assignment-10
input:

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

struct stuff
{
    char eng_stuff[20];
    int price;
    int  id;
    int quantity;

    struct stuff *link;
};

struct stuff *create(struct stuff *start);
void display (struct stuff *start);
struct stuff *insert(struct stuff *start);
struct stuff *del(struct stuff *start,char item[]);
struct stuff *sort (struct stuff *start);
void *search(struct stuff *start,char item[]);
struct stuff *modify(struct stuff *start,char item[]);
struct stuff *front=NULL;
struct stuff *rear=NULL;
void create_queue();
void del_queue();
void display_queue(struct stuff *front);
void graph(struct stuff *start,char [20]);
int isIDunique(struct stuff*start, int id);
struct stuff *p,*q,*temp;
char password[30]="krushna";
char pass[30];
char name[50];
int i,n;
char ch;
void login()
{
    printf("\nEnter your user name:");
    scanf("%s",name);
    printf("\nEnter the password:");
    //scanf("%s",pass);
    i=0;
    while(ch!=13)
    {
     ch=getch();
     pass[i]=ch;
     i++;
     printf("*");

    }
    pass[i-1]='\0';
```

```
        if(stricmp(password,pass)==0)
        {
            printf("system is open");

         }
        else
        {
          printf("\nYour system has been locked");
          exit (0);
        }
}
void main()
{
    login();
    printf("\n\t\t\t\t******WELCOME TO NOBEL_OF_NETWORK
ENG_STUFF******\n");
    char item[20];
    int ch;
    struct stuff *start=NULL;
    while(1)
    {
    printf("\nEnter choice-\n 1) to create\n 2) to create using
queue\n 3) to display\n 4) to display using queue\n 5) to insert\n 6)
to delete\n");
    printf(" 7) to delete using queue\n 8) to sort by price\n 9) to
search\n 10) to modify:\n 11) To check in Profit or in Loss\t:");
    scanf("%d",&ch);
    switch (ch)
    {
    case 1:
        start=create(start);
        break;
    case 2:
        create_queue();
        break;
    case 3:
        display(start);
        break;
    case 4:
        display_queue(front);
        break;
    case 5:
        start=insert(start);
        break;
    case 6:
        printf("Enter stuff you want to delete:");
        scanf("%s",item);
        start=del(start,item);
        break;
    case 7:
```

```
        del_queue();
        break;
    case 8:
        sort(start);
        break;
    case 9:
        printf("Enter stuff you want to search:");
        scanf("%s",item);
        search(start,item);
        break;
    case 10:
        printf("Enter stuff you want to modify:");
        scanf("%s",item);
        start=modify(start,item);
        break;
    case 11:
        printf("\n\t\t\t***|||*** represent PROFIT");
        printf("\n\n\t\t\t***||*** represent LOSS");
        printf("\nEnter item for it's profit / loss graph:\t");
        scanf("%s",item);
        graph(start,item);
        break;
    default:
        printf("error!");

    }
}
}
struct stuff *create(struct stuff* start)
{
    struct stuff *p,*temp;
    printf("Enter no of stuff:");
    scanf("%d",&n);

    temp=start;
    for(i=0;i<n;i++)
     { temp=(struct stuff*)malloc(sizeof(struct stuff)*1);
       printf("\nEnter Name of eng_stuff %d:",i+1);
       scanf("%s",(temp->eng_stuff));
       int error = 0;
       do {
            if(error)
        printf("\t\tID IS ALREADY PRESENT!n\n");
        printf("id of stuff:");
        scanf("%d",&(temp->id));
       error = 1;
       } while(!isIDunique(start, temp->id));
       printf("Quantity :");
       scanf("%d",&(temp->quantity));
       printf("Price of One eng_stuff:");
       scanf("%d",&(temp->price));
```

```
    if(start==NULL)
    {
     start=temp;
     temp->link=NULL;
    }
  else
  {
        p=start;
        while(p->link!=NULL)
        p=p->link;
        p->link=temp;
        temp->link=NULL;
  }
}
return start;
}

void display (struct stuff *start)
{ struct stuff *p;
  int sum=0;
  int j=0;

   if(start==NULL)
    {
        printf("\n*Queue Underflow*\n");
    }
    else
    {
        p=start;
        printf("\n\nS.No.\t\tstuff\t\t\tid\t\tQuantity\tPrice\t\tTotal
Price\n\n");
        while(p!=NULL)
        {
              printf("%d\t\t%s\t\t%d\t\t%d\t\t%d\t\t%d\n",++j,p-
>eng_stuff,p->id,p->quantity,p->price,(p->price*p->quantity));
              p=p->link;
        }
    }
       printf("\n");

}

struct stuff *insert(struct stuff *start)
{ int pos;

  struct stuff *p,*temp;
  temp=(struct stuff*)malloc(sizeof(struct stuff)*1);
  printf("Enter location you want to insert:");
  scanf("%d",&pos);

    if(pos==1)
  {
```

```
      printf("\nEnter Name of eng_stuff %d:",i+1);
      scanf("%s",(temp->eng_stuff));
      int error = 0;
      do{
           if(error)
               printf("Id already present!\n");

      printf("id:");
      scanf("%d",&(temp->id));
      error=1;
      }while(!isIDunique(start, temp->id));
      printf("Quantity :");
      scanf("%d",&(temp->quantity));
      printf("Price of One eng_stuff:");
      scanf("%d",&(temp->price));
      temp->link=start;
      start=temp;

    return start;
  }
     p=start;
     for(i=1;i<pos-1 && p!=NULL;i++)
        p=p->link;
     if(p==NULL)
        printf("\n*less no of stuff*\n\n");

    else
    {
       printf("\nEnter Name of eng_stuff %d:",i+1);
       scanf("%s",(temp->eng_stuff));
       int error=0;
       do{
            if(error)
                printf("\t\t\nID already present!\n");
        printf("id:");
        scanf("%d",&(temp->id));
       error = 1;
       }while(!isIDunique(start, temp->id));
       printf("Quantity :");
       scanf("%d",&(temp->quantity));
       printf("Price of One eng_stuff:");
       scanf("%d",&(temp->price));
       temp->link=p->link;
       p->link=temp;
    }
   return start;
}

struct stuff *del(struct stuff *start,char item[])
{

  struct stuff *temp,*p;
```

```
  if(strcmp(start->eng_stuff,item)==0)
  {
      temp=start;
      start=temp->link;
      free(temp);
      return start;
  }
      p=start;
      while(p->link!=NULL)
      { if(strcmp(p->link->eng_stuff,item)==0)
          {
            temp=p->link;
            p->link=temp->link;
            free(temp);
            return start;
      }
     p=p->link;
      }
     }

struct stuff *sort (struct stuff *start)
{   struct stuff *p,*q,*temp;
    temp=(struct stuff*)malloc(sizeof(struct stuff)*1);

   for(p=start;p->link!=NULL;p=p->link)
    {
        for(q=p->link;q!=NULL;q=q->link)
        {

        if(p->price > q->price)
          {
             temp->price=p->price;
             p->price=q->price;
             q->price=temp->price;
             strcpy(temp->eng_stuff,p->eng_stuff);
             strcpy(p->eng_stuff,q->eng_stuff);
             strcpy(q->eng_stuff,temp->eng_stuff);
             temp->quantity=p->quantity;
             p->quantity=q->quantity;
             q->quantity=temp->quantity;
             temp->id=p->id;
             p->id=q->id;
             q->id=temp->id;

          }
       }
    }
}

void *search(struct stuff *start,char item[])
{
```

```
  struct stuff *p;
  int flag=0;
  int j=0;

      if(strcmp(start->eng_stuff,item)==0)
         {
          printf("\n*ITEM FOUND*\n");
          printf("\n\nS.No.\t\tstuff\t\tid\t\tQuantity\tPrice\t\tTotal
Price\n\n");
          printf("%d\t\t%s\t\t%d\t\t%d\t\t%d\t\t%d\n",++j,start-
>eng_stuff,start->id,start->quantity,start->price,(start->price*p-
>quantity));
           flag=1;
         }


         p=start;
         while(p->link!=NULL)
           {
          if(strcmp(p->link->eng_stuff,item)==0)
          {
           printf("\n*ITEM FOUND*\n");
printf("\n\nS.No.\t\tstuff\t\tid\t\tQuantity\tPrice\t\tTotal
Price\n\n");
           printf("%d\t\t%s\t\t%d\t\t%d\t\t%d\t\t%d\n",++j,p->link-
>eng_stuff,p->link->id,p->link->quantity,p->link->price,p->link-
>price*p->link->quantity);

           }
           p=p->link;
           flag=1;
        }

        if(flag==0)
        printf("\n\nnot found\n\n");

}

struct stuff *modify(struct stuff *start,char item[])
{
    struct stuff *p;
    int choice;
    char ans1,ans2;
    int flag=0,chw=0;
    int j=0;

      if(strcmp(start->eng_stuff,item)==0)
     {
       printf("\nITEM FOUND\n");
       printf("\n\nSerial no\tstuff\t\tid\t\tQuantity\tPrice\t\tTotal
Price\n\n");
```

```
       printf("%d\t\t%s\t\t%d\t\t%d\t\t%d\t\t%d\n",++j,start-
>eng_stuff,start->id,start->quantity,start->price,start->price*start-
>quantity);

    do
     {
       printf("enter:\n 1) to modify eng_stuff name\n 2) to modify
eng_stuff price\n 3) to modify eng_stuff id\n 4) to modify eng_stuff
quantity:");
       scanf("%d",&choice);
       switch (choice)
       {
       case 1:
        printf("enter new eng_stuff name:");
        scanf("%s",(start->eng_stuff));
        break;
       case 2:
        printf("enter new eng_stuff price:");
        scanf("%d",&(start->price));
        break;
       case 3:
     do {
            if(error)
       printf("\t\tID IS ALREADY PRESENT!n\n");
       printf("enter new eng_stuff id:");
       scanf("%d",&(temp->id));
       error = 1;
       } while(!isIDunique(start, temp->id))

       break;
       case 4:
        printf("enter new eng_stuff quantity:");
        scanf("%d",&(start->quantity));
        break;
       }
       printf("press 1 to continue modifying:");
       scanf("%d",&chw);
     }
     while(chw==1);
            flag++;
            return start;
      }
       p=start;
       while(p->link!=NULL)
         {
            if(strcmp(p->link->eng_stuff,item)==0)
          {
           printf("\n*ITEM FOUND\n");
           printf("\n\nSerial
no\tstuff\t\tid\t\tQuantity\tPrice\t\tTotal Price\n\n");
```

```
           printf("%d\t\t%s\t\t%d\t\t%d\t\t%d\t\t%d\n",++j,p->link-
>eng_stuff,p->link->id,p->link->quantity,p->link->price,p->link-
>price*p->link->quantity);

            do
            {
             printf("enter:\n 1) to modify eng_stuff name\n 2) to
modify eng_stuff price\n  3) to modify eng_stuff id\n 4) to modify
eng_stuff quantity:");
             scanf("%d",&choice);

          switch (choice)
          {
           case 1:
             printf("enter new eng_stuff name:");
             scanf("%s",(p->link->eng_stuff));
           break;
           case 2:
             printf("enter new eng_stuff price:");
             scanf("%d",&(p->link->price));
           break;
           case 3:
             do {
            if(error)
        printf("\t\tID IS ALREADY PRESENT!n\n");
        printf("enter new eng_stuff id:");
        scanf("%d",&(temp->id));
       error = 1;
       } while(!isIDunique(start, temp->id))

           break;
           case 4:
             printf("enter new eng_stuff quantity:");
             scanf("%d",&(p->link->quantity));
           break;

           }printf("press 1 to continue modifying:");
            scanf("%d",&chw);
            }
            while(chw==1);
           }
               p=p->link;
               flag++;
               return start;
        }

        if(flag==0)
        printf("\n\nnot found\n\n");
}


void create_queue()
```

```
{
     struct stuff *temp,*p;
     int n;
     printf("Enter Number of types of orders:");
     scanf("%d",&n);
     for(i=0;i<n;i++)
     {
      if (rear==NULL)
    {
        rear=(struct stuff*)malloc(sizeof(struct stuff)*100);
        printf("\nEnter Name of eng_stuff %d:",i+1);
        scanf("%s",(rear->eng_stuff));
        printf("id:");
        scanf("%d",&(rear->id));
        printf("Quantity :");
        scanf("%d",&(rear->quantity));
        printf("Price of One eng_stuff:");
        scanf("%d",&(rear->price));
        rear->link=rear;
        front=rear;
    }
    else
    {
        temp=(struct stuff*)malloc(sizeof(struct stuff));
        printf("\nEnter Name of eng_stuff %d:",i+1);
        scanf("%s",(temp->eng_stuff));
        printf("id:");
        scanf("%d",&(temp->id));
        printf("Quantity :");
        scanf("%d",&(temp->quantity));
        printf("Price of One eng_stuff:");
        scanf("%d",&(temp->price));
        rear->link=temp;
        temp->link=NULL;
        rear=temp;
    }
     }
}

void display_queue(struct stuff *front)
{
    struct stuff *p;
    int sum=0;
    int j=0;

    if(front==NULL)
    {
        printf("\n*Queue Underflow*\n");
    }
    else
    {
        p=front;
```

```
        printf("\n\nS.No.\t\tstuff\t\tid\t\tQuantity\tPrice\t\tTotal
Price\n\n");
        while(p!=NULL)
        {
              printf("%d\t\t%s\t\t%d\t\t%d\t\t%d\t\t%d\n",++j,p-
>eng_stuff,p->id,p->quantity,p->price,(p->price*p->quantity));
              p=p->link;
        }
    }
}

void del_queue()
{
    struct stuff *temp;
    if(front==NULL)
    {
        printf("\n*Queue Underflow*\n");
    }
    else
    {
    temp=front;
    front=front->link;
    free (temp);
    }
}

void graph(struct stuff *start,char item[20])
{int tempqty,flag;
    if(strcmp(start->eng_stuff,item)==0)
    {
      printf("Quantity of eng_stuff %s is %d :",start-
>eng_stuff,start->quantity);
      tempqty=start->quantity;
      if((tempqty)>=5)
      {
        printf("\n\n***In Profit !!!***");

        for(i=0; i<tempqty; i++)
        {
            printf("\n \t\t||| \n");
        }
        printf("\n \t\t%s",start->eng_stuff);
      }

     else
     {
        printf("\n\n***In Loss !!!***");
        for(i=0; i<tempqty; i++)
        {
            printf("\n \t\t|| \n");
        }
        printf("\n \t\t%s",start->eng_stuff);
```

```
      }
      flag++;
      }
      p=start;

      while(p->link!=NULL)
      {
         if(strcmp(p->link->eng_stuff,item)==0)
         {
           printf("Quantity of eng_stuff %s is %d :",p->link-
>eng_stuff,p->link->quantity);
           tempqty=p->link->quantity;
           if((tempqty)>=5)
           {
             printf("\n\n***In Profit !!!***");
             for(i=0; i<tempqty; i++)
             {
               printf("\n \t\t||| \n");
             }
             printf("\n \t\t%s",p->link->eng_stuff);
           }
           else
           {
              printf("\n\n***In Loss !!!***");
              for(i=0; i<tempqty; i++)
              {
                printf("\n \t\t|| \n");
              }
              printf("\n \t\t%s",p->link->eng_stuff);
           }
           flag++;
         }  p=p->link;

         }
         if(flag==0)
         printf("\n\n**No item found**\n\n");
}
int isIDunique(struct stuff* head, int id)
{
    struct stuff* temp = head;
    while(temp!=NULL)
    {
        if(temp->id == id)
            return 0;
        temp = temp->link;
    }
    return 1;
}
```

Output-1

Enter your user name:admin

Enter the password:*********

Your system has been locked

Process returned 0 (0x0)   execution time : 16.504 s

Press any key to continue.

Output-2

Enter your user name:admin

Enter the password:********system is open

******WELCOME TO NOBEL_OF_NETWORK ENG_STUFF******

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss   :1

Enter no of stuff:2

Enter Name of eng_stuff 1:computer

id of stuff:1234

Quantity :5

Price of One eng_stuff:10004

Enter Name of eng_stuff 2:keyboard

id of stuff:1234

ID IS ALREADY PRESENT!n

id of stuff:1236

Quantity :51

Price of One eng_stuff:1002

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :3

S.No.	stuff	id	Quantity	Price	Total Price
1	computer	1234	5	10004	50020
2	keyboard	1236	51	1002	51102

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :5

Enter location you want to insert:2

Enter Name of eng_stuff 2:motherboard

id:1234

ID already present!

id:1235

Quantity :4

Price of One eng_stuff:100

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :3

S.No.	stuff	id	Quantity	Price	Total Price
1	computer	1234	5	10004	50020
2	motherboard	1235	4	100	400
3	keyboard	1236	51	1002	51102

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :8

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :3

S.No.	stuff	id	Quantity	Price	Total Price
1	motherboard	1235	4	100	400
2	keyboard	1236	51	1002	51102
3	computer	1234	5	10004	50020

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :6

Enter stuff you want to delete:motherboard

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :3

S.No.	stuff	id	Quantity	Price	Total Price
1	keyboard	1236	51	1002	51102
2	computer	1234	5	10004	50020

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :10

Enter stuff you want to modify:computer

*ITEM FOUND

Serial no	stuff	id	Quantity	Price	Total Price
1	computer	1234	5	10004	50020

enter:

1) to modify eng_stuff name

2) to modify eng_stuff price

3) to modify eng_stuff id

4) to modify eng_stuff quantity:2

enter new eng_stuff price:10

press 1 to continue modifying:1

enter:

1) to modify eng_stuff name

2) to modify eng_stuff price

3) to modify eng_stuff id

4) to modify eng_stuff quantity:4

enter new eng_stuff quantity:8

press 1 to continue modifying:0

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :3

S.No. stuff id Quantity Price Total Price

1	keyboard	1236	51	1002	51102
2	computer	1234	8	10	80

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :11

||| represent PROFIT

|| represent LOSS

Enter item for it's profit / loss graph: computer

Quantity of eng_stuff computer is 8 :

In Profit !!!

|||

|||

|||

|||

|||

|||

|||

|||

computer

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :10

Enter stuff you want to modify:keyboard

ITEM FOUND

Serial no	stuff	id	Quantity	Price	Total Price
1	keyboard	1236	51	1002	51102

enter:

1) to modify eng_stuff name

2) to modify eng_stuff price

3) to modify eng_stuff id

4) to modify eng_stuff quantity:4

enter new eng_stuff quantity:4

press 1 to continue modifying:0

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :3

S.No.	stuff	id	Quantity	Price	Total Price
1	keyboard	1236	4	1002	4008
2	computer	1234	8	100005	800040

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :11

||| represent PROFIT

|| represent LOSS

Enter item for it's profit / loss graph: keyboard

Quantity of eng_stuff keyboard is 4 :

In Loss !!!

||

||

||

||

keyboard

Enter choice-

1) to create

2) to create using queue

3) to display

4) to display using queue

5) to insert

6) to delete

7) to delete using queue

8) to sort by price

9) to search

10) to modify:

11) To check in Profit or in Loss :