****

**INF212**

**ALGORITHMS AND PROGRAMMING II**

**PROJECT-1**

**Sale Program**

**Deadline is April 1, 2021 at 23:00.**

**Projects that are not delivered on time are not accepted.**

**Upload the project to the Project 1 assignment section of the INF212 class.**

**The questions can be asked to course lecturer Dr. Tuba GÖZEL and teaching assistant Süleyman TUNCEL and Mehmet Türker TAKCI.**

**Complete your project, fill the report by using attached word file   
(INF212\_Project\_1\_Report\_Template.docx).**

**Upload the project codes and the report.**

**You can compress the project codes if number of the files is more than ten.**

**The file name of report should be as *P1\_ID\_FirstLetterofName\_.docx*   
(i.e. *P1\_141024001\_YKZ.docx* ).**

****

**INF 212 Algorithms and Programming II**

**2020-2021 Spring**

**Electronic Engineering**

**Proje ID:1 Project 1**

**Satış Programı**

|  |  |  |
| --- | --- | --- |
| **School ID** | **Name** | **Surname** |
| **1901022050** | **Merve** | **Tutar** |

|  |
| --- |
| PROJECT OBJECTIVE  Briefly describe the objectives of your project |
| Projenin amacı ürün bilgileri, müşteri bilgileri yardımı ile ürün satışı yapan bir satış otomasyon projesi oluşturmaktır. |
| PROBLEM  Briefly describe the problem you given to solve |
| Problemimiz ürün bilgileri ve müşteri bilgileriyle bağlı listeleri kullanarak ürün satışı yapabilmek ve bu verilerle analizler yapabilmektir. |
| ANALYSIS  Analyze of the problem |
| Problemimiz bağlı listeler kullanarak ürün bilgileri ve müşteri bilgileriyle ürün satışı yapıp bu ürünlerin analizini müşteriye ürün tipine, miktarına ve tutarına göre yapmaktır. Müşteri Bilgisi kısmında tüm müşterilerin, müşteri tipine göre müşterilerin veya tek bir müşterinin bilgileri listelenmelidir. Ürün Bilgisi kısmında tüm ürünlerin, ürün tipine göre ürünlerin veya tek bir ürünün bilgileri listelenmelidir.Son olarak müşteri ve ürün analizleri yapılmalıdır. |
| DATA REQUIREMENTS  Specify all data requirements of the problem |
| **Structured Data Type** |
| struct customer {  int ID;// müşteri numarası  char name[50];// müşteri adı  unsigned int type:1;// müşteri tipi ( bireysel müşteri ise 0, ticari müşteri ise 1)  double x\_coord;// müşterinin coğrafi bilgi sistemindeki x koordinatı  double y\_coord;// müşterinin coğrafi bilgi sistemindeki y koordinatı,  struct customer \*nextCusPtr;// bir sonraki müşteri bilgisinin bulunduğu bellek adresi  };  typedef struct customer Customer;  struct product {  int ID;// ürün numarası  char name[50];// ürün adı  int type; //ürün tipi  double price;// ürün fiyatı  struct product \*nextProPtr;// bir sonraki ürün bilgisinin bulunduğu bellek adresi  };  typedef struct product Product;  struct purchased {  int ID;//satılan parça numarası  int invoice\_ID;//fatura no  int customer\_ID;//müşteri no  int product\_ID;//ürün no  double cost;//maliyet  struct purchased \*nextPurchasedPtr;// bir sonraki satış yapılan ürün ile ilgili bilgilerin bulunduğu bellek adresi  };  typedef struct purchased Purchased; |
| **Problem Constants** |
| Problem sabiti kullanılmadı. |
| **Problem Inputs** |
| Product \*node;  Purchased \*pur;  Customer \*root;  unsigned int choice;//menüde seçim yapmak için kullanılır  unsigned int sec;//menülerde seçim yapmak için kullanılır  int id;//bulmak istediğimiz müşteri numarası için kullanılır  double sumcost;//toplam maliyet |
| **Problem Output** |
| Satış kısmında müşteri numarası,ürün numarası ve miktarı girildiğinde tutar hesaplanıp müşteriye ödemesi gereken ücreti yazdırır.müşteri bilgisi kısmında tüm müşterilerin, müşteri tipine göre müşterilerin veya tek bir müşterinin bilgileri listelenmelidir. Ürün Bilgisi kısmında tüm ürünlerin, ürün tipine göre ürünlerin veya tek bir ürünün bilgileri ekrana yazdırılır. Müşteri analizleri kısmında ise bir müşterinin satın aldığı ürünleri yazdırır. Bir müşterinin satın aldığı ürünlerin toplam tutarını gösterir. Tüm müşterinin satın aldığı ürünlerin toplam tutarı listelenir. Müşterilerin toplam kargo ücreti listelenir. Ürün analizleri kısmında ise bir ürünün toplam satın alınan tutarı listelenir. |
| DESIGN  ALGORITHM  Design algorithm of the problem |
| Menüden yapılmak istenen işlem seçilir  1. Satış  Müşteri bilgilerini al  Son fatura numarasını bul  Müşteri numarası isteyerek satış işlemine başla  Ürün no bul  Ürün miktarına göre tutarı hesapla  Kargo ücreti müşteri ile ürün satılan yerin uzaklığı ile orantılı olarak hesapla  2. Müşteri Bilgisi  Tüm müşterileri listele  Müşteri tipine göre müşterileri listele  Müşteri numarasına göre bir müşteriyi listele  3. Ürün Bilgisi  Tüm ürünleri listele  ürün tipine göre ürünleri listele  ürün numarasına göre bir ürünü listele  4. Müşteri Analizleri  Bir müşterinin satın aldığı ürünleri listele  Bir müşterinin satın aldığı toplam tutarı listele  Tüm müşterinin satın aldığı ürünlerin toplam tutarını listele  Müşterilerin kargo ücretini listele  5. Ürün Analizleri  Bir ürünün toplam satış tutarı |
| HEADER F;ILE(S) |
| #ifndef \_MY\_H\_  #define \_MY\_H\_  struct customer {  int ID;  char name[50];  unsigned int type:1;  double x\_coord;  double y\_coord;  struct customer \*nextCusPtr;  };  typedef struct customer Customer;  struct product {  int ID;  char name[50];  int type;  double price;  struct product \*nextProPtr;  };  typedef struct product Product;  struct purchased {  int ID;//satılan parça numarası  int invoice\_ID;//fatura no  int customer\_ID;//müşteri no  int product\_ID;//ürün no  double cost;//maliyet  struct purchased \*nextPurchasedPtr;  };  typedef struct purchased Purchased;  void instructions( void );  Customer \*insertCustomer(Customer \*cptr);  Customer \*tekmusteri(Customer \*z,int ID);  void displaycustomer(Customer \*head);  void musteritip(Customer \*cptr,int secim);  void displayurun(Product \*head);  Product \*tekurun(Product \*y,int idd);  void uruntip(Product \*cptr,int secim);  Purchased \*sonfaturano(Purchased\* head,int prevcostumerid,int lastinvoice);  void addpurchased(Purchased\* purchased\_head, Customer\* customer\_head, Product\* product\_head);  void satis\_urun(Purchased \*head,Product \*pro);  void satis\_tutar(Purchased \*head,Product \*pro);  void toplam\_tutar(Purchased \*head,Product \*pro);  void kargo(Purchased \*head,Customer \*cptr);  void uruntutar(Purchased \*head);  #endif |
| IMPLEMENTATION  Explain the method applied for the problem solution |
| İlk olarak müşteriler eklenir.Kullanıcıdan müşteri numarası alınarak satış işlemi başlatılır.Son fatura numarası bulunur.Müşterinin gireceği ürün numarasına göre ürün bilgileri verilir.Miktar ve uzaklık bilgisi ile doğru orantılı olarak toplam maliyet hesaplanır.Bu veriler bağlı listeler özelliği ile tutulur.Eklenen müşteriler ve main de tanımlanmış ürün bilgilerine,analizlere menüden seçilerek ulaşılabilir. |
| TESTING  Briefly describe how you test your code |
| Kod satış kısmında müşteriler eklendikten sonra satış yapmak için müşteri numarası isteyecektir.Müşteri numarası girildikten sonra gelen müşteri bilgilerinin eklediğimiz müşteriyle aynı olması beklenir. Ürün bilgileri (ürün no,ürün adı,ürün tipi,fiyatı) mainde tanımlanmıştır.Satış için ürün numarası istendiğinde gelen ürün bilgilerinin mainde tanımlanmış ürünlerle aynı olması beklenir.  Ürün miktarı girildikten sonra önceden tanımlanmış ürün fiyatına göre maliyetin doğru hesaplanması gerekir.Uzaklıkla doğru orantılı olarak kargo ücretiyle birlikte son maliyet hesaplanır.Menünün müşteri bilgisi kısmında 1. Satış seçeneğinde eklenmiş tüm müşterilerin, müşteri tipine göre müşterilerin(bireysel ya da ticari) veya tek bir müşterinin(ürün numarasına göre) bilgileri listelenmelidir.Menünün Ürün bilgisi kısmında tüm ürünlerin, ürün tipine göre(meyve ya da sebze) ürünlerin veya tek bir ürünün(ürün numarasına göre) bilgileri listelenmelidir.Müşteri analizleri kısmında bir müşterinin satın aldığı ürünler,bir müşterinin satın aldığı toplam tutar ,tüm müşterinin satın aldığı ürünlerin toplam tutarı ,müşterilerin kargo ücreti listelenmelidir. Ürün analizleri kısmında ise bir ürünün toplam satın alınan tutarı önceden saklanmış verilere göre listelenmelidir..Bu veriler bağlı liste ile tutulmaktadır. |
| USER’S GUIDE  (Describe briefly how the user can use your program (input parameters of program, output of program, possible limitations, possible errors, etc.) |
| Satış otomasyon menüsü aşağıdaki gibidir  1. Satış  2. Müşteri Bilgisi  3. Ürün Bilgisi  4. Müşteri Analizleri  5. Ürün Analizleri  Kullanıcı satış seçeneğinde müşteri eklemeli ve daha sonra satış için müşteri numarası girmelidir.  İstediği ürünün numarasını girip miktar belirtmelidir.Bu şekilde ödenecek tutarı öğrenebilir.  Müşteri bilgisinde eklenmiş tüm müşterilerin, müşteri tipine göre müşterilerin(bireysel ya da ticari) veya tek bir müşterinin(ürün numarasına göre) bilgilerini seçimleri doğrultusunda öğrenebilir.  Ürün bilgisinde tüm ürünlerin, ürün tipine göre(meyve ya da sebze) ürünlerin veya tek bir ürünün(ürün numarasına göre) bilgilerini seçimleri doğrultusunda öğrenebilir. Müşteri analizleri kısmında ise bir müşterinin satın aldığı ürünleri veya bir müşterinin satın aldığı ürünlerin toplam tutarını müşteri numarası girerek öğrenebilir..Yine bu menüden tüm müşterinin satın aldığı ürünlerin toplam tutarı, müşterilerin toplam kargo ücreti öğrenilebilir.Ürün analizleri kısmında ürün numarası girilerek bir ürünün toplam satın alınan tutarı listelenir. |
| GENERAL FLOWCHART OF THE PROJECT |
|  |
| CONCLUSION AND REMARKS  You can or can(not) put any remarks on your work. For example;  is your program works or not, if not why  which troubles during performance of project you faced with and how you overcome them;  is your program user friendly and how you can improve it;  what you achieved during performance of project; and so |
| Program kullanıcıdan istenildiği şekilde kullanılırsa düzgün çalışıyor.Örneğin bir sayı girilmesi istendiğinde karakter girilirse hatalı çalışabilir.Bunu daha fazla if else ya da while flag yapılarıyla kontrol ettirerek geliştirebilirim.Satış kısmında satış yapabilmek için öncelikle müşteri eklenmesi gerekir.Müşteri ekleyebilmek için menüden önce satış daha sonra müşteri ekle seçilmelidir.Main de 1 numaralı müşteri ve ürünler test ederken kolaylık olması açısından önceden tanımlanmıştır.Programın sade ve anlaşılır bir arayüzü olduğu için kullanıcı dostu olduğunu düşünüyorum. Menüler arası geçişler de kullandığım döngüler sayesinde oldukça kolay. |
| REFERENCES  Put the list of references and sources (books, links to websites, videos, etc), which you used for project. |
| https://www.youtube.com/watch?v=r3uOBb3BM-0&list=PLh9ECzBB8tJN9bckI6FbWB03HkmogKrFT&index=22  <https://www.youtube.com/watch?v=DGi_gSfYfKo&list=PLh9ECzBB8tJN9bckI6FbWB03Hkmo>  gKrFT&index=2  <https://www.youtube.com/watch?v=wDAf9Er6Qq8&list=PLh9ECzBB8tJN9bckI6FbWB03Hkmo>  gKrFT&index=3  https://www.geeksforgeeks.org/data-structures/linked-list/ |

|  |
| --- |
| APPENDIX |
| PROJECT CODES  Put here code of your project. |
| **my.h**  #ifndef \_MY\_H\_  #define \_MY\_H\_  struct customer {  int ID;  char name[50];  unsigned int type:1;  double x\_coord;  double y\_coord;  struct customer \*nextCusPtr;  };  typedef struct customer Customer;  struct product {  int ID;  char name[50];  int type;  double price;  struct product \*nextProPtr;  };  typedef struct product Product;  struct purchased {  int ID;//satılan parça numarası  int invoice\_ID;//fatura no  int customer\_ID;//müşteri no  int product\_ID;//ürün no  double cost;//maliyet  struct purchased \*nextPurchasedPtr;  };  typedef struct purchased Purchased;  void instructions( void );  Customer \*insertCustomer(Customer \*cptr);  Customer \*tekmusteri(Customer \*z,int ID);  void displaycustomer(Customer \*head);  void musteritip(Customer \*cptr,int secim);  void displayurun(Product \*head);  Product \*tekurun(Product \*y,int idd);  void uruntip(Product \*cptr,int secim);  Purchased \*sonfaturano(Purchased\* head,int prevcostumerid,int lastinvoice);  void addpurchased(Purchased\* purchased\_head, Customer\* customer\_head, Product\* product\_head);  void satis\_urun(Purchased \*head,Product \*pro);  void satis\_tutar(Purchased \*head,Product \*pro);  void toplam\_tutar(Purchased \*head,Product \*pro);  void kargo(Purchased \*head,Customer \*cptr);  void uruntutar(Purchased \*head);  #endif  **main.c**  #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include <math.h>  #include "my.h"  int main(){  //müşteri  Customer \*root;  root=(Customer\*)malloc(sizeof(Customer));  root->ID=1; strcpy(root->name,"abc def");  root->type=0;  root->x\_coord=3.0;  root->y\_coord=4.0;  root->nextCusPtr= NULL;    //ürün  Product \*node;  node=(Product\*)malloc(sizeof(Product));  strcpy(node->name,"elma");  node->ID=1;  node->type=1;//meyve  node->price=5.5;  node->nextProPtr=NULL;  node->nextProPtr=(Product\*)malloc(sizeof(Product));  node->nextProPtr->ID=2;;  node->nextProPtr->price=4.5;  node->nextProPtr->type=2;//sebze  strcpy(node->nextProPtr->name,"lahana");  node->nextProPtr->nextProPtr=NULL ;    node->nextProPtr->nextProPtr=(Product\*)malloc(sizeof(Product));  node->nextProPtr->nextProPtr->ID=3;  node->nextProPtr->nextProPtr->price=3.5;  node->nextProPtr->nextProPtr->type=1;//meyve  strcpy(node->nextProPtr->nextProPtr->name,"armut");  node->nextProPtr->nextProPtr->nextProPtr=NULL;    //satış  Purchased \*pur;  pur=(Purchased\*)malloc(sizeof(Purchased));  pur->nextPurchasedPtr=NULL;  pur->invoice\_ID=0;    int id,secim;  unsigned int choice,sec;  instructions();  scanf( "%u", &choice );  while ( choice !=5) {  switch ( choice ) {  case 1:  system("cls");  printf("musteri eklemek icin 1\nsatis yapmak icin 2\nseciniz:\n");  scanf("%u",&sec);  switch (sec){  case 1:  root=insertCustomer(root);  break;  case 2:  addpurchased(pur,root,node);  break;  default:  puts( "Verilen seçeneklerden birini girmediniz!\n" );  break;  }  break;  case 2: // müşteri bilgisi  printf("islem secin\n");  printf("1. tum musteriler\n2.musteri tipine gore musteriler\n3.tek musteri bulma\n");  scanf("%u",&sec);  switch (sec){    case 1:  displaycustomer(root);  break;  case 2:  printf("Bireysel Musterileri yazdirmak icin 0\nTicari Musterileri yazdirmak icin 1 i girin:");  scanf("%d",&secim);  musteritip(root,secim);  break;  case 3:  printf("musteri id girin:");//2.3 Bir müsteri  scanf("%d",&id);  tekmusteri(root,id);  break;  default:  puts( "Verilen seçeneklerden birini girmediniz!\n" );  break;  }    break;  case 3://ürün bilgisi  printf("islem secin\n");  printf("1. tum urunler\n2.urun tipine gore urunler\n3.tek urun bulma\n");  scanf("%u",&sec);  switch (sec){  case 1:  displayurun(node);//Tüm ürünler  break;  case 2:  secim=0;  printf("Meyveleri yazdirmak icin 1\nSebzeleri yazdirmak icin 2 i girin:");  scanf("%d",&secim);  uruntip(node,secim);//Ürün tipine göre  break;  case 3:  printf("urun id girin:");  scanf("%d",&id);  tekurun(node,id);//Bir ürün  break;  default:  puts( "Verilen seçeneklerden birini girmediniz!\n" );  break;  }  break;  case 4: //müşteri analizleri  printf("islem secin\n");  printf("1.Bir musterinin satin aldigi urunler\n2.Bir musterinin satin aldigi toplam tutar \n3.Tum musterinin satin aldigi urunlerin toplam tutari\n4.Musterilerin kargo ucreti\n");  scanf("%u",&sec);  switch (sec){  case 1:  satis\_urun(pur,node);//Bir müşterinin satın aldığı ürünler  break;    case 2:  satis\_tutar(pur,node);//Bir müşterinin satın aldığı toplam tutar  break;  case 3:  toplam\_tutar(pur,node);//Tüm müşterinin satın aldığı ürünlerin toplam tutarı  break;  case 4:  kargo(pur,root);// Müşterilerin kargo ücreti  break;  default:  puts( "Verilen seçeneklerden birini girmediniz!\n" );  break;  }  break;  case 5://ürün analizi  uruntutar(pur);  break;    default:  puts( "Invalid choice.\n" );  break;  } // end switch  instructions();  scanf( "%u", &choice );  } // end while    return 0;  }  **my.c**  #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include <math.h>  #include "my.h"  void instructions( void )  {  puts( "\nsecim yapiniz\n"  "1. Satis \n2. Musteri Bilgisi \n3. Urun Bilgisi \n4. Musteri Analizleri \n5. Urun Analizleri" );  }  Customer \*insertCustomer(Customer \*cptr){  Customer\* iter;  iter = cptr;  int idx;  char name[50];  unsigned int tip;  double x,y;  printf("Musteri Numarasi Giriniz\n") ;  scanf("%d",&idx) ;  printf("musteri adi girin:");  scanf(" %[^\n]",name);  printf("musteri tipi secin ticari ise 1, bireysel ise 0:");  scanf("%d",&tip);  printf("x koordinatini girin:");  scanf("%lf",&x);  printf("y koordinatini girin:");  scanf("%lf",&y);  if( iter->ID == 0){  iter = (Customer\*)malloc(sizeof(Customer));  iter->ID = idx;  strcpy(cptr->name,name);  iter->type = tip;  iter->x\_coord = x;  iter->y\_coord = y;  iter->nextCusPtr = NULL ;  printf("%d %s %lf %lf %u\n bilgileri girilmistir..\n\n",iter->ID, iter->name,  iter->x\_coord, iter->y\_coord, iter->type );  cptr = iter ;  return cptr ;  }  else{  while(iter->nextCusPtr != NULL){  iter = iter -> nextCusPtr ;  }  iter->nextCusPtr = (Customer\*)malloc(sizeof(Customer));  iter = iter->nextCusPtr;  iter->nextCusPtr = NULL ;  strcpy(iter -> name, name) ;  iter -> ID = idx ;  iter -> type = tip ;  iter -> x\_coord = x ;  iter -> y\_coord = y ;  printf("%d %s %.2lf %.2lf %u\n bilgileri girilmistir..\n\n",iter->ID, iter->name, iter->x\_coord, iter->y\_coord, iter->type );  return cptr ;  }  }  Customer \*tekmusteri(Customer \*z,int ID){  system("cls");  if(z==NULL){  printf("musteri listesi bos!!");  }  else{  while(z->nextCusPtr!=NULL){  if(ID==z->ID){  printf("\nmusteri numarasi :%d\n",z->ID);  printf("musteri adi :%s\n",z->name);  printf("musteri tipi :%d\n",z->type);  printf("x koordinati :%.2lf\n",z->x\_coord);  printf("y koordinati :%.2lf\n",z->y\_coord);  }  z=z->nextCusPtr;  }  if(ID==z->ID){  printf("\nmusteri numarasi :%d\n",z->ID);  printf("musteri adi :%s\n",z->name);  printf("musteri tipi :%d\n",z->type);  printf("x koordinati :%.2lf\n",z->x\_coord);  printf("y koordinati :%.2lf\n",z->y\_coord);  }  printf("\n");  }  return z;  }  void displaycustomer(Customer \*head){  system("cls");  if(head==NULL){  printf("musteri listesi bos!!");  }  else{  Customer \*p=head;  while(head->nextCusPtr!= NULL){  printf("musteri no:%d adi:%s koordinatlari:%.2lf,%.2lf tipi:%u \n",head->ID, head->name, head->x\_coord, head->y\_coord, head->type );  head=head->nextCusPtr;  }  printf("------------\n");  printf("musteri no:%d adi:%s koordinatlari:%.2lf,%.2lf tipi:%u \n\n",head->ID, head->name, head->x\_coord, head->y\_coord, head->type );  }    }  void musteritip(Customer \*cptr,int secim){  system("cls");  Customer \*iter=cptr;  while(iter->nextCusPtr != NULL )  {  if(iter->type == secim){  printf("musteri no : %d\nmusteri adı : %s\nmusteri tipi : %u\n", iter->ID, iter->name, iter->type);  }  iter = iter->nextCusPtr;  }    if(iter->type == secim){  printf("musteri no : %d\nmusteri adı : %s\nmusteri tipi : %u\n", iter->ID, iter->name, iter->type);  }  }  void displayurun(Product \*head){  system("cls");  printf("\ntum urunler:\n\n");  while(head!= NULL){  printf("%d %s %.2lf %u \n",head->ID, head->name, head->price, head->type );  head=head->nextProPtr;  }  printf("\n\n");  }  Product \*tekurun(Product \*y,int idd){  system("cls");  if(y==NULL){  printf("urun listesi bos!!");  }  else{  Product \*p=y;  while(p!=NULL){  if(p->ID==idd){  printf("\nurun numarasi :%d\n",p->ID);  printf("urun adi :%s\n",p->name);  printf("urun tipi :%d\n",p->type);  printf("urun ucreti :%.2lf\n",p->price);  }  p=p->nextProPtr;  }  printf("\n");  }  return y;  }  void uruntip(Product \*cptr,int secim){  system("cls");  Product \*iter=cptr;  while(iter->nextProPtr != NULL )  {  if(iter->type == secim){  printf("urun no : %d\nurun adi : %s\nurun tipi : %u\nurun ucreti : %.2lf", iter->ID, iter->name, iter->type,iter->price);  }  iter = iter->nextProPtr;  }    if(iter->type == secim){  printf("urun no : %d\nurun adi : %s\nurun tipi : %u\nurun ucreti : %.2lf", iter->ID, iter->name, iter->type,iter->price);  }  }  Purchased \*sonfaturano(Purchased\* head,int prevcostumerid,int lastinvoice){  lastinvoice = head->invoice\_ID ;  Purchased\* iter = head;  if(iter->nextPurchasedPtr == NULL)  {  iter->invoice\_ID++;  return iter;  }  while(iter->nextPurchasedPtr != NULL)  { prevcostumerid=iter->customer\_ID ;  if(iter->customer\_ID == prevcostumerid )  {  iter->invoice\_ID = lastinvoice;  }  else  {  iter->invoice\_ID++;  lastinvoice = iter->invoice\_ID ;  }  iter = iter->nextPurchasedPtr ;  return iter;  }  }  void addpurchased(Purchased\* purchased\_head, Customer\* customer\_head, Product\* product\_head)  {  system("cls");  int prevcustomerid, prevdinvoiceid ;  Purchased\* purchasedx = purchased\_head ;  Customer\* customerx = customer\_head ;  Product\* productx = product\_head ;  int idx, adet=0,flag=0;  double sumcost=0;  while( flag == 0 )  {  printf("satis icin musteri Numarasi Giriniz\n") ;  scanf("%d",&idx) ;  customerx = tekmusteri(customer\_head, idx) ;  if(customerx == NULL)  {  printf("\nDevam etmek icin 0\n Cıkmak icin 99 girin") ;  scanf("%d",&flag) ;  }  else{  flag = 1;  }  while( flag == 1 )  {  purchasedx = sonfaturano(purchased\_head,prevcustomerid,prevdinvoiceid);  if(prevcustomerid == purchasedx->customer\_ID)  {  purchasedx->invoice\_ID = prevdinvoiceid ;  }  else  {  purchasedx->invoice\_ID++ ;  prevcustomerid = purchasedx->customer\_ID ;  }  printf("Sepete Urun eklemek icin urun no giriniz\n") ;  scanf("%d",&idx) ;  productx = tekurun(product\_head,idx) ;  if( productx == NULL )  {  printf("Urun bulunamadı!!!! Satis devam icin 1 cikmak icin 99 giriniz\n");  scanf("%d",&flag);  continue ;  }  printf("Urun Adedi giriniz\n") ;  scanf("%d",&adet) ;  if( adet == 0 )  {  printf("Urun adet 0 olarak girildi!!\n") ;  printf("Satisa devam icin 1 cikmak icin 99 giriniz\n");  scanf("%d",&flag);  continue ;  }  else  {  purchasedx->cost = (adet)\*(productx->price);//sepet tutarı  sumcost+= (adet)\*(productx->price) ;//toplam maliyet  }  purchasedx->ID++ ;  purchasedx->product\_ID = productx->ID ;  purchasedx->customer\_ID = customerx->ID ;  printf("Tekrar urun eklemek icin 1 giriniz, \nToplam maliyet icin 2,\nUst menu icin 0, \n") ;  scanf("%d",&flag) ;  if( flag == 2 )  {  printf("Sepet Tutari : %.2lf TL'dir\n",purchasedx->cost ) ;  if( sumcost == 0 )  {  printf("Urun adeti girilmedi devam etmek icin 0 girin\n") ;  scanf("%d",&flag) ;  break ;  }  else  {  sumcost += 2\*(sqrt(pow(customerx->x\_coord,2)+pow(customerx->y\_coord,2)));  printf("Odenecek Tutar : %.2lf TL'dir \n",sumcost ) ;  }  }  }  }  return;  }  void satis\_urun(Purchased \*head,Product \*pro){  int idd;  Product \*xx=pro;  printf("musteri no giriniz:\n");  scanf("%d",&idd);  if(head==NULL){  printf("satis yok");  }  else{  Purchased \*sat = head;  while(sat!=NULL){  if(idd==sat->customer\_ID){  xx=tekurun(pro,sat->product\_ID);  }  sat=sat->nextPurchasedPtr;  }  }  return;  }  void satis\_tutar(Purchased \*head,Product \*pro){  int idd;  Product \*xx=pro;  printf("musteri no giriniz:\n");  scanf("%d",&idd);  if(head==NULL){  printf("satis yok");  }  else{  Purchased \*sat = head;  while(sat!=NULL){  if(idd==sat->customer\_ID){  printf("musterinin satin aldigi urunlerin toplam tutari:%.2lf TL\n",sat->cost);  }  sat=sat->nextPurchasedPtr;  }  }  return;  }  void toplam\_tutar(Purchased \*head,Product \*pro){  double counter=0;  Product \*xx=pro;  if(head==NULL){  printf("satis yok");  }  else{  Purchased \*sat = head;  while(sat->nextPurchasedPtr!=NULL){  counter+=sat->cost;  sat=sat->nextPurchasedPtr;  }  counter+=sat->cost;  printf("tum musterilerin satin aldigi urunlerin toplam tutari:%.2lf TL\n",counter);  }  return;  }  void kargo(Purchased \*head,Customer \*cptr){  double sumcost;  if(head==NULL){  printf("satis yok");  }  else{  Customer \*cus=cptr;  Purchased \*sat = head;  while(sat->nextPurchasedPtr!=NULL){  sumcost += 2\*(sqrt(pow(cus->x\_coord,2)+pow(cus->y\_coord,2)));  sat=sat->nextPurchasedPtr;  }  sumcost += 2\*(sqrt(pow(cus->x\_coord,2)+pow(cus->y\_coord,2)));  printf("Odenecek Tutar : %.2lf TL'dir \n",sumcost ) ;  }  return;  }  void uruntutar(Purchased \*head){  int idd;  printf("urun no girin:\n");  scanf("%d",&idd);  if(head==NULL){  printf("satis yok");  }  else{  Purchased \*sat = head;  while(sat!=NULL){  if(idd==sat->product\_ID){  printf("musterinin satin aldigi toplam urun tutari:%.2lf TL\n",sat->cost);  }  sat=sat->nextPurchasedPtr;  }  }  return;  } |