```c
#include<stdio.h>
#include<malloc.h>
#include<string.h>

struct node
{
    int pid,quantity;
    float price;
    char name[50];
    int out;
    int count;
    struct node *next;
};

struct pop
{
    int pid,quantity;
    float price;
    char name[50];
    int out;
    int count;
};

struct pop obj;
struct node *head=NULL,*p,*last=NULL,*q=NULL;

int n,m,i;
int pid,quantity;
float price;
char name[50];
void create();
void display();
void sort();
void search();
struct node* insert();
void reverse();
void write();
void read();
void delete1();
void modify();
//void enter();
void graph();
int search_pid(int);
void queue();
int count();
int search_name();
int nameser();


int main()
{
    system("color 9");
    int ch,ch1,ch2,m=0,m1=0,ph_no,emp,pid1,loc,x,choice;
    char str1[20],str2[20],str3[]="abcd",ch3,ch5,ch4,c_name[50],address[50];
    char password[100],c=' ',str[20];
    int i=0;
    printf("\n\n---------------------------------------------\n");
    printf("     WELCOME TO My Restaurant                     ");
    printf("\n---------------------------------------------\n");
    printf("\n1.Admin \n2.User \nEnter Your Choice:");
    scanf("%d",&ch);
    switch(ch)
    {
    case 1:
        do
        {
            //printf("\nNote:Use only lower case letters & digits\n");
        printf("\n Enter Name:");
        scanf("%s",str1);
        printf("\n Enter Password:");
        while(i<9)
        {
            str[i]=getch();
            c=str[i];
            if(c==13)
            break;
            else printf("*");
            i++;
        }

        str[i]='\0';
        i=0;
        strlwr(str);
        if((strcmp(str,str3))==0)
```

```
          {

      do
      {
     printf("\n\n----------------------------------------------\n");
     printf("                   MENU                         ");
     printf("\n----------------------------------------------\n");
      printf("\n1 Create\n2.Display\n3.Insert
\n4.Sort\n5.Reverse\n6.Search\n7.Delete\n8.Graph\n9.Modify\n.10 Exit");
      printf("\nEnter Your Choice:");
      scanf("%d",&ch);
      switch(ch)
      {
      case 1:system("CLS");
      create();
      break;
      case 2:system("CLS");
      display();
      break;
      case 3:system("CLS");
      insert();
      break;
      //n=n+1;*/
      case 4:system("CLS");
      sort(head);
      break;
      case 5:system("CLS");
      reverse();
      break;
      /*case 6:system("CLS");
      write();
      break;
      case 7:system("CLS");
      read();
      break;*/
      case 7:system("CLS");
     delete1();
     break;
      case 6:system("CLS");
      search();
      break;
      case 9:system("CLS");
      modify();
      break;

      case 8:system("CLS");
      graph();
      break;
      }
      }while(ch!=10);
  return 0;
  }
  else
          {
              printf("\nPASSWORD IS WRONG !!!");
          }
          printf("\nDo You Want To Continue Y OR N:");
          scanf("%s",&ch3);
          strlwr(ch3);
      }while(ch3=='y');
     return 0;
     break;
     case 2:
            //display();


            do
      {
     printf("\n\n----------------------------------------------\n");
     printf("                 Restaurant MENU                     ");
     printf("\n----------------------------------------------\n");
      printf("\n\n1.Display\n2.place your order\n3.Cancel order\n4.Exit");
      printf("\nEnter Your Choice:");
      scanf("%d",&choice);
      switch(choice)
      {

      case 1:system("CLS");
      display();
      break;
      case 2://
          system("CLS");
      order();
      break;
```

```c
      case 3:system("CLS");
      cancelorder();
      break;

      }
            //printf("\nDo You Want To See Again Data Y OR N:");
            //scanf("%s",&ch4);
        }while(choice!=4);
        return 0;
        break;
      }
}


    /* void enter()
      {
      printf("\nEnter The DishID:");
      scanf("\t%d",&(pid));
      printf("Enter The Name:");
      scanf("\t%s",name);
      printf("Enter The Quantity:");
      scanf("\t%d",&(quantity));
      printf("Enter The Price:");
      scanf("\t%f",&(price));

           q=head;
      for(;q!=NULL;)
      {
          if(pid==q->pid&&strcmp(name,q->name))
          {
              printf("Record with same name or pid already exist.Please enter again");
              enter();
          }

          q=q->next;
      }
      }*/

void create()
{
     int f=0,f2=0,x=0,y=0,cnt,not_entered=1;
     read();
     printf("\n\n---------------------------------------------\n");
     printf("                  Creating Data                      ");
     printf("\n--------------------------------------------\n");
     printf("\nEnter Number of Items:");
     scanf("%d",&n);
    // enter();
     struct node *p;
     int i=0;
     p=last;
     if(last==NULL)
     {
     i=1;
     last=(struct node*)malloc(sizeof(struct node));
     last->next=NULL;
     //enter();
     printf("Enter The DishID:");
     while(1)
     {
     scanf("\t%d",&(x));
     f=search_pid(x);
     if(f==0)
     {
         last->pid=x;
         break;
     }
     else
     {
         printf("DishID Already Exist...Enter The DishID Again:");
         f=0;
     }
     }
       printf("Enter The Name:");
     //while(not_entered)
     {

     scanf("\t%s",&(obj.name));
     /*f=nameser();
     if(f==0)
     {
         strcpy(last->name,obj.name);
         not_entered=1;
         break;
```

```
            //printf("DishName Already Exist...Enter The DishName Again:");
            //f=0;
        }
        else
        {
            printf("DishName Already Exist...Enter The DishName Again:");
            f=0;
            //strcpy(last->name,obj.name);
            //not_entered=1;
            //break;
        }*/
        }


        //printf("\nEnter The DishID:");
        //scanf("\t%d",&(last->pid));
        //printf("Enter The Name:");
        //scanf("\t%s",last->name);
        printf("Enter The Quantity:");
        scanf("\t%d",&(last->quantity));
        printf("Enter The Price:");
        scanf("\t%f",&(last->price));
        last->out = 0;              //patient is in queue
        cnt = count();
        last->count  = cnt+1;

        p=last;
        head=last;
        }//if
        p=last;
        for(;i<n;i++)
        {

        p->next=(struct node*)malloc(sizeof(struct node));
        p=p->next;
        last=p;
        printf("\nEnter The DishID:");
        while(1)
        {
        scanf("\t%d",&x);
        f2 = search_pid(x);
          if(f2 == 0)
           {
           p->pid = x;
           break;
          }
          else{
            printf("\nDishID Already Exists..Enter a New DishID:");
            f2 = 0;

           }
           //break;
           //fflush(stdin);
        }

          printf("\nEnter The Name:");
         while(not_entered)
        {

        scanf("\t%s",&(obj.name));
        f2=nameser();
        if(f2==0)
        {
            strcpy(p->name,obj.name);
            not_entered=1;
            break;
            //printf("DishName Already Exist...Enter The DishName Again:");
            //f2=0;
        }
        else
        {
            printf("DishName Already Exist...Enter The DishName Again:");
            f2=0;
            //strcpy(p->name,obj.name);
            //not_entered=1;
            //break;
        }
        }

        //printf("\nEnter The DishID:");
        //scanf("%d",&(p->pid));
        //printf("Enter The Name:");
        //scanf("%s",(p->name));
        printf("Enter The Quantity:");
```

```
    scanf("%d",&(p->quantity));
    printf("Enter The Price:");
    scanf("%f",&(p->price));
    p->out = 0;                // patient is in queue
    cnt = count();
    last->count  = cnt+1;
    //enter();
    p->next=NULL;
    }//i for
    write();

    }


void display()

{
    read();
    int i;
    printf("\n\n----------------------------------------------\n");
    printf("                 Displaying Data                     ");
    printf("\n----------------------------------------------\n");
    p=head;
    if(p==NULL)
    {
    printf("\nList Is Empty!!!");
    }
    else
    {
    printf("\n List Is:");
    printf("\n\tDishID\tDishName\tQuantity\tRate\n");
       for(i=0;p!=NULL;i++)
       {
           printf("\n");
           printf("\t%d\t",p->pid);
           printf("\t%s\t",p->name);
           printf("\t%d\t",p->quantity);
           printf("\t%.2f\t",p->price);
           p=p->next;
       }//for

    }//else

}

struct node* insert()
{
    int f=0,x=0,not_entered=1,cnt;
    read();
    printf("\n\n----------------------------------------------\n");
    printf("                 Inserting Data                     ");
    printf("\n----------------------------------------------\n");
    p=head;
    q=head;

    int i,loc,pid,quantity;
    float price;
    char name[50];
    printf("\nEnter The New Position:");
    scanf("%d",&loc);
    /*if(strcmp(pid1,pid))
    {
        printf("These DishID Already Exists!!!");

    }
    else
    {*/
    printf("Enter New DishID:");
    while(1){
      scanf("\t%d",&x);
      f = search_pid(x);
      if(f == 0){
       pid = x;
       break;
      }
      else{
        printf("\nDish ID already exists..Enter a new one ");
        f=0;
       }
       fflush(stdin);
    }

    printf("Enter The Name:");
    while(not_entered)
```

```
      {

      scanf("\t%s",(obj.name));
      f=nameser();
      if(f==0)
      {
          strcmp(last->name,obj.name)==0;
          not_entered=1;
          break;
          //printf("DishName Already Exist...Enter The DishName Again:");
          //f=0;
      }
      else
      {
          printf("DishName Already Exist...Enter The DishName Again:");
          f=0;
          //strcpy(last->name,obj.name);
          //not_entered=1;
          //break;
      }
      }

     /*/printf("Enter New DishID:");
      scanf("%d",&pid);
      printf("Enter New Name:");
      scanf("%s",name);*/
      printf("Enter New Quantity:");
      scanf("%d",&quantity);
      printf("Enter The New Price:");
      scanf("%f",&price);
      //enter();

      p=(struct node*)malloc(sizeof(struct node));
      p->pid=pid;
      strcpy(p->name,obj.name);
      p->quantity=quantity;
      p->price=price;
      p->out = 0;      //patient is in queue
      cnt = count();
      p->count  = cnt+1;
      p->next=NULL;
      if(loc==1)
      {
          p->next=head;
      head=p;
          write();
          return(p);
      }
      q=head;
      for(i=1;i<loc-1;i++)
      {

          if(q!=NULL)
          {
              q=q->next;
          }
      }
      p->next=q->next;
      q->next=p;
      n=n+1;
      write();
      return(head);
}

void modify()
{
      int f=0,x=0,not_entered=1,cnt;
      int i=1,ch,choice=1,loc=0;
      printf("\n\n---------------------------------------------\n");
      printf("                 Modifying Data                     ");
      printf("\n---------------------------------------------\n");
      read();
      printf("\nEnter The Location You Want To Modify:");
      scanf("%d",&loc);
      p=head;
      for(i=1;i<=loc;i++)
      {
      if(loc==i)
      {
      do{
      printf("\n0.Exit");
      printf("\t1.DishID:%d",p->pid);
      printf("\t2.Name:%s",p->name);
      printf("\t3.Quantity:%d",p->quantity);
      printf("\t4.Price:%.2f\n",p->price);
      printf("\nEnter Your Choice To Edit:");
      scanf("%d",&ch);
      switch(ch)
      {
     case 1:

      printf("\nEnter The New DishID:");
      while(1){
        scanf("%d",&x);
        f = search_pid(x);
        if(f == 0){
         p->pid = x;
         break;
        }
        else{
          printf("Dish ID already exists..Enter a new one ");
          f=0;
         }
      }
      //scanf("%d",&p->pid);
      //enter();
      break;
      case 2:
          printf("Enter The Name:");
     while(not_entered)
      {

      scanf("\t%s",(obj.name));
      f=nameser();
      if(f==0)
      {
          strcpy(last->name,obj.name);
          not_entered=1;
          break;
          //printf("DishName Already Exist...Enter The DishName Again:");
          //f=0;
      }
     else
      {
          printf("DishName Already Exist...Enter The DishName Again:");
          f=0;
          //strcpy(last->name,obj.name);
          //not_entered=1;
          //break;
      }
      }

      //enter();
      break;
      case 3:

      printf("Enter The New Quantity:");
      scanf("%d",&p->quantity);
      //enter();
      break;
      case 4:

      printf("Enter New Price:");
      scanf("%f",&p->price);
      //enter();
      break;
      }//switch
      }while(ch!=0);

      }//if
      if(p->next==NULL)
      {
      break;
      }//if
      p=p->next;

      }//for

      write();
  }

  void sort()
      {
      int k;
      printf("\n\n---------------------------------------------\n");
     printf("                    Sorting Data                    ");
     printf("\n---------------------------------------------\n");
      read();
```

```c
    int temp;
    struct node *i,*j;
    for(i=head;i!=NULL;i=i->next)
    {
    for(j=i->next;j!=NULL;j=j->next)
    {
    if((i->pid)>(j->pid))            //write integer variable only in place of data
    {
    obj.pid=i->pid;
    obj.quantity=i->quantity;
    obj.price=i->price;

    i->pid=j->pid;
    i->quantity=j->quantity;
    i->price=j->price;

    j->pid=obj.pid;
    j->quantity=obj.quantity;
    j->price=obj.price;
    for(k=0;k<sizeof(obj.name);k++)
    {
    obj.name[k]=i->name[k];
    i->name[k]=j->name[k];
    j->name[k]=obj.name[k];
    }
    }//if
    }//j for
    }//i for

    write();
    }//void sort


void reverse()
    {
    read();
    printf("\n\n---------------------------------------------\n");
    printf("                Reversing Data                    ");
    printf("\n---------------------------------------------\n");

    struct node *prenode,*currnode;
    if(head!=NULL)
    {
    prenode=head;
    currnode=head->next;
    prenode->next=NULL;
    }
    while(head!=NULL)
    {
    head=currnode->next;
    currnode->next=prenode;
    prenode=currnode;
    currnode=head;
    }
    head=prenode;

    write();

    }//void reverse

void delete1()
{
    read();
    printf("\n\n---------------------------------------------\n");
    printf("                Deleting Data                    ");
    printf("\n---------------------------------------------\n");
    int loc,i;
    printf("\nEnter The Location To Delete:");
    scanf("%d",&loc);
    printf("\nNumber %d is Being Deleted..Please Wait",loc);

    if(p!=NULL)
    {
    q=head;
    p=q->next;

    for(i=1;i<=loc;)
    {
    if(i==2)
    {
    q=head;
    p=q->next;
    }
    if(i==loc&&i==1)
    {
    head=head->next;
    printf("\nDishID:%d",q->pid);
    free(q);
    write();
    printf("\nfree q");
    break;
    }
    else
    {
    if(i==loc)
    {
    q->next=p->next;
    printf("\nq->pid:%d",p->pid);
    write();
    free(p);
    printf("\nfree p");
    }//if
    }//else
    printf("\nIncremented");
    p=p->next;
    q=q->next;
    i++;
    }//for
    p=head;
    q=head;
    }
    else
    {
    printf("\nSorry,The List Is Empty!!!");
    }
    }//void delete

void search()
    {
        FILE *ne=fopen("newfile.txt","r");
    int pid,found=0;
    printf("\n\n-----------------------------------------------\n");
   printf("                  Searching Data                    ");
   printf("\n---------------------------------------------\n");
    p=head;

    printf("\nEnter The Dish ID To Search:");
    scanf("%d",&pid);

    for(i=1;p!=NULL;i++)
    {
    if(p->pid==pid)
    {
    printf("\nData Found At %dth Location",i);
    printf("\nDishID:%d \tName:%s \tQuantity:%d
\tPrice:%f\n",p->pid,p->name,p->quantity,p->price);
    found=1;
    }
    p=p->next;
    }
    if(!found)
    {
    printf("\nNo Entry Found Corresponding To Your Data\n");
    }
    fclose(ne);


    }

void write()
    {

    FILE *ne=fopen("newfile.txt","w");
    int i;
    struct node *temp;
    temp=head;
    if(temp==NULL)
    {
        printf("\nList Is Empty!!!");
    }
    else
    {
    printf("\nList Is Being Saved!!!");
    for(i=0;temp!=NULL;i++)
    {
    fprintf(ne,"%d %s %d %f \n",temp->pid,temp->name,temp->quantity,temp->price);

    temp=temp->next;
    }//for
    printf("\nDone\n");

    }//else

    fclose(ne);
    }

void read()
    {
    int i,filempty=0;
    FILE *infile=fopen("newfile.txt","r");
    p=head;
    fseek(infile,0,SEEK_END);

    if(p==NULL)
    {
    p=(struct node*)malloc(sizeof(struct node));
    head=p;
    }

    int len=(int)ftell(infile);
    if(len<=0)
    {
    filempty=1;
    printf("\nFile Empty!!!");
    p=NULL;
    head=p;
    }

    if(filempty==0)
    {
    rewind(infile);
    while(fscanf(infile,"%d %s %d %f \n",&p->pid,p->name,&p->quantity,&p->price))
    {
    if(feof(infile))
    {
    break;
    }
    p->next=(struct node*)malloc(sizeof(struct node));
    p=p->next;
    last=p;
    p->next=NULL;
    }
    }
    }

void graph()
{
    int j;

    int value;
    float height=0;
    int length;
    read();
    printf("\n\n-----------------------------------------------\n");
   printf("                  Graph                    ");
   printf("\n-----------------------------------------------\n");
    read();
    p=head;
    for(i=1;p!=NULL;i++)
    {
    if(height<p->quantity)
    {
    height=p->quantity;
    }
    p=p->next;
    }
    length=40/i;
    p=head;

    for(i=(int)height+10;i>=0;i--)
    {
    printf("\n*");
    p=head;
    for(;p!=NULL;)
    {
    value=(int)p->quantity;
    if(i<=value)
    {
    printf("\t*");
    }
    else{printf("\t");}
    p=p->next;
```

```c
        }
        }
        p=head;
        printf("\n\n");
        printf("0");
        for(j=1;p!=NULL;j++)
        {
        printf("\t%d",j);
        p=p->next;
        }
        printf("\n\n");
        p=head;
        for(;p!=NULL;)
        {
        printf("%s(%.2d)\t",p->name,p->quantity);
        p=p->next;
        }

}

void queue()
{
        int i,max;
        printf("\n\n----------------------------------------------\n");
       printf("                    Queue                    ");
       printf("\n----------------------------------------------\n");
        read();
        struct node* x = head;
        max = count();
        if(x==NULL){
            printf("\n No Data!!!");
            return;
        }else
        {
            for(i=1;i<= max; i++)
                {
                 x = head;
                 while( x!= NULL)
                 {
                  if( (x->count == i) && (x->out ==0) )
                  {
                     printf("\n DishID : %d",x->pid);
                     printf("\n DishName : %s",x->name);
                     x->out = 1;
                     write();
                     return;
                  }
                  x= x->next;
                }
            }
            printf("\n No Data!!!");
            return ;
        }

}

int search_pid(int pid){
        struct node* x = head;
        int i=0;
        if(x==NULL)
        {
          return 0;
        }else{
            for(i=0;x!=last;i++)
            {
              if(x->pid == pid)
                return 1;
              x=x->next;


            }

        }
        return 0;
}

int nameser()
{

struct node*y=head;
//y=head;
       for(i=0;y!=last;i++)
       {
           if(!strcmp(y->name,obj.name))
        {
        return 1;

        }
        y=y->next;
    }

return 0;
}


int count(){
    int i=0,max;
    struct node* x =head;

    if(x==NULL)
    {
      return 0;
    }else{
        max = x->count;
        for(i=0;x!=last;i++)
        {
          if(x->count > max)
           max = x->count;
          x=x->next;

        }

    }
    return max;
}
void order()
    {
        char ch;
        int qut;
         int tbill;

        FILE *ne=fopen("newfile.txt","r");
    int pid,found=0;
    printf("\n\n------------------------------------------------\n");
    printf("                  Place Your Order                   ");
    printf("\n------------------------------------------------\n");
    p=head;


    printf("\nEnter The Dish ID To Order:");
    scanf("%d",&pid);

    for(i=1;p!=NULL;i++)
    {
    if(p->pid==pid)
    {
    //printf("\nData Found At %dth Location",i);
    printf("\nDishID:%d \tName:%s  \tPrice:%f\n",p->pid,p->name,p->price);
    found=1;
    printf("\nPlease Enter Quantity of Item");
    scanf("%d",&qut);
    tbill=p->price*qut;
    printf("\nYour Total Bill  is----> %d",tbill);

    }
    p=p->next;
    }


    if(found==1);
    printf("\nYour Order Has been Placed Sucessfully!!!!!!");
    if(!found)
    {
    printf("\nNo Entry Found Corresponding To Your Data\n");
    }
    fclose(ne);



    }
    void cancelorder()
    {
        char ch;
        FILE *ne=fopen("newfile.txt","r");
    int pid,found=0;
    printf("\n\n------------------------------------------------\n");
    printf("                  Canceling  Your Order                   ");
    printf("\n------------------------------------------------\n");
```

```c
	p=head;


	printf("\nEnter The Dish ID To Cancel Order:");
	scanf("%d",&pid);

	for(i=1;p!=NULL;i++)
	{
	if(p->pid==pid)
	{
	//printf("\nData Found At %dth Location",i);
	printf("\nDishID:%d \tName:%s  \tPrice:%f\n",p->pid,p->name,p->price);
	found=1;
	}
	p=p->next;
	}
	if(found==1);
	printf("\nYour Order Has been   Canceled Sucessfully!!!!!!");
	if(!found)
	{
	printf("\nNo Entry Found Corresponding To Your Data\n");
	}
	fclose(ne);



	}

```