Assignment – 1B

- **Address Book for Students**

Code :

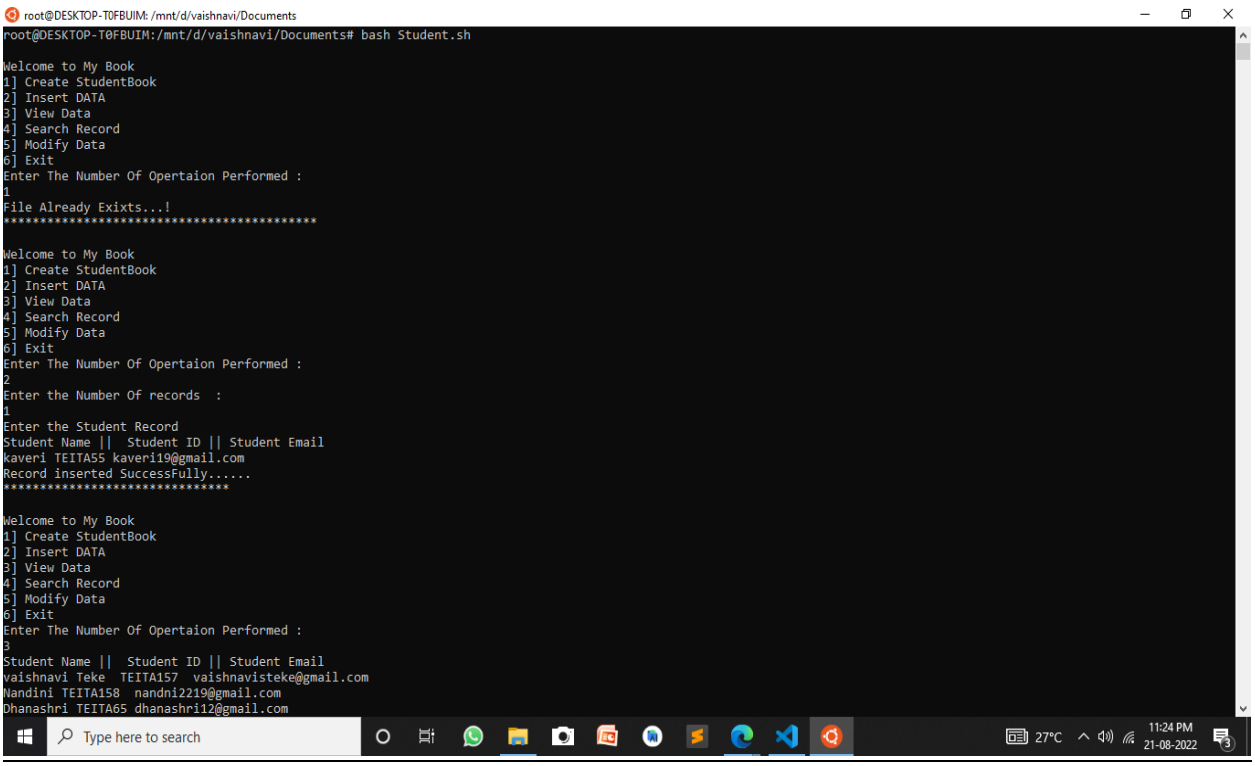
```
#!/bin/bash
createbook()
{
    if [ -e Student.txt ]
    then
    echo "File Already Exixts...!"
    echo "****************************************"
    else
    touch Student.txt
    echo "File Created Sucessfully.....!"
    echo "****************************************"
fi
}
#insert
Insertrec()
{
    echo "Enter the Number Of records  : "
    read n
    while [ $n -gt 0 ]
    do
    echo "Enter the Student Record"
    echo "Student Name ||  Student ID || Student Email"
    read name
    echo "$name">>Student.txt
```

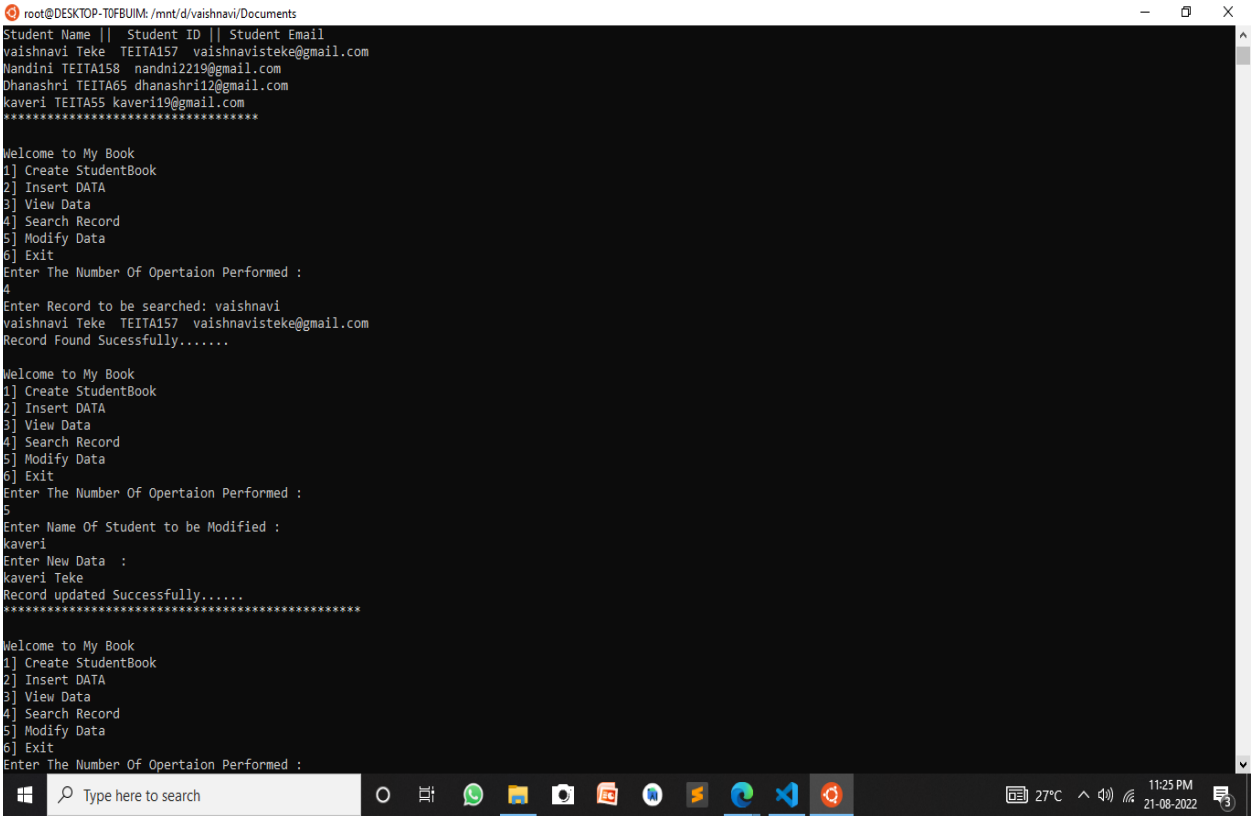
```
        echo "Record inserted SuccessFully....."
        echo "*******************************"
        ((n--))
        done
 }
viewrec()
 {
       if [ ! -e Student.txt ]
        then
        echo "File Not Found......."
        echo "*********************************"
        else
        echo "Student Name ||  Student ID || Student Email "
        cat Student.txt
        echo "***********************************"
        fi
 }
searchrec()
 {
       read -p "Enter Record to be searched: " re
        if [ -f Student.txt ]
        then
        result=$(grep -i "$re" Student.txt)
        if [ "$result" ]
        then
        echo "$result"
        echo "Record Found Sucessfully......."
        else
        echo "Record Not Found....."
```

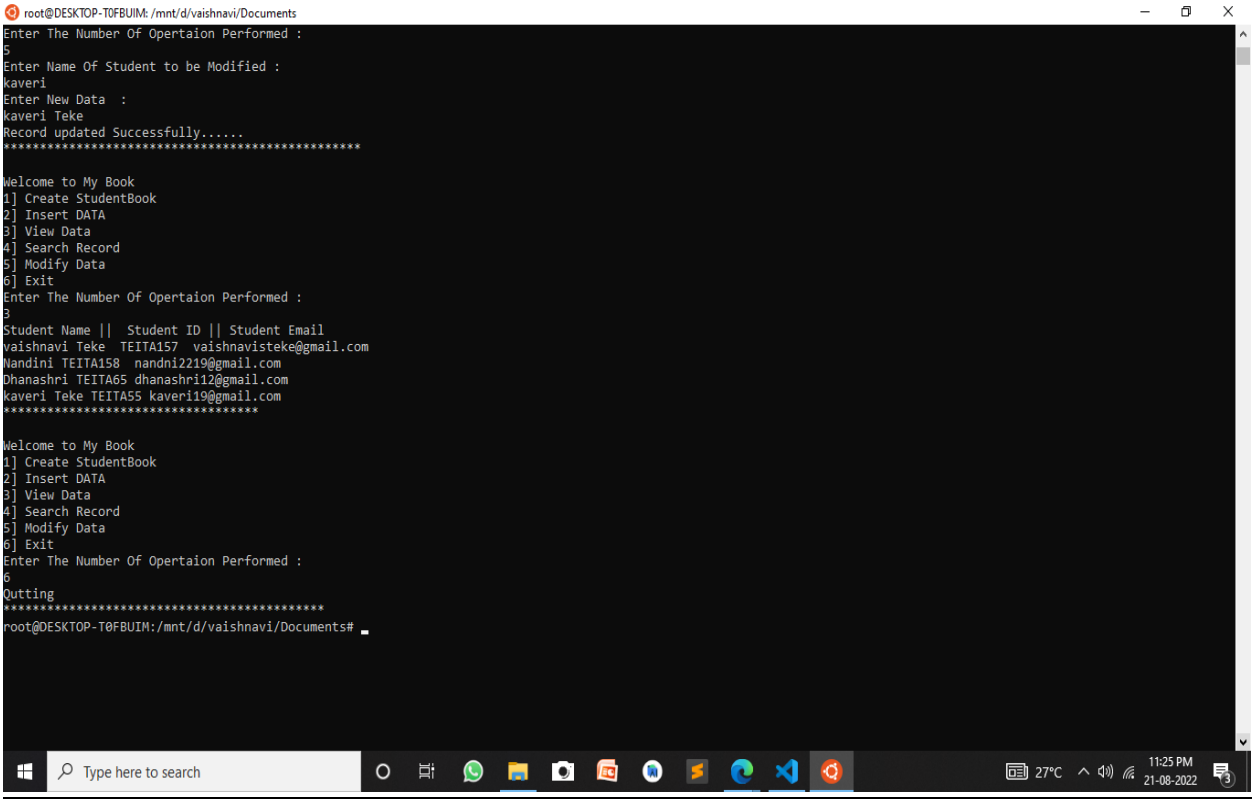
```
        fi
        fi
}
modifyrec()
{
        echo "Enter Name Of Student to be Modified :"
        read name
        find=$(grep -i "$name" Student.txt)
        if [ "$find" ]
        then
        echo "Enter New Data  :"
        read mod
        sed -i "s/$name/$mod/" Student.txt
        echo "Record updated Successfully....."
        echo "**********************************************"
        else
        echo "Recoed not Found.....!"
        fi
}
while true
do
echo "  "
echo "Welcome to My Book"
echo "1] Create StudentBook"
echo "2] Insert DATA"
echo "3] View Data"
echo "4] Search Record"
echo "5] Modify Data"
echo "6] Exit "
```

```
echo "Enter The Number Of Opertaion Performed :"
read choice

case $choice in
1) createbook ;;
2) Insertrec ;;
3) viewrec ;;
4) searchrec ;;
5) modifyrec ;;
6) echo "Qutting"
    echo "****************************************"
    break;;
    echo "Invalid choice ....."
    echo "************************************************************"
    ;;
    esac
    done
```

- Output :


root@DESKTOP-T0FBUIM: /mnt/d/vaishnavi/Documents
root@DESKTOP-T0FBUIM:/mnt/d/vaishnavi/Documents# bash Student.sh
Welcome to My Book
1] Create StudentBook
2] Insert DATA
3] View Data
4] Search Record
5] Modify Data
6] Exit
Enter The Number Of Opertaion Performed :
1
File Already Exixts...!
Welcome to My Book
1] Create StudentBook
2] Insert DATA
3] View Data
4] Search Record
5] Modify Data
6] Exit
Enter The Number Of Opertaion Performed :
2
Enter the Number Of records :
1
Enter the Student Record
Student Name || Student ID || Student Email
kaveri TEITA55 kaveri19@gmail.com
Record inserted SuccessFully.....
Welcome to My Book
1] Create StudentBook
2] Insert DATA
3] View Data
4] Search Record
5] Modify Data
6] Exit
Enter The Number Of Opertaion Performed :
3
Student Name || Student ID || Student Email
vaishnavi Teke TEITA157 vaishnavisteke@gmail.com
Nandini TEITA158 nandni2219@gmail.com
Dhanashri TEITA65 dhanashri12@gmail.com
Type here to search
27°C
11:24 PM
21-08-2022


root@DESKTOP-T0FBUIM: /mnt/d/vaishnavi/Documents
Student Name || Student ID || Student Email
vaishnavi Teke TEITA157 vaishnavisteke@gmail.com
Nandini TEITA158 nandni2219@gmail.com
Dhanashri TEITA65 dhanashri12@gmail.com
kaveri TEITA55 kaveri19@gmail.com
Welcome to My Book
1] Create StudentBook
2] Insert DATA
3] View Data
4] Search Record
5] Modify Data
6] Exit
Enter The Number Of Opertaion Performed :
4
Enter Record to be searched: vaishnavi
vaishnavi Teke TEITA157 vaishnavisteke@gmail.com
Record Found Sucessfully.....
Welcome to My Book
1] Create StudentBook
2] Insert DATA
3] View Data
4] Search Record
5] Modify Data
6] Exit
Enter The Number Of Opertaion Performed :
5
Enter Name Of Student to be Modified :
kaveri
Enter New Data :
kaveri Teke
Record updated Successfully.....
Welcome to My Book
1] Create StudentBook
2] Insert DATA
3] View Data
4] Search Record
5] Modify Data
6] Exit
Enter The Number Of Opertaion Performed :
Type here to search
27°C
11:25 PM
21-08-2022


root@DESKTOP-T0FBUIM: /mnt/d/vaishnavi/Documents
Enter The Number Of Opertaion Performed :
5
Enter Name Of Student to be Modified :
kaveri
Enter New Data :
kaveri Teke
Record updated Successfully.....
****
Welcome to My Book
1] Create StudentBook
2] Insert DATA
3] View Data
4] Search Record
5] Modify Data
6] Exit
Enter The Number Of Opertaion Performed :
3
Student Name || Student ID || Student Email
vaishnavi Teke TEITA157 vaishnavisteke@gmail.com
Nandini TEITA158 nandni2219@gmail.com
Dhanashri TEITA65 dhanashri12@gmail.com
kaveri Teke TEITA55 kaveri19@gmail.com
****
Welcome to My Book
1] Create StudentBook
2] Insert DATA
3] View Data
4] Search Record
5] Modify Data
6] Exit
Enter The Number Of Opertaion Performed :
6
Qutting
****
root@DESKTOP-T0FBUIM:/mnt/d/vaishnavi/Documents#
Type here to search
27°C
11:25 PM
21-08-2022

Assignment – 2

- **Orphan State :**

```
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
#include<sys/wait.h>
#include<stdlib.h>
// Quick Sort
int partition(int arr[],int l,int h)
{
   int pivot=arr[l];
   int i=l;
   int j=h;
   while(i<j)
   {
      while(arr[i]<=pivot)
      {
         i++;
      }
      while(arr[j]>pivot)
      {
         j--;
      }
      if(i<j)
      {
         int temp=arr[i];
         arr[i]=arr[j];
         arr[j]=temp;
      }
   }
   int temp=arr[l];
   arr[l]=arr[j];
   arr[j]=temp;
   return j;
}
void quickSort(int arr[],int l,int h)
{
   if(l<h)
   {
      int pivot=partition(arr,l,h);
      quickSort(arr,l,pivot);
      quickSort(arr,pivot+1,h);
   }
}
//Merge Sort
void merge(int arr[],int l, int mid, int h)
```

```
{
   int i,j,k;
   int b[h-l+1];
  for(i=l,j=mid+1,k=l; i<=mid && j<=h; k++ )
  {
      if(arr[i]<arr[j])
      {
         b[k]=arr[i++];
      }
      else
      {
         b[k]=arr[j++];
      }

   }
      while (j<=h)
      {
         b[k++]=arr[j++];
      }
      while(i<=mid)
      {
         b[k++]=arr[i++];
      }
   for(k=l; k<=h; k++)
   {
      arr[k]=b[k];
   }
}
void mergeSort(int arr[],int l, int h)
{
   if(l<h)
   {
   int mid=(l+h)/2;
   mergeSort(arr,l,mid);
   mergeSort(arr,mid+1,h);
   merge(arr,l,mid,h);
   }

}
int main()
{
   int n;
   printf("Enter size of array:\n");
   scanf("%d",&n);
   int arr[n];
   printf("Enter array elements:\n");
   for(int i=0; i<n; i++)
   {
```

```
    scanf("%d",&arr[i]);
  }

  //Orphan state
  int pid;
  pid=fork();
  if(pid==0)
  {
    int stc=30;
    wait(&stc);
    printf("Child process with id %d\n",getpid());
    quickSort(arr,0,n);
    printf("Array After Sorting:\n");
    for(int i=1; i<=n; i++)
    {
      printf("%d\n",arr[i]);
    }
  }
  else
  {
    printf("Parent process with id %d\n",getpid());
    mergeSort(arr,0,n);
    printf("Array After Sorting:\n");
    for(int i=1; i<=n; i++)
    {
      printf("%d\n",arr[i]);
    }
  }
  return 0;
}
```

- Zombie State:

```
#include<stdio.h>

#include<unistd.h>

#include<sys/types.h>

#include<sys/wait.h>

#include<stdlib.h>

// Quick Sort

int partition(int arr[],int l,int h)

{
```

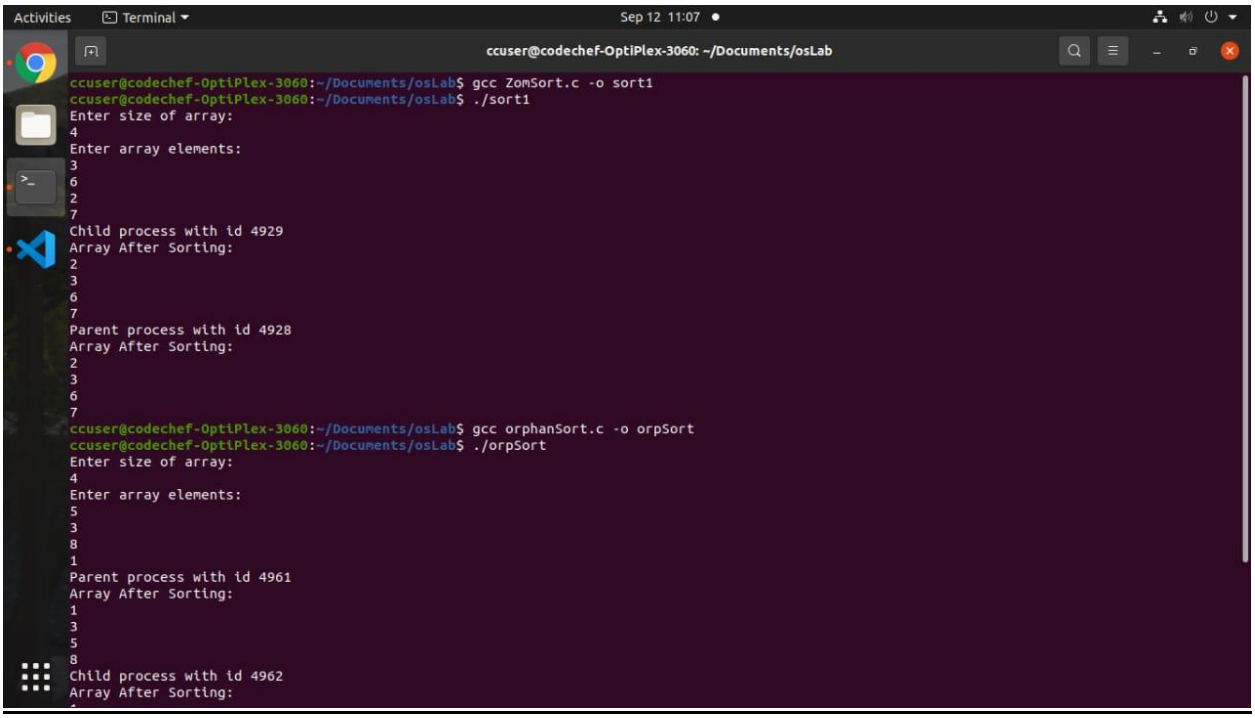
```
    int pivot=arr[l];
    int i=l;
    int j=h;
    while(i<j)
    {
        while(arr[i]<=pivot)
        {
            i++;
        }
        while(arr[j]>pivot)
        {
            j--;
        }
        if(i<j)
        {
            int temp=arr[i];
            arr[i]=arr[j];
            arr[j]=temp;
        }
    }
    int temp=arr[l];
    arr[l]=arr[j];
    arr[j]=temp;
    return j;
}
void quickSort(int arr[],int l,int h)
{
    if(l<h)
    {
```

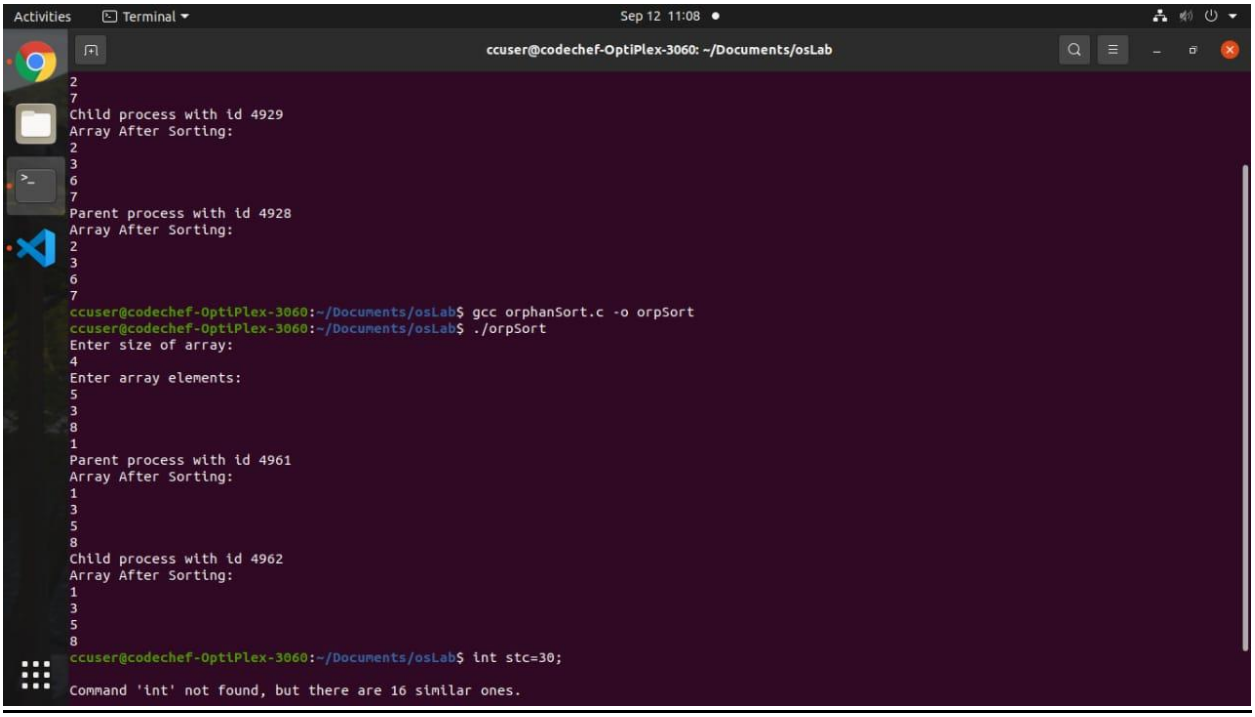
```
        int pivot=partition(arr,l,h);
        quickSort(arr,l,pivot);
        quickSort(arr,pivot+1,h);
    }
}
//Merge Sort
void merge(int arr[],int l, int mid, int h)
{
    int i,j,k;
    int b[h-l+1];
   for(i=l,j=mid+1,k=l; i<=mid && j<=h; k++ )
   {
        if(arr[i]<arr[j])
        {
            b[k]=arr[i++];
        }
        else
        {
            b[k]=arr[j++];
        }

    }
        while (j<=h)
        {
            b[k++]=arr[j++];
        }
        while(i<=mid)
        {
            b[k++]=arr[i++];
```

```
    }
  for(k=l; k<=h; k++)
  {
    arr[k]=b[k];
  }
}
void mergeSort(int arr[],int l, int h)
{
  if(l<h)
  {
  int mid=(l+h)/2;
  mergeSort(arr,l,mid);
  mergeSort(arr,mid+1,h);
  merge(arr,l,mid,h);
  }

}
int main()
{
  int n;
  printf("Enter size of array:\n");
  scanf("%d",&n);
  int arr[n];
  printf("Enter array elements:\n");
  for(int i=0; i<n; i++)
  {
    scanf("%d",&arr[i]);
  }
```

```
    //Zombie state
    int pid;
    pid=fork();
    if(pid==0)
    {
        int stc=30;
        wait(&stc);
        printf("Child process with id %d\n",getpid());
        quickSort(arr,0,n);
        printf("Array After Sorting:\n");
        for(int i=1; i<=n; i++)
        {
            printf("%d\n",arr[i]);
        }
    }
    else
    {
        int stc=30;
        wait(&stc);
        printf("Parent process with id %d\n",getpid());
        mergeSort(arr,0,n);
        printf("Array After Sorting:\n");
        for(int i=1; i<=n; i++)
        {
            printf("%d\n",arr[i]);
        }
    }
    return 0;
}
```

Output :


Activities
Terminal
Sep 12 11:07
ccuser@codechef-OptiPlex-3060: ~/Documents/osLab
ccuser@codechef-OptiPlex-3060:~/Documents/osLab$ gcc ZomSort.c -o sort1
ccuser@codechef-OptiPlex-3060:~/Documents/osLab$ ./sort1
Enter size of array:
4
Enter array elements:
3
6
2
7
Child process with id 4929
Array After Sorting:
2
3
6
7
Parent process with id 4928
Array After Sorting:
2
3
6
7
ccuser@codechef-OptiPlex-3060:~/Documents/osLab$ gcc orphanSort.c -o orpSort
ccuser@codechef-OptiPlex-3060:~/Documents/osLab$ ./orpSort
Enter size of array:
4
Enter array elements:
5
3
8
1
Parent process with id 4961
Array After Sorting:
1
3
5
8
Child process with id 4962
Array After Sorting:


Activities
Terminal
Sep 12 11:08
ccuser@codechef-OptiPlex-3060: ~/Documents/osLab
2
7
Child process with id 4929
Array After Sorting:
2
3
6
7
Parent process with id 4928
Array After Sorting:
2
3
6
7
ccuser@codechef-OptiPlex-3060:~/Documents/osLab$ gcc orphanSort.c -o orpSort
ccuser@codechef-OptiPlex-3060:~/Documents/osLab$ ./orpSort
Enter size of array:
4
Enter array elements:
5
3
8
1
Parent process with id 4961
Array After Sorting:
1
3
5
8
Child process with id 4962
Array After Sorting:
1
3
5
8
ccuser@codechef-OptiPlex-3060:~/Documents/osLab$ int stc=30;
Command 'int' not found, but there are 16 similar ones.

Assignment – 3

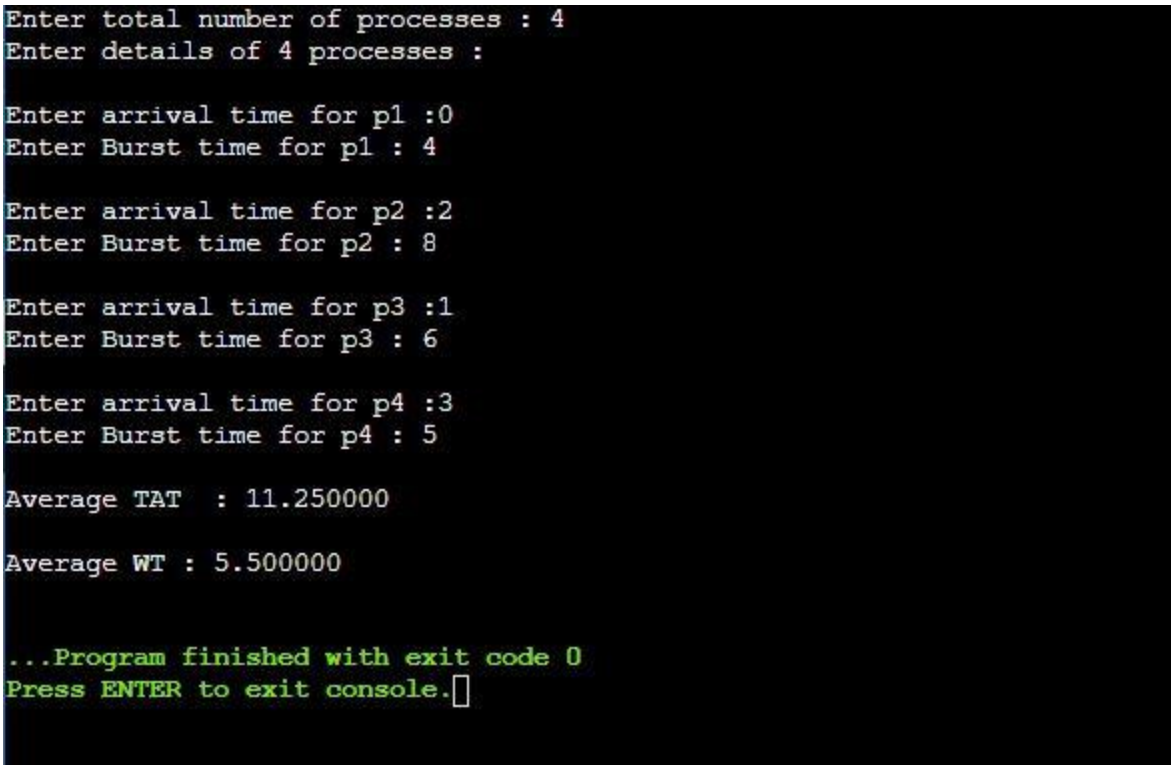
Shortest Job First (Preemptive) :

- Code :

```
#include<stdio.h>
int arrival_t[100],burst_t[100],ct[100],temp[100];
int main()
{
  int i,smallest,count=0,j,n;
  double avg_wt=0,avg_tat=0,end;

  printf("Enter total number of processes : ");
  scanf("%d",&n);

  printf("Enter details of %d processes :\n ",n);
  for(int i=0;i<n;i++)
  {
    printf("\nEnter arrival time for p%d :",i+1);
    scanf("%d",&arrival_t[i]);
    printf("Enter Burst time for p%d : ",i+1);
    scanf("%d",&burst_t[i]);
    temp[i]=burst_t[i];
  }
  burst_t[99]=1000;
  for(int i=0;count!=n;i++)
  {
    smallest=99;
    for(int j=0;j<n;j++)
    {
```

```
        if(arrival_t[j]<=i && burst_t[j]<burst_t[smallest] && burst_t[j]>0)
        {
           smallest=j;
        }
      }
      burst_t[smallest]--;
      if(burst_t[smallest] ==0)
      {
         count++;
         ct[smallest]=i+1;
      }
    }
    for(int i=0;i<n;i++)
    {
      int TAT=ct[i]-arrival_t[i];
      avg_tat+=TAT;
      avg_wt+=TAT-temp[i];

    }
    printf("\nAverage TAT  : %lf\n",avg_tat/n);
    printf("\nAverage WT : %lf\n",avg_wt/n);
    return 0;
}
```

Output :



- **Round Robin Algorithm :**

Code :

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int i, NOP, sum=0,count=0, y, quant, wt=0, tat=0, at[10], bt[10], temp[10];
    float avg_wt, avg_tat;
    printf("Total number of process in the system: ");
    scanf("%d", &NOP);
    y = NOP;
   for(i=0; i<NOP; i++)
  {
printf("\nEnter the Arrival and Burst time of the Process[%d]\n", i+1);
```

```
printf("Arrival time is: ");
scanf("%d", &at[i]);
printf("Burst time is: ");
scanf("%d", &bt[i]);
temp[i] = bt[i];
}
 printf("\nEnter the Time Quantum for the process: ");
scanf("%d", &quant);
printf("\nProcess No \t\t Burst Time \t\t\t TAT \t\t\t Waiting Time ");
for(sum=0, i = 0; y!=0; )
{
    if(temp[i] <= quant && temp[i] > 0)
    {
         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--;
        printf("\nProcess No[%d] \t\t %d\t\t\t\t %d\t\t\t %d", i+1, bt[i], sum-at[i], 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;
}
avg_wt = wt * 1.0/NOP;
avg_tat = tat * 1.0/NOP;
printf("\nAverage Turn Around Time: \t%f", avg_wt);
printf("\nAverage Waiting Time:  \t%f", avg_tat);
}
```

Output :

```
Total number of process in the system: 4

Enter the Arrival and Burst time of the Process[1]
Arrival time is: 0
Burst time is: 5

Enter the Arrival and Burst time of the Process[2]
Arrival time is: 1
Burst time is: 4

Enter the Arrival and Burst time of the Process[3]
Arrival time is: 2
Burst time is: 2

Enter the Arrival and Burst time of the Process[4]
Arrival time is: 4
Burst time is: 1

Enter the Time Quantum for the process: 2

Process No              Burst Time                      TAT                     Waiting Time
Process No[3]           2                               4                       2
Process No[4]           1                               3                       2
Process No[2]           4                               10                      6
Process No[1]           5                               12                      7
Average Turn Around Time:       4.250000
Average Waiting Time:   7.250000

...Program finished with exit code 0
Press ENTER to exit console.
```

Assignment - 4

- **Reader- Writer Problem :**

Code:

```
//READER WRITER PROBLEM
#include<stdio.h>
#include<stdlib.h>
#include<pthread.h>
#include<semaphore.h>
#include<unistd.h>
void *writer_thr(int temp);
void *reader_thr(int temp);
sem_t mutex;
sem_t wrt;
int readcount=0,nwt,nrd;
void main()
{
 long int i;
 sem_init(&mutex,0,1);
 sem_init(&wrt,0,1);
 pthread_t reader[100],writer[100];
 printf("\nEnter number of readers:");
 scanf("%d",&nrd);
 printf("\nEnter number of writers:");
 scanf("%d",&nwt);
for(i=1;i<=nwt;i++)
 {
   pthread_create(&writer[i],NULL,(void *)writer_thr,(int *)i);
   pthread_join(writer[i],NULL);
```

```
  }
 for(i=1;i<=nrd;i++)
  {
    pthread_create(&reader[i],NULL,(void *)reader_thr,(int *)i);
  }
for(i=1;i<=nrd;i++)
  {
    pthread_join(reader[i],NULL);
  }
  sem_destroy(&wrt);
  sem_destroy(&mutex);
}
void *reader_thr(int temp)
{
  printf("\nReader %d is trying to enter database for reading.",temp);
  sem_wait(&mutex);
  readcount++;
  if(readcount==1)
  sem_wait(&wrt);
  sem_post(&mutex);
  printf("\nReader %d is now reading in database.",temp);
  sem_wait(&mutex);
  readcount--;
  if(readcount==0)
  sem_post(&wrt);
  sem_post(&mutex);
  printf("\nReader %d has left the database.\n",temp);
  sleep(3);
}
```

```
void *writer_thr(int temp)
{
 printf("\nWriter %d is trying to enter database for modifying data",temp);
 sem_wait(&wrt);
 printf("\nWriter %d is writing in database.",temp);
 sleep(3);
 printf("\nWriter %d is leaving the database.\n",temp);
 sem_post(&wrt);
}
```

Output :

```
Enter number of readers:2

Enter number of writers:2

Writer 1 is trying to enter database for modifying data
Writer 1 is writing in database.
Writer 1 is leaving the database.

Writer 2 is trying to enter database for modifying data
Writer 2 is writing in database.
Writer 2 is leaving the database.

Reader 1 is trying to enter database for reading.
Reader 1 is now reading in database.
Reader 1 has left the database.

Reader 2 is trying to enter database for reading.
Reader 2 is now reading in database.
Reader 2 has left the database.


...Program finished with exit code 0
Press ENTER to exit console.
```

- **Producer – Consumer Problem**

//Producer – Consumer Problem

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

```
#include<pthread.h>
#include<stdlib.h>
#include<semaphore.h>
#include<unistd.h>
#define buffer_size 10
sem_t full,empty;
int buffer[buffer_size];
pthread_mutex_t mutex;
void *producer(void *p);
void *consumer(void *p);
void insert_item(int);
int remove_item();
int counter;

void initialize()
{
 pthread_mutex_init(&mutex,NULL);
 sem_init(&full,1,0);
 sem_init(&empty,1,buffer_size);
 counter=0;
}

int main()
{
 int n1,n2,i;
 printf("Enter no. of producers you want to create:");
 scanf("%d",&n1);
 printf("Enter no. of consumers you want to create:");
 scanf("%d",&n2);
```

```
 initialize();
 pthread_t tid[n1],tid1[n2];
 for(i=0;i<n1;i++)
    pthread_create(&tid[i],NULL,producer,NULL);
 for(i=0;i<n2;i++)
     pthread_create(&tid1[i],NULL,consumer,NULL);
 sleep(50);
 exit(0);
}
void *producer(void *p)
{
 int item,waittime;
 waittime=rand()%5;
 sleep(waittime);
 item =rand()%10;
 sem_wait(&empty);
 pthread_mutex_lock(&mutex);
 printf("\n Producer produced %d item",item);
 insert_item(item);
 pthread_mutex_unlock(&mutex);
 sem_post(&full);
}
void *consumer(void *p)
{
 int item,waittime;
 waittime=rand()%10;
 sleep(waittime);
 sem_wait(&full);
 pthread_mutex_lock(&mutex);
```

```
  item=remove_item();
  printf("\n Consumer consumed %d item",item);
  pthread_mutex_unlock(&mutex);
  sem_post(&empty);
}
void insert_item(int item)
{
  buffer[counter++]=item;
}
int remove_item()
{
  return(buffer[--counter]);
}
```

- Output: