("Enter Arrival Time, Burst Time and Priority for Process %d: ", p[i].id);
      scanf("%d%d%d", &p[i].at, &p[i].bt, &p[i].pr);
      p[i].d = 0;
   }

   while (c != n) {
      int x = -1, hp = 1e9;
      for (int i = 0; i < n; i++) {
         if (!p[i].d && p[i].at <= t &&
            (p[i].pr < hp || (p[i].pr == hp && p[i].at < p[x].at))) {
            hp = p[i].pr;
            x = i;
         }
      }
      if (x != -1) {
         t += p[x].bt;
         p[x].ct = t;
         p[x].tat = t - p[x].at;
         p[x].wt = p[x].tat - p[x].bt;
         p[x].d = 1;
         c++;
         tt += p[x].tat;
         tw += p[x].wt;
      } else {
         t++;
      }
```

```
    }

    printf("\nProcess ID\tArrival Time\tBurst Time\tPriority\tCompletion Time\tTurnaround Time\t
Waiting Time\n");
    for (int i = 0; i < n; i++) {
        printf("P%d\t\t%d\t\t%d\t\t%d\t\t%d\t\t%d\t\t%d\n",
            p[i].id, p[i].at, p[i].bt, p[i].pr, p[i].ct, p[i].tat, p[i].wt);
    }

    printf("\nAverage Turnaround Time = %.2f\n", tt / n);
    printf("Average Waiting Time = %.2f\n", tw / n);

    return 0;
}
```

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.5737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ASL lab 04>set path=C:\MinGW\bin

C:\Users\ASL lab 04>cd Desktop

C:\Users\ASL lab 04\Desktop>gcc pri.c -o p

C:\Users\ASL lab 04\Desktop>p.exe
Enter number of processes: 5
Enter Arrival Time, Burst Time and Priority for Process 1: 0 4 1
Enter Arrival Time, Burst Time and Priority for Process 2: 0 3 2
Enter Arrival Time, Burst Time and Priority for Process 3: 6 7 1
Enter Arrival Time, Burst Time and Priority for Process 4: 11 4 3
Enter Arrival Time, Burst Time and Priority for Process 5: 12 2 2

PID     AT      BT      PRI     CT      TAT     WT
1       0       4       1       4       4       0
2       0       3       2       7       7       4
3       6       7       1       14      8       1
4       11      4       3       20      9       5
5       12      2       2       16      4       2

Average Turnaround Time: 6.40
Average Waiting Time: 2.40
```

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>

#define SIZE 10

int buffer[SIZE], i = 0, j = 0;
sem_t empty, full;
pthread_mutex_t mutex;

void* prod(void* arg) {
   int item = 0;
   while (1) {
      item++;
      sem_wait(&empty);
      pthread_mutex_lock(&mutex);

      buffer[i] = item;
      printf("Produced: %d at %d\n", item, i);
      i = (i + 1) % SIZE;

      pthread_mutex_unlock(&mutex);
      sem_post(&full);
      sleep(1);
   }
   return NULL;
}

void* cons(void* arg) {
   while (1) {
      sem_wait(&full);
      pthread_mutex_lock(&mutex);

      int item = buffer[j];
      printf("Consumed: %d from %d\n", item, j);
      j = (j + 1) % SIZE;
```

```
        pthread_mutex_unlock(&mutex);
        sem_post(&empty);
        sleep(2);
    }
    return NULL;
}

int main() {
    pthread_t PthreadID, CthreadID;

    sem_init(&empty, 0, SIZE);
    sem_init(&full, 0, 0);
    pthread_mutex_init(&mutex, NULL);

    pthread_create(&PthreadID, NULL, prod, NULL);
    pthread_create(&CthreadID, NULL, cons, NULL);

    pthread_join(PthreadID, NULL);
    pthread_join(CthreadID, NULL);

    sem_destroy(&empty);
    sem_destroy(&full);
    pthread_mutex_destroy(&mutex);

    return 0;
}
```

```
l4-Vostro-230: ~                                                    En  2:48 P
asl4@asl4-Vostro-230:~$ gcc semaphore.c -o algo -pthread
asl4@asl4-Vostro-230:~$ ./algo
Produced: 1 at 0
Consumed: 1 from 0
Produced: 2 at 1
Consumed: 2 from 1
Produced: 3 at 2
Produced: 4 at 3
Consumed: 3 from 2
Produced: 5 at 4
Produced: 6 at 5
Consumed: 4 from 3
Produced: 7 at 6
Produced: 8 at 7
Consumed: 5 from 4
Produced: 9 at 8
Produced: 10 at 9
Consumed: 6 from 5
Produced: 11 at 0
Produced: 12 at 1
Consumed: 7 from 6
Produced: 13 at 2
Produced: 14 at 3
Consumed: 8 from 7
Produced: 15 at 4
Produced: 16 at 5
```

```c
#include <stdio.h>

int i, j;

void firstfit(int b[], int m, int p[], int n)
{
    int a[n], o[m];

    for(i = 0; i < n; i++)
        a[i] = -1;

    for(i = 0; i < m; i++)
        o[i] = 0;

    for(i = 0; i < n; i++)
    {
        for(j = 0; j < m; j++)
        {
            if(!o[j] && b[j] >= p[i])
            {
                a[i] = j;
                o[j] = 1;
                break;
            }
        }
    }

    printf("\nProc\tSize\tBlock\n");
    for(i = 0; i < n; i++)
    {
        printf("%d\t%d\t", i + 1, p[i]);
        if(a[i] != -1)
            printf("%d\n", a[i] + 1);
        else
            printf("NA\n");
    }
}

void main()
{
    int b[] = {500, 200, 800, 400, 100, 700, 300, 600, 1000, 900};
    int p[] = {212, 150, 375, 950, 350, 632, 400, 717, 811};
    int m = sizeof(b) / sizeof(b[0]);
    int n = sizeof(p) / sizeof(p[0]);

    firstfit(b, m, p, n);
}
```

```
#include <stdio.h>

int i, j;

void bestfit(int b[], int m, int p[], int n) {
    int a[n], o[m];
    for(i = 0; i < n; i++) a[i] = -1;
    for(i = 0; i < m; i++) o[i] = 0;

    for(i = 0; i < n; i++) {
        int x = -1;
        for(j = 0; j < m; j++) {
            if(b[j] >= p[i] && !o[j]) {
                if(x == -1 || b[j] < b[x]) x = j;
            }
        }
        if(x != -1) {
            a[i] = x;
            o[x] = 1;
        }
    }

    printf("\nP.No\tP.Size\tB.No\n");
    for(i = 0; i < n; i++) {
        printf("%d\t%d\t", i+1, p[i]);
        if(a[i] != -1)
            printf("%d\n", a[i]+1);
        else
            printf("NA\n");
    }
}

void main() {
    int b[] = {500, 200, 800, 400, 100, 700, 300, 600, 1000, 900};
    int p[] = {212, 150, 375, 950, 350, 632, 400, 717, 811};
    int m = sizeof(b)/sizeof(b[0]);
    int n = sizeof(p)/sizeof(p[0]);
    bestfit(b, m, p, n);

}
```

```c
#include <stdio.h>

void worstFit(int b[], int m, int p[], int n) {
    int alloc[n], i, j;
    for (i = 0; i < n; i++) alloc[i] = -1;

    for (i = 0; i < n; i++) {
        int idx = -1;
        for (j = 0; j < m; j++)
            if (b[j] >= p[i] && (idx == -1 || b[j] > b[idx]))
                idx = j;

        if (idx != -1) {
            alloc[i] = idx;
            b[idx] -= p[i];
        }
    }

    printf("Process No.\tSize\tBlock\n");
    for (i = 0; i < n; i++)
        printf("%d\t\t%d\t%s\n", i + 1, p[i], alloc[i] != -1 ? (char[4]){alloc[i] +
1 + '0', '\0'} : "NA");
}

int main() {
    int b[] = {500, 200, 800, 400, 100, 700, 300, 600, 1000, 900};
    int p[] = {212, 150, 375, 950, 350, 632, 400, 717, 811};
    worstFit(b, sizeof(b)/sizeof(b[0]), p, sizeof(p)/sizeof(p[0]));
    return 0;
}
```


```
asl4@asl4-Vostro-230: ~/Documents/expt8
asl4@asl4-Vostro-230:~/Documents/expt8$ gcc firstfit.c -o first
asl4@asl4-Vostro-230:~/Documents/expt8$ ./first

Process No.     Process Size    Block no.
1                       212                     1
2                       150                     2
3                       375                     3
4                       950                     9
5                       350                     4
6                       632                     6
7                       400                     8
8                       717                     10
9                       811                     Not Allocated
asl4@asl4-Vostro-230:~/Documents/expt8$ gcc bestfit.c -o best
asl4@asl4-Vostro-230:~/Documents/expt8$ ./best

Process No.     Process Size    Block no.
1                       212                     7
2                       150                     2
3                       375                     4
4                       950                     9
5                       350                     1
6                       632                     6
7                       400                     8
8                       717                     3
9                       811                     10
asl4@asl4-Vostro-230:~/Documents/expt8$ gcc worstfit.c -o worst
stro-230:~/Documents/expt8$ ./worst

Process No.     Process Size    Block no.
1                       212                     7
2                       150                     2
3                       375                     4
4                       950                     9
5                       350                     1
6                       632                     6
7                       400                     8
8                       717                     3
9                       811                     10
asl4@asl4-Vostro-230:~/Documents/expt8$
```

1. PROGRAM FOR FIF0 PAGE REPLACEMENT

```
#include <stdio.h>
void main() {
   int a[] = {4,1,2,4,5}, f[3] = {-1,-1,-1}, i, j, k = 0, pf = 0;
   printf("Page\tF1\tF2\tF3\n");
   for (i = 0; i < 5; i++) {
      int hit = 0;
      for (j = 0; j < 3; j++) if (f[j] == a[i]) hit = 1;
      if (!hit) { f[k] = a[i]; k = (k + 1) % 3; pf++; }
      printf("%d\t", a[i]);
      for (j = 0; j < 3; j++) ( f[j] == -1 ) ? printf("-\t") : printf("%d\t", f[j]);
      printf("\n");
   }
   printf("Total Page Faults: %d\n", pf);
}
```

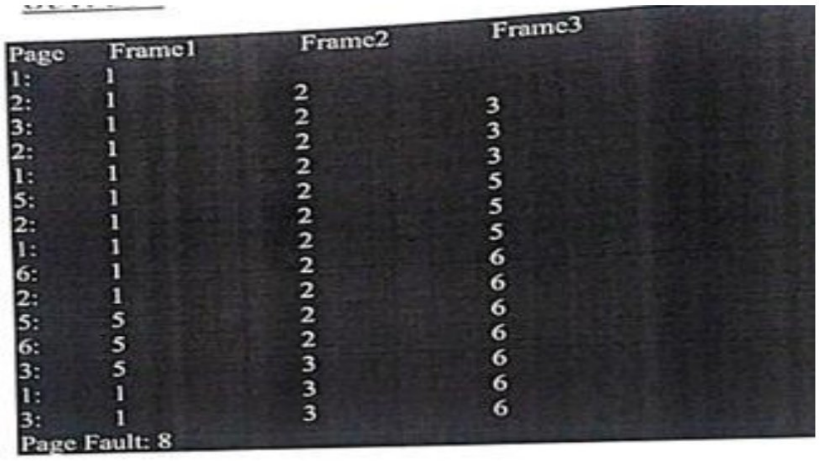
Output:

2. PROGRAM FOR LRU PAGE REPLACEMENT

```
#include <stdio.h>
#include <limits.h>
int checkHit(int p, int q[], int occ) {
   for (int i = 0; i < occ; i++)
      if (p == q[i]) return 1;
   return 0;
}
void printFrame(int q[], int occ) {
   for (int i = 0; i < 3; i++)
```

```
        i < occ ? printf("%d\t", q[i]) : printf("-\t");
    printf("\n");
}
void main() {
    int in[] = {1,2,3,2,1,5,2,1,6,2,5,6,3,1,3}, q[3], d[3], occ = 0, pf = 0;
    int n = sizeof(in)/sizeof(in[0]);
    printf("Page\tF1\tF2\tF3\n");
    for (int i = 0; i < n; i++) {
        printf("%d\t", in[i]);
        if (checkHit(in[i], q, occ)) { printFrame(q, occ); continue; }
        if (occ < 3) {
            q[occ++] = in[i]; pf++;
            printFrame(q, occ); continue;
        }
        for (int j = 0; j < 3; j++) {
            d[j] = 0;
            for (int k = i - 1; k >= 0; k--) {
                d[j]++;
                if (q[j] == in[k]) break;
            }
        }
        int max = 0;
        for (int j = 1; j < 3; j++)
            if (d[j] > d[max]) max = j;
        q[max] = in[i]; pf++;
        printFrame(q, occ);
    }    printf("Page Faults: %d\n", pf);
}
```

Output:

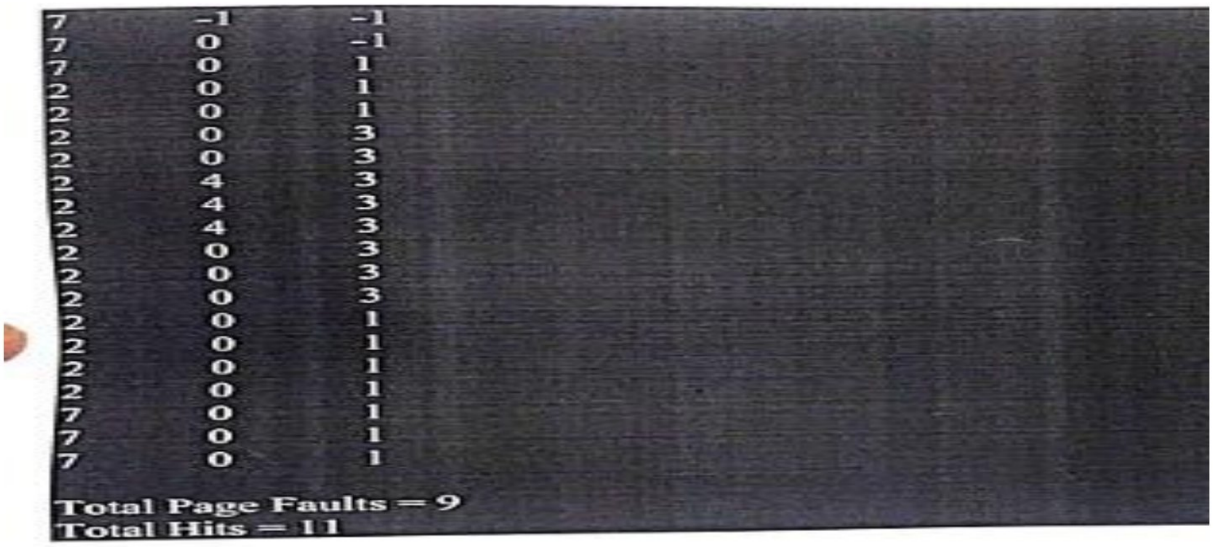


3. **PROGRAM FOR OPTIMAL PAGE REPLACEMENT**

```
#include <stdio.h>
void main() {
   int in[] = {7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1};
   int f[3], t[3], n = 20, faults = 0, i, j, k, pos, max;
   for (i = 0; i < 3; i++) f[i] = -1;
   for (i = 0; i < n; i++) {
      int hit = 0;
      for (j = 0; j < 3; j++)
         if (f[j] == in[i]) { hit = 1; break; }
      if (!hit) {
         int empty = -1;
         for (j = 0; j < 3; j++)
            if (f[j] == -1) { f[j] = in[i]; faults++; empty = 1; break; }
         if (!empty) {
            for (j = 0; j < 3; j++) {
               t[j] = -1;
               for (k = i + 1; k < n; k++)
                  if (f[j] == in[k]) { t[j] = k; break; }
            }
            for (j = 0; j < 3; j++)
               if (t[j] == -1) { pos = j; break; }
               else {
                  max = t[0]; pos = 0;
                  for (j = 1; j < 3; j++)
```

```
                if (t[j] > max) { max = t[j]; pos = j; }
            break;
        }
      f[pos] = in[i]; faults++;
    }
  }
  for (j = 0; j < 3; j++) f[j] == -1 ? printf("-\t") : printf("%d\t", f[j]);
  printf("\n");
 }
 printf("\nTotal Page Faults = %d", faults);
 printf("\nTotal Hits = %d\n", n - faults);
}
```

Output:



1. PROGRAM OF FCFS SHEDULING:

```
#include<stdio.h>

#include<conio.h>

void main(){

int i,j,sum=0,n;

 int ar[20],tm[20];

 int disk;

printf("enter number of location\t");

scanf("%d",&n);
printf("enter position of head t");
scanf("%d",&disk);
printf("enter elements of disk queue\n");
for(i=0;i<n;i++){
 scanf("%d",&ar[i]);

 }

 for(i=0;i<n;i++) {

 tm[i]=disk-ar[i];

if(tm[i]<0) {    tm[i]=ar[i]-disk;          }

disk=ar[i];

sum=sum+tm[i];

 }

printf("\nmovement of total cylinders %d",sum);

getch();

}
```

Output:

Enter number of requests: **8**

Enter initial head position: **53**

Enter the request sequence**: 95 180 34 119 11 123 62 64**

Movement of total cylinders : **641**

2. PROGRAM OF SCAN SCHEDULING

```
#include<stdio.h>

#include<stdlib.h>
```

```
void main() {
int RQ[100],i,j,n,TotalHeadMoment=0,initial,size,move;
 printf("Enter the number of Requests\n");
 scanf("%d",&n);
 printf("Enter the Requests sequence\n");
 for(i=0;i<n;i++)
 scanf("%d",&RQ[i]);
 printf("Enter initial head position\n");
 scanf("%d",&initial);
 printf("Enter total disk size\n");
 scanf("%d",&size);
 printf("Enter the head movement direction for high 1 and for low 0\n");
 scanf("%d",&move);
for(i=0;i<n;i++){
   for(j=0;j<n-i-1;j++)
   {
      if(RQ[j]>RQ[j+1])
      {
         int temp;
                  temp = RQ[j];
                  RQ[j] = RQ[j+1];
                  RQ[j+1] = temp;
         }        }        }
   int index;
   for(i=0;i<n;i++){
      if(initial<RQ[i])  {
         index=i;
         break;
      }  }
   if(move==1){
      for(i=index;i<n;i++){
         TotalHeadMoment=TotalHeadMoment+abs(RQ[i]-initial);
         initial=RQ[i];
      }
```

```
        TotalHeadMoment=TotalHeadMoment+abs(size-RQ[i-1]-1);
        initial=size-1;
        for(i=index-1;i>=0;i--){
        TotalHeadMoment=TotalHeadMoment+abs(RQ[i]-initial);
        initial=RQ[i];
    }     }
else{
    for(i=index-1;i>=0;i--){
        TotalHeadMoment=TotalHeadMoment+abs(RQ[i]-initial);
        initial=RQ[i];
    }
    TotalHeadMoment=TotalHeadMoment+abs(RQ[i+1]-0);
    initial=0;
    for(i=index;i<n;i++){
        TotalHeadMoment=TotalHeadMoment+abs(RQ[i]-initial);
        initial=RQ[i];
    }     }
printf("Total head movement is %d",TotalHeadMoment);
}
```

Output:

Enter number of requests: **8**

Enter the request sequence: **95 180 34 119 11 123 62 64**

Enter initial head position: **50**

Enter total disk size: **200**

Enter movement direction for high 1 and for low 0: **1**

Total head movement is: **337**

3. PROGRAM OF Circular SCAN (C-SCAN)

```
#include <stdio.h>
#include <stdlib.h>

void main() {
```

```
    int RQ[] = {95,180,34,119,11,123,62,64}, n = 8;
    int head = 50, size = 200, move = 1, total = 0;

    for (int i = 0; i < n-1; i++)
        for (int j = 0; j < n-i-1; j++)
            if (RQ[j] > RQ[j+1]) {
                int t = RQ[j]; RQ[j] = RQ[j+1]; RQ[j+1] = t;
            }

    int i, idx;
    for (i = 0; i < n; i++) if (head < RQ[i]) { idx = i; break; }

    if (move) {
        for (i = idx; i < n; i++) total += abs(RQ[i] - head), head = RQ[i];
        total += abs(size - head - 1) + (size - 1);
        head = 0;
        for (i = 0; i < idx; i++) total += abs(RQ[i] - head), head = RQ[i];
    } else {
        for (i = idx-1; i >= 0; i--) total += abs(RQ[i] - head), head = RQ[i];
        total += abs(head - 0) + (size - 1);
        head = size - 1;
        for (i = n-1; i >= idx; i--) total += abs(RQ[i] - head), head = RQ[i];
    }

    printf("Total head movement: %d\n", total);
}
```

Output:

```
Total head movement: 382
```