

République Tunisienne  
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique  
Université de Carthage  
École Nationale d'Ingénieurs de Carthage

# Rapport de Projet Programmation C Site E-Commerce

Département Génie Informatique  
**1ère Info Groupe D**

Réaliser par :  
**CHEBL YOUSSEF**  
**MAAMAR MOHAMED**

Année Universitaire 2023-2024

# TABLE DES MATIÈRES

1.	Fichier structure (MyLibrary.h): .....	1
2.	Fichier prototypes (MyFunctions.h): .....	2
3.	Code source complet : .....	4
3.1.	MyFunctions.cpp .....	4
3.2.	ECommerce.cpp(code menu) .....	19
4.	Test exécution 1 (categorie): .....	25
5.	Test exécution 2 (Produit): .....	29
6.	Test exécution 3 (client) : .....	32
7.	Test execution 4 (commande): .....	34
8.	Test fichiers : .....	36
9.	Test tableau dynamique des clients : .....	39
10.	Menu du programme (Execution): .....	40

## 1. Fichier structure (MyLibrary.h):

---

```
//----- LES  STRUCTURES -----//
```

---

```
struct product {
    int productID;
    char name[100];
    char description[500];
    float price;
    int stockQuantity;
    char brand[50] ;
};
typedef struct product  PRODUCT ;

struct category {
    int categoryID;
    char name[50];
    PRODUCT * products;
    int numProducts ; // Le nombre de produits dans une categorie
};
typedef struct category CATEGORY ;

struct customer {
    int customerID;
    char name[100];
    char email[100];
    int  nbrOrders ;
};
typedef struct customer CUSTOMER ;

struct order {
    int orderID;
    CUSTOMER * customers;
    CATEGORY * categories;
    int  nbrProducts;
    int  nbrCategories;
    float totalAmount ;
};
typedef struct order ORDER ;

struct result {
    int orderID;
    float totalAmount ;
};
typedef struct result RESULT ;
//----- FIN -----//
```

## 2. Fichier prototypes (MyFunctions.h):

```
//***** LES  PROTOTYPES  
*****//
```

```
#include "MyLibrary.h"
```

---

```
//----- Fonctions d'ajout -----//
```

---

```
void addCategory (CATEGORY **, int *) ;  
void addProduct (PRODUCT **, CATEGORY *) ;  
void addProductToCategory (PRODUCT **, int *, CATEGORY *, int);  
void addCustomer (CUSTOMER **, int *) ;  
void createOrder ( CUSTOMER *, CATEGORY *, ORDER **, int *, int, int) ;
```

---

```
//----- Fonctions pour trouver les indexs -----//
```

---

```
int findCustomerIndex (int, CUSTOMER *, int) ;  
int findProductIndex (int, PRODUCT *, int) ;  
int findCategoryIndex (int, CATEGORY *, int) ;  
int findOrderIndex (int, ORDER *,int );
```

---

```
//----- Fonctions de suppression -----  
//
```

---

```
void removeProductFromCategory(CATEGORY *, int, int, int);  
void removeCustomer(CUSTOMER *, int, int) ;  
void removeOrder(ORDER *, int, int);  
void removeCustomer(CUSTOMER *, int, int);  
void removeCategory(CATEGORY **, int *, int);
```

---

```
//----- Fonctions pour l'affichage -----  
-----//
```

---

```

float calculateCategoryTotal(CATEGORY *) ;
void displayProduct(PRODUCT) ;
void displayCategory(CATEGORY) ;
void displayCustomer(CUSTOMER);
void displayOrder(ORDER) ;
void displayOrders(ORDER *, int) ;
void displayAllCategories(CATEGORY *, int) ;
void displayProductDetails(CATEGORY *, int, int);
void displayResult(RESULT) ;
void displayResults(RESULT *, int);

```

---

```

//----- Fonctions pour tester l existence des ID -----//

```

---

```

int TestIDCat (int, CATEGORY **, int *) ;
int TestIDPro (int, PRODUCT **, int) ;
int TestIDCli (int, CUSTOMER **, int *) ;

```

---

```

//----- Result -----//

```

---

```

void fillResults(RESULT ***, ORDER *, int);
int findMostExpensiveOrderIndex(RESULT **, int);
void displayMostExpensiveOrder(RESULT **, int) ;

```

---

```

//----- Fonctions pour les fichier -----
-----//

```

---

```

void createfile(FILE **,FILE **);
void fillfile(FILE *,FILE *);
void addCustomer(CUSTOMER * );
CUSTOMER readcustomer (FILE *);
void displayfile(FILE *,FILE *);
void modifyCustomer(FILE *, FILE *, int);

```

```

//***** FIN *****//

```

### 3. Code source complet :

#### 3.1. *MyFunctions.cpp*

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <locale.h> // pour appeler la fonction - setlocale -

#include "MyFunctions.h"



---


//----- Fonctions d'ajout -----//


---



// Ajouter une catégorie
void addCategory(CATEGORY **categories, int *categoryCount) {
    CATEGORY newCategory;

    do {
        printf("\nEntrez l'id de la catégorie ( > 0 ): ");
        scanf("%d", &newCategory.categoryID);
        if (TestIDCat(newCategory.categoryID, categories, categoryCount) == 1) printf("\nDeja exist :");

        } while ( (isdigit(newCategory.categoryID)) || (newCategory.categoryID <= 0) ||
        (TestIDCat(newCategory.categoryID, categories, categoryCount) == 1) );
        //test pour id

    do{
        printf("\nEntrez le nom de la catégorie (le premiere doit etre un caractere): ");
        scanf("%s", newCategory.name);
        } while( !isalpha(newCategory.name[0]) );

    newCategory.products = NULL;
    newCategory.numProducts = 0;

    // Ajouter nouvelle catégorie
    *categories = (CATEGORY *)realloc(*categories, (*categoryCount + 1) * sizeof(CATEGORY));
    (*categories+*categoryCount) = newCategory;
    (*categoryCount)++;

    printf("\n----- NB ----- \n");
    printf("Catégorie ajoutée avec succès\n");
}

// Ajouter produit à une catégorie selectionné
void addProduct(PRODUCT **products, CATEGORY *category) {
    PRODUCT newProduct;
```

```

do{
printf("Entrez l'identifiant du produit : ");
scanf("%d", &newProduct.productID);
    } while ( (newProduct.productID <= 0) ||
(TestIDPro(newProduct.productID,products,category->numProducts) ==1 ) );
    //test pour id

    do{
printf("Entrez le nom du produit : ");
scanf("%s", newProduct.name);
    } while( !isalpha(newProduct.name[0]) );

    do{
printf("Entrez la description du produit : ");
scanf("%s", newProduct.description);
    } while( !isalpha(newProduct.description[0]) );

    do{
printf("Entrez la marque du produit : ");
scanf("%s", newProduct.brand);
    } while( !isalpha(newProduct.brand[0]) );

    do{
printf("Entrez le prix du produit : ");
scanf("%f", &newProduct.price);
    } while(isdigit(newProduct.price) );

    do{
printf("Entrez la quantité en stock du produit : ");
scanf("%d", &newProduct.stockQuantity);
    } while(isdigit(newProduct.stockQuantity) );

    // Ajouter nouvelle produit a la catégorie
category->products = (PRODUCT *)realloc(category->products, (category->numProducts + 1) *
sizeof(PRODUCT));
*(category->products+category->numProducts) = newProduct;
category->numProducts++;

    printf("\n----- NB ----- \n");
printf("Produit ajouté avec succès à la catégorie\n");
}

// Ajouter produit a une catégorie exist
void addProductToCategory(PRODUCT **products, int *productCount, CATEGORY *categories, int
categoryCount) {
    int productID, quantity, categoryID;
    CATEGORY *selectedCategory;

    do{
printf("Entrez l'identifiant de la catégorie à laquelle ajouter le produit : ");
scanf("%d", &categoryID);
    }while (findCategoryIndex(categoryID, categories, categoryCount)==-1);

```

```

// Vérifier si la catégorie choisie existe
int existingCategoryIndex = findCategoryIndex(categoryID, categories, categoryCount);

if (existingCategoryIndex != -1) {
    selectedCategory = (categories+existingCategoryIndex);
}
printf("\nEntrez l'identifiant du produit : ");
scanf("%d", &productID);

// Vérifier si le produit exist
int existingProductIndex = findProductIndex(productID, categories->products, categories-
>numProducts);

if (existingProductIndex != -1) {
    printf("\nLe produit existe déjà dans la catégorie. \nEntrez la quantité à ajouter : ");
    scanf("%d", &quantity);

    // Ajout la quantité au stock
    selectedCategory->products[existingProductIndex].stockQuantity += quantity;

    printf("\n----- NB ----- \n");
    printf("Quantité ajoutée avec succès au produit existant dans la catégorie\n");
} else {
    // Le produit n exist pas, ajouter un nouveau produit
    addProduct(products, selectedCategory);

    printf("\n----- NB ----- \n");
    printf("Nouveau produit ajouté à la catégorie\n");
}
}

// Ajouter un client
void addCustomer(CUSTOMER **customers, int *customerCount) {
    CUSTOMER newCustomer;

    do{
        printf("Entrez l'identifiant du client : ");
        scanf("%d", &newCustomer.customerID);
    } while ((newCustomer.customerID <= 0) ||
(TestIDCli(newCustomer.customerID,customers,customerCount) == 1 ));

    do{
        printf("Entrez le nom du client : ");
        scanf("%s", newCustomer.name);
    } while(isdigit(newCustomer.name[0]) );

    printf("Entrez l'e-mail du client : ");
    scanf("%s", newCustomer.email);

```



```

newCustomer.nbrOrders = 0;

// Ajouter nouveau client
*customers = (CUSTOMER *)realloc(*customers, (*customerCount + 1) * sizeof(CUSTOMER));
>(*customers+*customerCount) = newCustomer;
(*customerCount)++;

printf("\n----- NB ----- \n");
printf("Client ajouté avec succès\n");
}

```

---

```

//----- Fonctions pour trouver les indexes -----
--//

```

---

```

// Trouver l'index du produit
int findProductIndex(int productID, PRODUCT *products, int numProducts) {
    for (int i = 0; i < numProducts; i++) {
        if ((products+i)->productID == productID) {
            return i; // Retourne l'index si trouvr
        }
    }
    return -1; // sinon retourne -1
}

// Trouver l'index du categorie
int findCategoryIndex(int categoryID, CATEGORY *categories, int categoryCount) {
    for (int i = 0; i < categoryCount; i++) {
        if ((categories+i)->categoryID == categoryID) {
            return i;
        }
    }
    return -1;
}

// Trouver l'index du client
int findCustomerIndex(int customerID, CUSTOMER *customers, int numCustomers) {
    for (int i = 0; i < numCustomers; i++) {
        if ((customers+i)->customerID == customerID) {
            return i;
        }
    }
    return -1;
}

// Trouver l'index du commande
int findOrderIndex(int orderID, ORDER *orders, int numOrders) {
    for (int i = 0; i < numOrders; i++) {
        if ((orders+i)->orderID == orderID) {
            return i;
        }
    }
}

```

```

    }
    return -1;
}

```

---

```

//----- Fonctions de suppression -----//

```

---

```

// supprimer categorie

```

```

void removeCategory(CATEGORY **categories, int *categoryCount, int categoryID) {
    int categoryIndex = findCategoryIndex(categoryID, *categories, *categoryCount);

```

```

    if (categoryIndex != -1) {
        // Libérer la mémoire pour les produits
        free((*categories)[categoryIndex].products);

```

```

        for (int i = categoryIndex; i < *categoryCount - 1; i++)
        {
            (*categories)[i] = (*categories)[i + 1];
            // *(categories+i) = *(categories+i + 1);
        }

```

```

        (*categoryCount)--;

```

```

        printf("\n----- NB ----- \n");
        printf("Catégorie supprimée avec succès\n");
    } else {
        printf("\n----- NB ----- \n");
        printf("Catégorie introuvable.\n");
    }
}

```

```

// supprimer produit

```

```

void removeProductFromCategory(CATEGORY *categories, int categoryCount, int categoryID, int
productID) {

```

```

    for (int i = 0; i < categoryCount; i++) {
        if ((categories + i)->categoryID == categoryID)
        {
            PRODUCT *products = (categories + i)->products;
            int numProducts = (categories + i)->numProducts;

```

```

            int productIndex = findProductIndex(productID, products, numProducts);

```

```

            if (productIndex != -1) {
                for (int j = productIndex; j < numProducts - 1; j++)
                {

```

```

//                *(products+j) = *(products+j + 1);
                    products[j] = products[j + 1];

```

```

                }
                categories[productIndex].numProducts--;

```

```

        printf("\n----- NB ----- \n");
        printf("Produit supprimé avec succès de la catégorie\n");
    } else
    {
        printf("\n----- NB ----- \n");
        printf("Produit introuvable dans la catégorie\n");
    }
    return;
}
}
    printf("\n----- NB ----- \n");
    printf("Catégorie introuvable\n");
}

// supprimer client
void removeCustomer(CUSTOMER *customers, int customerCount, int customerID) {
    int customerIndex = findCustomerIndex(customerID, customers, customerCount);

    if (customerIndex != -1) {
        for (int i = customerIndex; i < customerCount - 1; i++)
        {
            customers[i] = customers[i + 1];
            /*(customers+i) = *(customers+i + 1);
        }
        printf("\n----- NB ----- \n");
        printf("Client supprimé avec succès\n");
    } else {
        printf("\n----- NB ----- \n");
        printf("Client introuvable\n");
    }
}

// supprimer commande
void removeOrder(ORDER *orders, int orderCount, int orderID) {
    int orderIndex = findOrderIndex(orderID, orders, orderCount);

    if (orderIndex != -1) {
        for (int i = orderIndex; i < orderCount - 1; i++) {
            /*(orders+i) = *(orders+i+ 1);
            orders[i] = orders[i + 1];
        }
        printf("\n----- NB ----- \n");
        printf("Commande supprimée avec succes\n");
    } else {
        printf("\n----- NB ----- \n");
        printf("Commande introuvable\n");
    }
}

```

---

```
//-----La création du commande -----  
-//
```

---

```
//creation d une commande  
void createOrder(CUSTOMER *customers, CATEGORY *categories, ORDER **orders, int *orderCount,  
int categoryCount, int customerCount) {  
    ORDER newOrder;  
    newOrder.customers = NULL;  
    newOrder.categories = NULL;  
    newOrder.nbrProducts = 0;  
    newOrder.nbrCategories = 0;  
    newOrder.totalAmount = 0.0;  
  
    int clientID;  
    printf("Entrez l'identifiant du client : ");  
    scanf("%d", &clientID);  
  
    for (int i = 0; i < customerCount; i++) {  
        if ( (customers+i)->customerID == clientID) {  
            printf("\nID Client trouvé :)\n");  
            newOrder.customers = (customers+i);  
                                //newOrder.customers = &customers[i];  
            break;  
        }  
    }  
  
    if (newOrder.customers == NULL) {  
        printf("Client introuvable, ajouter le avant de crée une commande :)\n");  
        return;  
    }  
  
    // Identifier la catégorie  
    int categoryID;  
    printf("Entrez l'identifiant de la catégorie : ");  
    scanf("%d", &categoryID);  
  
    // Rechercher la catégorie  
    CATEGORY *selectedCategory = NULL;  
    for (int i = 0; i < categoryCount; i++) {  
        if ((categories+i)->categoryID == categoryID) {  
            selectedCategory = categories+i;  
                                //selectedCategory = &categories[i];  
            break;  
        }  
    }  
  
    if (selectedCategory == NULL) {  
        printf("Catégorie introuvable\n");  
        return;  
    }  
}
```

```

int productID;
printf("Entrez l'identifiant du produit dans la catégorie : ");
scanf("%d", &productID);

// Rechercher le produit dans catégorie
PRODUCT *selectedProduct = NULL;
for (int i = 0; i < selectedCategory->numProducts; i++) {
    if ((selectedCategory->products+i)->productID == productID) {
        printf("\nProduit trouvé dans la catégorie :)\n");
        //selectedProduct = &selectedCategory->products[i];
        selectedProduct = (selectedCategory->products+i);
        newOrder.nbrProducts+=1 ;
        break;
    }
}

if (selectedProduct == NULL) {
    printf("Produit introuvable\n");
    return;
}

int quantity;
printf("Entrez la quantité : ");
scanf("%d", &quantity);

// Allouer la mémoire pour la nouvelle commande
*orders = (ORDER *)realloc(*orders, (*orderCount + 1) * sizeof(ORDER));

// Initialiser la nouvelle commande
newOrder.orderID = *orderCount+1 ;
newOrder.categories = (CATEGORY *)malloc(sizeof(CATEGORY));
newOrder.categories[0] = *selectedCategory;
newOrder.nbrCategories = 1;

// Allouer pour les produits du nouvelle commande
newOrder.categories[0].products = (PRODUCT *)malloc(quantity * sizeof(PRODUCT));
newOrder.nbrProducts = quantity;

// Copier le produit dans la commande
for (int i = 0; i < quantity; i++) {
    *(newOrder.categories[0].products+i) = *selectedProduct;
    newOrder.totalAmount += selectedProduct->price;
}

// Incrémenter le nombre de commandes du client
newOrder.customers->nbrOrders++;

// Ajouter la commande au tab d'ordre
(*orders)[*orderCount] = newOrder;
(*orderCount)++;

```

```

printf("\n----- NB ----- \n");
printf("Commande créée avec succès\n");
}

```

---

```

//----- Fonctions pour l'affichage -----
-----//

```

---

```

// Afficher un seul produit
void displayProduct(PRODUCT product)
{
    printf("ID du produit : %d\n", product.productID);
    printf("Nom du produit : %s\n", product.name);
    printf("Description : %s\n", product.description);
    printf("Prix : %.2f\n", product.price);
    printf("Quantité en stock : %d\n", product.stockQuantity);
    printf("Marque : %s\n", product.brand);
    printf("\n----- \n");
}

// Afficher une seule catégorie
void displayCategory(CATEGORY category)
{
    printf("ID de la catégorie : %d\n", category.categoryID);
    printf("Nom de la catégorie : %s\n", category.name);
    printf("Nombre de produit : %d\n", category.numProducts);

    printf("\n----- Liste des produits ----- \n");
    // Afficher les produits de la catégorie
    for (int i = 0; i < category.numProducts; i++)
    {
        displayProduct(*(category.products+i));
    }
    printf("\nLe montant total de la catégorie est : %.2f", calculateCategoryTotal(&category));
    printf("\n----- \n");
}

// calculer le montant total de la catégorie
float calculateCategoryTotal(CATEGORY *category)
{
    float total = 0.0;
    for (int i = 0; i < category->numProducts; i++)
    {
        total += (category->products+i)->price;
    }
    return total;
}

```

```

// Afficher un client
void displayCustomer(CUSTOMER customer)
{
    printf("ID du client : %d\n", customer.customerID);
    printf("Nom du client : %s\n", customer.name);
    printf("E-mail du client : %s\n", customer.email);
    printf("Nombre des commandes : %d\n", customer.nbrOrders);
    printf("\n-----\n");
}

// Afficher une commande
void displayOrder(ORDER order)
{
    printf("\n-----\n");
    printf("ID de la commande : %d\n", order.orderID);
    printf("Client de la commande : %s\n", order.customers->name);
    printf("Nombres des categories: %d\n", order.nbrCategories);
    printf("Montant totale de la commande : %.2f\n", order.totalAmount);
    printf("Nombre des produits dans la commande : %d", order.nbrProducts);
    printf("\n-----\n");
}

// Afficher un produit
void displayProductDetails(CATEGORY *categories, int categoryCount, int productID) {
    for (int i = 0; i < categoryCount; i++) {
        for (int j = 0; j < ((*categories+i)).numProducts; j++) {
            if ( ((*categories+i)).products[j].productID == productID) {
                printf("\nDétails du Produit :\n");
                printf("ID du Produit: %d\n", ((*categories+i)).products[j].productID);
                printf("Nom du Produit: %s\n", ((*categories+i)).products[j].name);
                printf("Description: %s\n", ((*categories+i)).products[j].description);
                printf("Marque: %s\n", ((*categories+i)).products[j].brand);
                printf("Prix: %.2f\n", ((*categories+i)).products[j].price);
                printf("Quantité en Stock: %d\n", ((*categories+i)).products[j].stockQuantity);
                return;
            }
        }
    }
    printf("\n----- NB ----- \n");
    printf("Produit introuvable\n");
}

// Afficher toutes les catégories
void displayAllCategories(CATEGORY *categories, int categoryCount)
{
    printf("\n*-----* Toutes les catégories *-----*\n");
    for (int i = 0; i < categoryCount; i++)
    {
        displayCategory((*categories+i));
    }
    printf("\n-----\n");
}

```

```

// Afficher toutes les orders
void displayOrders(ORDER *orders, int orderCount)
{
    printf("\n*-----* Toutes les orders *-----*\n");
    for (int i = 0; i < orderCount; i++)
    {
        displayOrder(*(orders+i));
    }
    printf("\n-----\n");
}

// Afficher une seule structure RESULT
void displayResult(RESULT result) {
    printf("Order ID: %d, Total Amount: %.2f\n", result.orderID, result.totalAmount);
}

// Affichage du Liste RESULT
void displayResults(RESULT *resultArray, int numResults) {
    printf("\n----- Liste des Résultats ----- \n");

    for (int i = 0; i < numResults; i++) {
        displayResult(*(resultArray+i) );
    }
    printf("\n-----\n");
}

```

---

*//----- Fonctions pour tester l'existence des ID -----//*

---

```

int TestIDCat (int ID, CATEGORY **Tab, int * TabSize)
{
    for (int i = 0; i < *TabSize; i++) {
        if ((*Tab+i)->categoryID == ID)
        {
            return 1;
        }
    }
    return 0;
}

int TestIDPro (int ID, PRODUCT **Tab, int TabSize)
{
    for (int i = 0; i < TabSize; i++) {
        if ((*Tab+i)->productID == ID)
        {
            return 1;
        }
    }
    return 0;
}

```



```

}
int TestIDCli (int ID, CUSTOMER **Tab, int *TabSize)
{
    for (int i = 0; i < *TabSize; i++) {
        if ((*Tab+i)->customerID == ID)
        {
            return 1;
        }
    }
    return 0;
}

```

---

*//----- Result -----//*

---

```

// Fonction pour remplir le tableau dynamique de structures RESULT
void fillResults(RESULT ***resultArray, ORDER *orders, int orderCount) {
    *resultArray = (RESULT **)malloc(orderCount * sizeof(RESULT*));
    if (resultArray==NULL)
    {
        exit(-2);
    }
    else
    {
        for (int i = 0; i < orderCount; i++) {
            (*resultArray)[i] = (RESULT *)malloc(sizeof(RESULT));
            (*resultArray)[i]->orderID = orders[i].orderID;
            (*resultArray)[i]->totalAmount = orders[i].totalAmount;
//            ((*resultArray)+i) = (RESULT *)malloc(sizeof(RESULT));
//            ((*resultArray)+i)->orderID = (*(orders+i)).orderID;
//            ((*resultArray)+i)->totalAmount = (*(orders+i)).totalAmount;
        }
    }
    printf("\n----- NB ----- \n");
    printf("remplissage avec succès :)");

    printf("\n----- Liste des Commandes ----- \n");
    for (int i = 0; i < orderCount; i++) {
        printf("Order ID: %d, \nTotal Amount: %.2f\n", ((*resultArray)+i)->orderID, ((*resultArray)+i)->totalAmount);
    }
}

// Fonction pour trouver l'index de la commande la plus chère
int findMostExpensiveOrderIndex(RESULT ***resultArray, int orderCount) {
    if (orderCount == 0) {
        return -1; // Pas de commandes
    }
}

```

```

int maxIndex = 0;
float maxAmount = resultArray[0]->totalAmount;

for (int i = 1; i < orderCount; i++) {
    if ((*resultArray+i)->totalAmount > maxAmount) {
        maxIndex = i;
        maxAmount = (*resultArray+i)->totalAmount;
    }
}

return maxIndex;
}

// Fonction pour afficher la commande la plus chère
void displayMostExpensiveOrder(RESULT **resultArray, int orderCount)
{
    int mostExpensiveIndex = findMostExpensiveOrderIndex(resultArray, orderCount);

    if (mostExpensiveIndex != -1)
    {
        printf("\n--- Commande la plus chère ---\n");
        printf("Order ID: %d, \nTotal Amount: %.2f\n", resultArray[mostExpensiveIndex]->orderId,
resultArray[mostExpensiveIndex]->totalAmount);
    } else
    {
        printf("\n----- NB ----- \n");
        printf("\nAucune commande disponible.\n");
    }
}

```

---

*//----- Fonctions pour les fichier -----*

---

```

void addCustomer(CUSTOMER * c)
{
    printf("Entrez l'identifiant du client : ");
    scanf("%d", &c->customerID);
    printf("Entrez le nom du client : ");
    scanf("%s", c->name);
    printf("Entrez l'e-mail du client : ");
    scanf("%s", c->email);
    printf("Entrez le nombre d'order du client : ");
    scanf("%d", &c->nbrOrders);
}

```

```

CUSTOMER readcustomer (FILE *f)
{

```

```

CUSTOMER c;
fread(&c.customerID,sizeof(int),1,f);
fread(&c.name,50*sizeof(char),1,f);
fread(&c.email,50*sizeof(char),1,f);
fread(&c.nbrOrders,sizeof(int),1,f);
return c;
}

void createfile(FILE **fp,FILE **fi)
{
    *fp=fopen("Y:\\ENICarthage/cours/1st Sem/C/Projet_E_Commerce/file_res","wb+");
    if(!*fp) exit(-1);
    *fi=fopen("Y:\\ENICarthage/cours/1st Sem/C/Projet_E_Commerce/index_file_res","wb+");
    if(!*fi) exit(-1);
}

void fillfile(FILE *f,FILE *findex)
{
    CUSTOMER c;
    int x,stop=1;
    do
    {
        addCustomer(&c);
        if (stop == 0) break;
        x=ftell(f);
        fwrite(&x,sizeof(int),1,findex);
        fwrite(&c.customerID,sizeof(int),1,f);
        fwrite(&c.name,50*sizeof(char),1,f);
        fwrite(&c.email,50*sizeof(char),1,f);
        fwrite(&c.nbrOrders,sizeof(int),1,f);
        printf("\nAjouter un client (tapez 0 si vous voulez quitter)\n");
        scanf("%d",&stop);
    }while(stop!=0);
}

void displayfile(FILE *f,FILE *fi)
{
    CUSTOMER c ;
    int x;

    rewind(f);
    rewind(fi);

    printf("\nAffichage fichier index client\n");
    while(1)
    {
        fread(&x,sizeof(int),1,fi);
        if(feof(fi)) break;
        printf("\n%d",x);
    }
    rewind(fi);

    printf("\nAffichage fichier Client\n");

```

```

while(1)
{
    fread(&x,sizeof(int),1,fi);
    if(feof(fi)) break;
    fseek(f,x,0);
    c=readcustomer(f);
    displayCustomer(c);
    printf("\n");
}
}

void modifyCustomer(FILE *f, FILE *fi, int position) {
    CUSTOMER c;

    // déplacer à la position dans fichier index
    fseek(fi, position * sizeof(int), SEEK_SET);
    fread(&position, sizeof(int), 1, fi);

    // Déplacer à la position dans le fichier struct
    fseek(f, position, SEEK_SET);

    // Lire le client existant
    c = readcustomer(f);

    // Afficher les détails du client existant
    printf("\nDétails du client existant :\n");
    displayCustomer(c);

    // Modifier les détails du client
    addCustomer(&c);

    // Revenir à la position dans le fichier de données
    fseek(f, position, SEEK_SET);

    // Écrire les nouvelles données du client
    fwrite(&c.customerID, sizeof(int), 1, f);
    fwrite(&c.name, 50 * sizeof(char), 1, f);
    fwrite(&c.email, 50 * sizeof(char), 1, f);
    fwrite(&c.nbrOrders, sizeof(int), 1, f);

    printf("\nClient modifié avec succès.\n");
}

```

```

//***** FIN *****//

```

### 3.2. *ECommerce.cpp(code menu)*

```
/*CHEBL YOUSSEF 1 INFO D*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <locale.h> // pour appeler la fonction - setlocale -

#include "MyFunctions.cpp"

int main() {

    setlocale(LC_CTYPE, ""); //définir les paramètres régionaux

    CATEGORY *categories = NULL;
    int categoryCount = 0;

    PRODUCT *products = NULL;
    int productCount = 0;

    CUSTOMER *customers = NULL;
    int customerCount = 0;

    ORDER *orders = NULL;
    int orderCount = 0;

    CATEGORY * selectedCat = NULL ;

    RESULT ** resultArray = NULL;
    RESULT *res    = NULL ;
    int prodCount =0 ;

    CUSTOMER* c ;

    FILE *file;FILE *indexfile;

    int choix ,x, selectedCatID , nid, catID, orderID ,clientID ,categoryID, productID;

    _____
    //-----Menu-----//
    _____

    while (1)
    {
        printf("\n***** MENU PRINCIPALE *****\n");
        printf("0- Quitter le programme \n");
        printf("1- Gestion des catégories\n");
        printf("2- Gestion des produits \n");
```

```

printf("3- Gestion des clients \n");
printf("4- Gestion des commandes \n");
printf("5- Gestion du tableau RESULTAT \n");
printf("6- Gestion des fichiers \n");
printf("\n*****\n");
;

do
{
    printf("*** Votre choix : ");
    scanf("%d", &choix);
} while ( (choix < 0) || (choix > 6) );
//system("cls");
switch (choix) {
case 0:
    exit(-1);
    break;
case 1: do
{
    printf("\n***** Gestion des categories *****\n");
    printf("1- Pour ajouter une catégorie\n");
    printf("2- Pour afficher les catégories\n");
    printf("3- Pour supprimer une catégorie\n");
    printf("4- Pour afficher le nombres totale des categories \n");
    printf("5- Retour\n");
    do
    {
        printf("\n*** Votre choix :");
        scanf("%d",&x);
    }while( x<1 || x>5 );

    //system("cls");
    switch(x)
    {
        case 1:
            categories = (CATEGORY*) realloc
(categories, (categoryCount + 1) * sizeof(CATEGORY));
            if(! categories)
            { printf("\n!! Erreur d'allocation du
mémoire !!\n");
                break;
            }else
            {
                addCategory(&categories,&categoryCount);
            };
            break;
        case 2: displayAllCategories(categories, categoryCount);
            break;
        case 3:
            printf("\nEntrez l'identifiant du
categorie a supprimer:");

```

```

scanf("%d",&catID);
removeCategory(&categories,
&categoryCount, catID);

break;

case 4: printf("\nLe nombres totale des categories est %d",categoryCount);
break;

}

}while (x!=5);
break;

case 2: do
{
printf("\n***** Gestion des produits *****\n");
printf("1- Pour ajouter un produit\n");
printf("2- Pour afficher un produit\n");
printf("3- Pour supprimer un produit\n");
printf("4- Pour afficher le nombres totale des produits \n");
printf("5- Retour\n");
do
{
printf("\n*** Votre choix :");
scanf("%d",&x);
}while( x<1 || x>5 );

//system("cls");
switch(x)
{
case 1:
products = (PRODUCT *) realloc(products, (productCount + 1) *
sizeof(PRODUCT));
if(! products)
{
printf("\n!! Erreur d'allocation du mémoire !!\n");
break;
}
else
{ addProductToCategory(&products, &productCount, categories,
categoryCount);
prodCount+=1;}
break;

case 2:
printf("\nEntrez l'identifiant du produit a afficher:");
scanf("%d",&catID);
displayProductDetails(categories, categoryCount, catID);
break;

case 3:
printf("\nEntrez l'identifiant du categorie :");
scanf("%d",&categoryID);
printf("\nEntrez l'identifiant du produit a supprimer:");
scanf("%d",&catID);
removeProductFromCategory(categories, categoryCount, categoryID, catID);
break;

```

```

        case 4:
printf("\nLe nombres totale des produits est %d",prodCount);
break;
    }
        } while (x!=5);
        break;

        case 3:
            do
{
    printf("\n***** Gestion des clients *****\n") ;
    printf("1- Pour ajouter un client\n");
    printf("2- Pour afficher les clients\n");
    printf("3- Pour supprimer un client\n");
    printf("4- Pour afficher le nombres totale des clients \n");
    printf("5- Retour\n");
    do
    {
        printf("\n*** Votre choix :");
        scanf("%d",&x);
    }while( x<1 || x>5 );

    //system("cls");
    switch(x)
    {
        case 1:
            customers = (CUSTOMER *)
realloc(customers, (customerCount + 1) * sizeof(CUSTOMER));
            addCustomer(&customers, &customerCount);
            break;

        case 2:
            for (int i = 0; i < customerCount; i++)
            {
                displayCustomer(*(customers+i));
            }
            break;

        case 3:
            printf("\nEntrez l'identifiant du client a
supprimer:");

            scanf("%d",&clientID);
            removeCustomer(customers,
customerCount, clientID);

            break;

        case 4: printf("\nLe nombres totale des clients est %d",customerCount);
            break;

    }

        }while (x!=5);
        break;

        case 4:
            do
{
    printf("\n***** Gestion des commandes *****\n") ;

```



```

printf("1- Pour ajouter une commande\n");
printf("2- Pour afficher les commandes\n");
printf("3- Pour supprimer une commande\n");
printf("4- Pour afficher le nombres totale des commandes \n");
printf("5- Retour\n");
do
{
    printf("\n*** Votre choix :");
    scanf("%d",&x);
}while( x<1 || x>5 );

//system("cls");
switch(x)
{
    case 1:
        orders = (ORDER *) realloc(orders, (orderCount + 1) * sizeof(ORDER));
        createOrder(customers, categories, &orders, &orderCount, categoryCount,
customerCount) ;
                                break;
    case 2:
                                displayOrders(orders, orderCount);
                                break;
    case 3:
                                printf("\nEntrez l'identifiant de la commande a
supprimer:");
                                scanf("%d",&orderID);
                                removeOrder(orders, orderCount, orderID);
                                break;
    case 4: printf("\nLe nombre total des commandes est %d",orderCount);
                                break;

}
                                }while (x!=5);
                                break;
    case 5:
do
{
    printf("\n***** Gestion du RESULTAT *****\n");
    printf("1- Pour remplir le tableau RESULT\n");
    printf("2- Pour afficher le tableau RESULT\n");
    printf("3- Retour\n");
    do
    {
        printf("\n*** Votre choix :");
        scanf("%d",&x);
    }while( x<1 || x>3 );

//system("cls");
switch(x)
{
    case 1:
        fillResults(&resultArray, orders, orderCount);

```

```

                break;
            case 2:
                displayMostExpensiveOrder(resultArray,orderCount);
                break;

        }

        }while (x!=3);
        break;
        case 6:
do
{
    printf("\n***** Gestion des fichiers RESULTAT *****\n");
    printf("1- Pour remplir les fichiers du RESULT \n");
    printf("2- Pour afficher les fichiers du RESULT \n");
    printf("3- Retour\n");
    do
    {
        printf("\n*** Votre choix :");
        scanf("%d",&x);
    }while( x<1 || x>3 );

    //system("cls");
    switch(x)
    {
        case 1:
            createfile(&file,&indexfile);
            fillfile(file,indexfile);
            break;

        case 2:
            displayfile(file,indexfile);
            break;

    }

    }while (x!=3);
    break;
    default:
    printf("\n!! Choix invalid !!\n");
}

}

free(categories) ;
free(products) ;
free(customers) ;
free(orders) ;

free(resultArray);

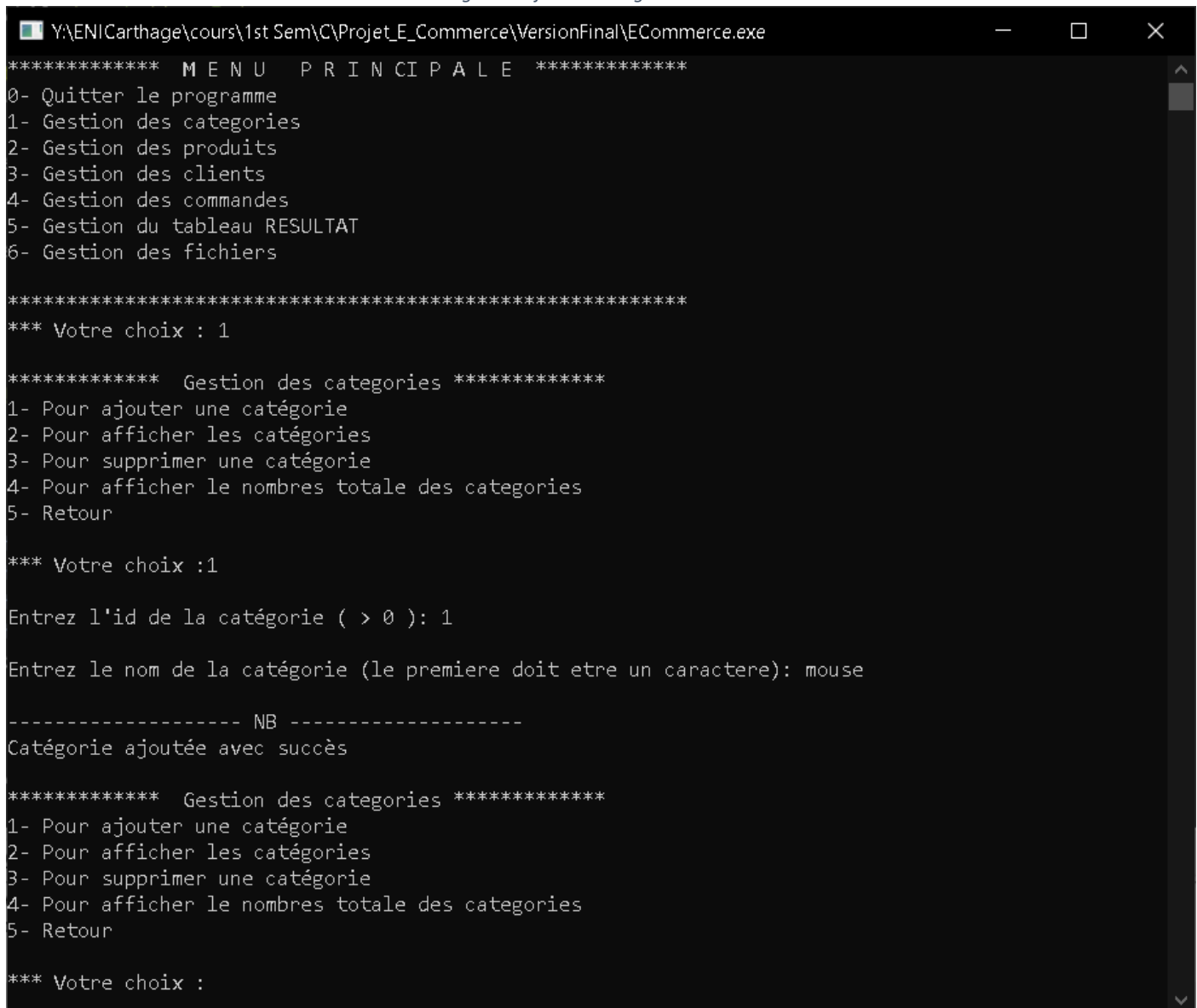
return 0;

}

```

## 4. Test exécution 1 (categorie):

Figure 1: ajout du categorie



```
Y:\ENI\Carthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe
***** M E N U   P R I N C I P A L E *****
0- Quitter le programme
1- Gestion des categories
2- Gestion des produits
3- Gestion des clients
4- Gestion des commandes
5- Gestion du tableau RESULTAT
6- Gestion des fichiers

*****
*** Votre choix : 1

***** Gestion des categories *****
1- Pour ajouter une catégorie
2- Pour afficher les catégories
3- Pour supprimer une catégorie
4- Pour afficher le nombres totale des categories
5- Retour

*** Votre choix :1

Entrez l'id de la catégorie ( > 0 ): 1

Entrez le nom de la catégorie (le premiere doit etre un caractere): mouse

----- NB -----
Catégorie ajoutée avec succès

***** Gestion des categories *****
1- Pour ajouter une catégorie
2- Pour afficher les catégories
3- Pour supprimer une catégorie
4- Pour afficher le nombres totale des categories
5- Retour

*** Votre choix :
```

```
Y:\ENICarthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe
*** Votre choix :1

Entrez l'id de la catégorie ( > 0 ): 2

Entrez le nom de la catégorie (le premiere doit etre un caractere): pc

----- NB -----
Catégorie ajoutée avec succès

***** Gestion des categories *****
1- Pour ajouter une catégorie
2- Pour afficher les catégories
3- Pour supprimer une catégorie
4- Pour afficher le nombres totale des categories
5- Retour

*** Votre choix :2

*-----* Toutes les catégories *-----*
ID de la catégorie : 1
Nom de la catégorie : mouse
Nombre de produit : 0

----- Liste des produits -----

Le montant totale du category est : 0.00
-----
ID de la catégorie : 2
Nom de la catégorie : pc
Nombre de produit : 0

----- Liste des produits -----

Le montant totale du category est : 0.00
-----
```

Figure 2: affichage des categories

```
Y:\ENICarthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe
***** Gestion des categories *****
1- Pour ajouter une catégorie
2- Pour afficher les catégories
3- Pour supprimer une catégorie
4- Pour afficher le nombres totale des categories
5- Retour

*** Votre choix :3

Entrez l'identifrier du categorie a supprimer:2

----- NB -----
Catégorie supprimée avec succès

***** Gestion des categories *****
1- Pour ajouter une catégorie
2- Pour afficher les catégories
3- Pour supprimer une catégorie
4- Pour afficher le nombres totale des categories
5- Retour

*** Votre choix :2

*-----* Toutes les catégories *-----*
ID de la catégorie : 1
Nom de la catégorie : mouse
Nombre de produit : 0

----- Liste des produits -----

Le montant totale du category est : 0.00
-----

***** Gestion des categories *****
```

Figure 3 : Supprimer une categorie

```
Y:\ENICarthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe
-----
***** Gestion des categories *****
1- Pour ajouter une catégorie
2- Pour afficher les catégories
3- Pour supprimer une catégorie
4- Pour afficher le nombres totale des categories
5- Retour

*** Votre choix :1

Entrez l'id de la catégorie ( > 0 ): 11

Entrez le nom de la catégorie (le premiere doit etre un caractere): mo

----- NB -----
Catégorie ajoutée avec succès

***** Gestion des categories *****
1- Pour ajouter une catégorie
2- Pour afficher les catégories
3- Pour supprimer une catégorie
4- Pour afficher le nombres totale des categories
5- Retour

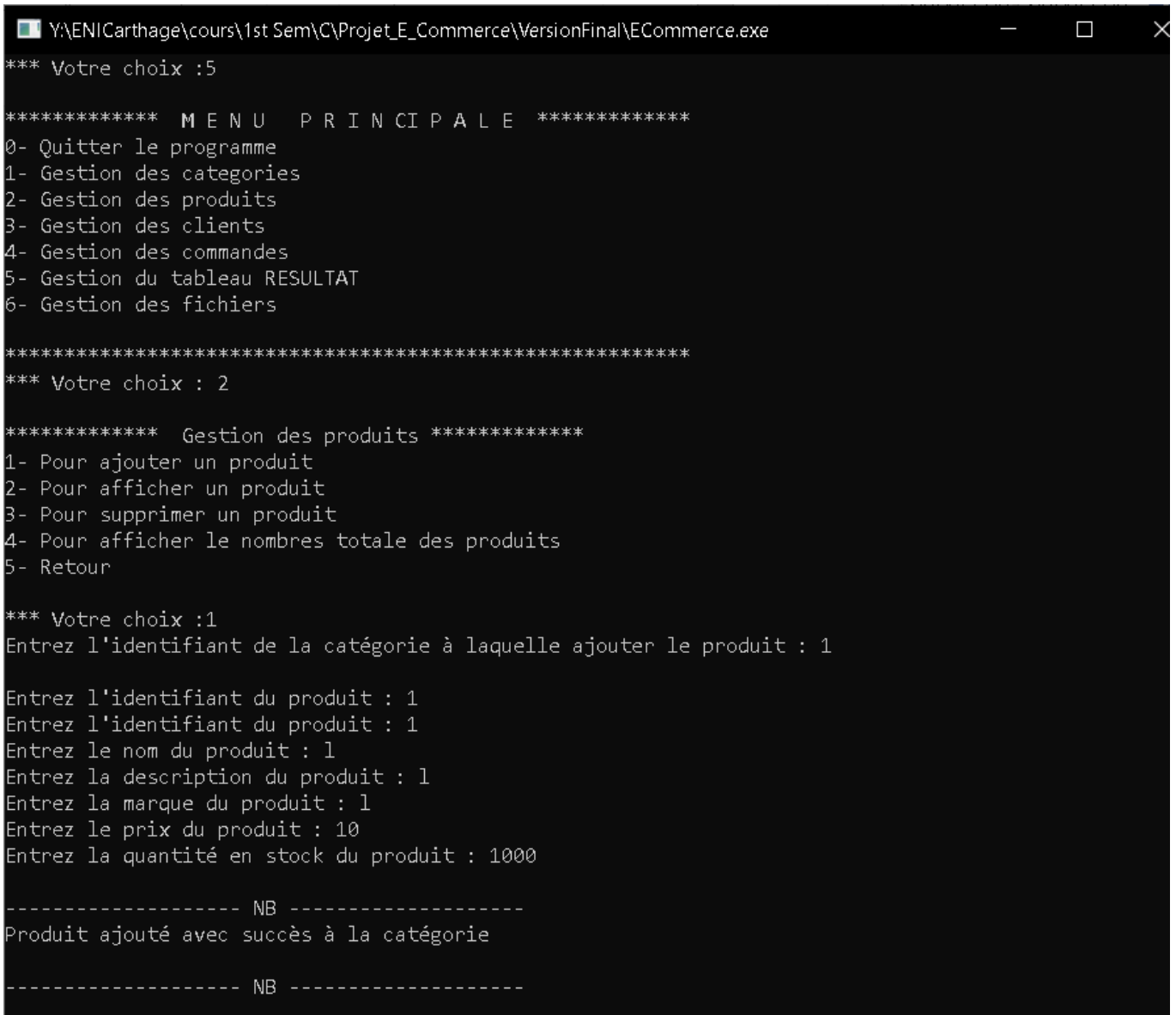
*** Votre choix :4

Le nombres totale des categories est 2
***** Gestion des categories *****
1- Pour ajouter une catégorie
2- Pour afficher les catégories
3- Pour supprimer une catégorie
4- Pour afficher le nombres totale des categories
5- Retour

*** Votre choix :5_
```

Figure 4 : afficher le nbres totales des catégories

## 5. Test exécution 2 (Produit):



```
Y:\ENI\Carthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe
*** Votre choix :5

***** M E N U   P R I N C I P A L E *****
0- Quitter le programme
1- Gestion des categories
2- Gestion des produits
3- Gestion des clients
4- Gestion des commandes
5- Gestion du tableau RESULTAT
6- Gestion des fichiers

*****
*** Votre choix : 2

***** Gestion des produits *****
1- Pour ajouter un produit
2- Pour afficher un produit
3- Pour supprimer un produit
4- Pour afficher le nombres totale des produits
5- Retour

*** Votre choix :1
Entrez l'identifiant de la catégorie à laquelle ajouter le produit : 1

Entrez l'identifiant du produit : 1
Entrez l'identifiant du produit : 1
Entrez le nom du produit : 1
Entrez la description du produit : 1
Entrez la marque du produit : 1
Entrez le prix du produit : 10
Entrez la quantité en stock du produit : 1000

----- NB -----
Produit ajouté avec succès à la catégorie

----- NB -----
```

Figure 5 : ajout d'un produit

```

***** Gestion des produits *****
1- Pour ajouter un produit
2- Pour afficher un produit
3- Pour supprimer un produit
4- Pour afficher le nombres totale des produits
5- Retour

*** Votre choix :2

Entrez l'identifiant du produit a afficher:22

Détails du Produit :
ID du Produit: 22
Nom du Produit: j
Description: h
Marque: h
Prix: 15.00
Quantité en Stock: 1000

***** Gestion des produits *****
1- Pour ajouter un produit
2- Pour afficher un produit
3- Pour supprimer un produit
4- Pour afficher le nombres totale des produits
5- Retour

*** Votre choix :4

Le nombres totale des produits est 2
***** Gestion des produits *****
1- Pour ajouter un produit
2- Pour afficher un produit
3- Pour supprimer un produit
4- Pour afficher le nombres totale des produits
5- Retour

*** Votre choix :5_

```

Figure 6 : afficher un produit et le nombres totales des produits



```
Y:\ENICarthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe
3- Pour supprimer un produit
4- Pour afficher le nombres totale des produits
5- Retour

*** Votre choix :5

***** M E N U   P R I N C I P A L E *****
0- Quitter le programme
1- Gestion des categories
2- Gestion des produits
3- Gestion des clients
4- Gestion des commandes
5- Gestion du tableau RESULTAT
6- Gestion des fichiers

*****

*** Votre choix : 2

***** Gestion des produits *****
1- Pour ajouter un produit
2- Pour afficher un produit
3- Pour supprimer un produit
4- Pour afficher le nombres totale des produits
5- Retour

*** Votre choix :3

Entrez l'identifiant du categorie :1

Entrez l'identifiant du produit a supprimer:1

----- NB -----
Produit supprimé avec succès de la catégorie

***** Gestion des produits *****
1- Pour ajouter un produit
2- Pour afficher un produit
3- Pour supprimer un produit
4- Pour afficher le nombres totale des produits
5- Retour

*** Votre choix :2

Entrez l'identifiant du produit a afficher:1

----- NB -----
Produit introuvable
```

Figure 7: supprimer produit

## 6. Test exécution 3 (client) :

```
*****
*** Votre choix : 3

***** Gestion des clients *****
1- Pour ajouter un client
2- Pour afficher les clients
3- Pour supprimer un client
4- Pour afficher le nombres totale des clients
5- Retour

*** Votre choix :1
Entrez l'identifiant du client : 1
Entrez le nom du client : sami
Entrez l'e-mail du client : sami@gmail.com

----- NB -----
Client ajouté avec succès

***** Gestion des clients *****
1- Pour ajouter un client
2- Pour afficher les clients
3- Pour supprimer un client
4- Pour afficher le nombres totale des clients
5- Retour

*** Votre choix :1
Entrez l'identifiant du client : 2
Entrez le nom du client : firs
Entrez l'e-mail du client : jhgdf

----- NB -----
Client ajouté avec succès

***** Gestion des clients *****
1- Pour ajouter un client
2- Pour afficher les clients
3- Pour supprimer un client
4- Pour afficher le nombres totale des clients
5- Retour

*** Votre choix :4

Le nombres totale des clients est 2
*****
```

Figure 8: ajout des clients et afficher le nbre total des clients

```
Y:\ENI\Carthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe

***** Gestion des clients *****
1- Pour ajouter un client
2- Pour afficher les clients
3- Pour supprimer un client
4- Pour afficher le nombres totale des clients
5- Retour

*** Votre choix :1
Entrez l'identifiant du client : 3
Entrez le nom du client : f
Entrez l'e-mail du client : v

----- NB -----
Client ajouté avec succès

***** Gestion des clients *****
1- Pour ajouter un client
2- Pour afficher les clients
3- Pour supprimer un client
4- Pour afficher le nombres totale des clients
5- Retour

*** Votre choix :2

----- Liste des Clients -----
ID du client : 1
Nom du client : s
E-mail du client : s
Nombre des commandes : 0

-----
ID du client : 2
Nom du client : a
E-mail du client : e
Nombre des commandes : 0

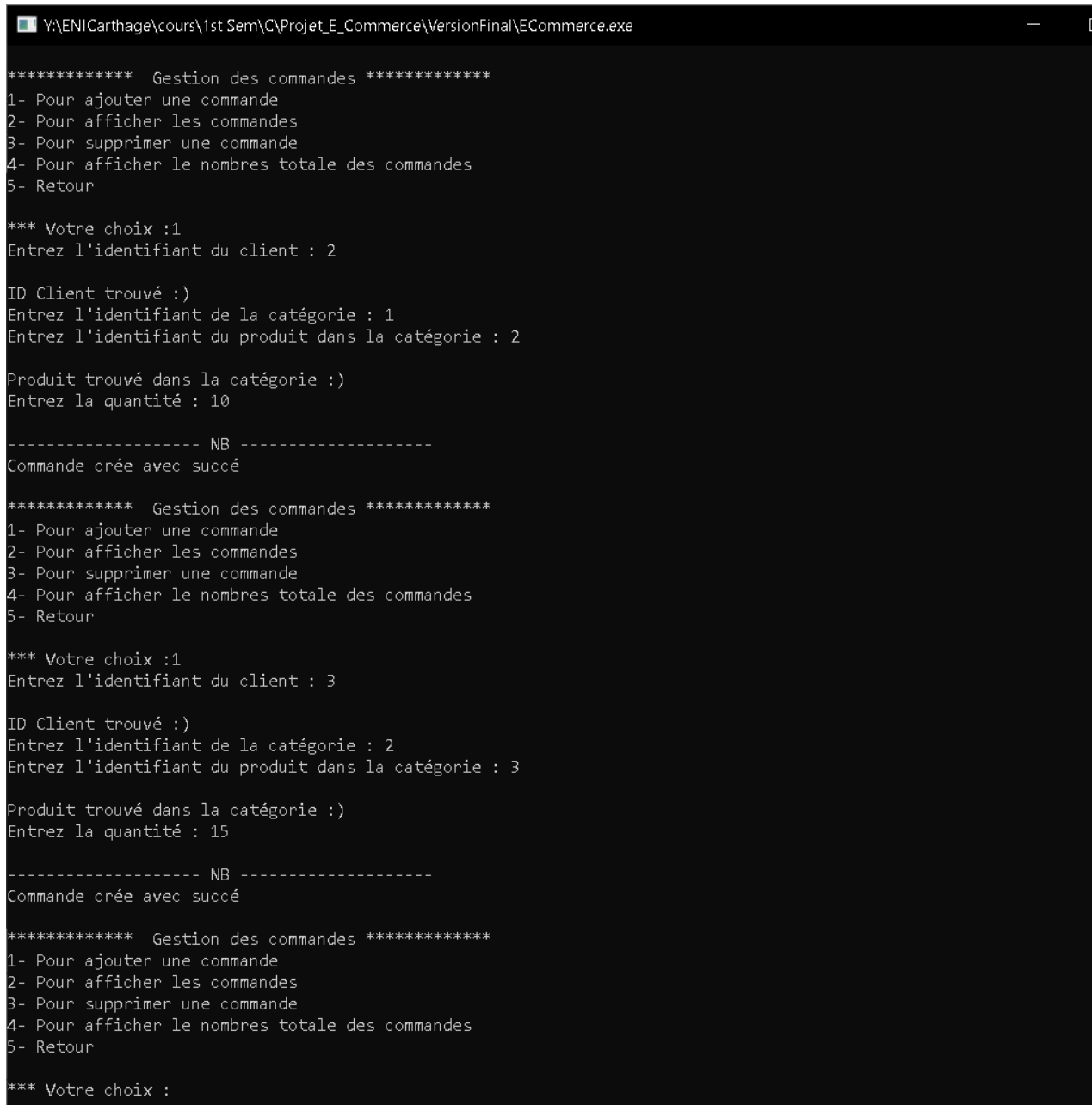
-----
ID du client : 3
Nom du client : f
E-mail du client : v
Nombre des commandes : 0

-----

***** Gestion des clients *****
1- Pour ajouter un client
2- Pour afficher les clients
3- Pour supprimer un client
```

Figure 9 : ajout et affichage des clients

## 7. Test execution 4 (commande):



```
Y:\ENI\Carthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe

***** Gestion des commandes *****
1- Pour ajouter une commande
2- Pour afficher les commandes
3- Pour supprimer une commande
4- Pour afficher le nombres totale des commandes
5- Retour

*** Votre choix :1
Entrez l'identifiant du client : 2

ID Client trouvé :)
Entrez l'identifiant de la catégorie : 1
Entrez l'identifiant du produit dans la catégorie : 2

Produit trouvé dans la catégorie :)
Entrez la quantité : 10

----- NB -----
Commande crée avec succès

***** Gestion des commandes *****
1- Pour ajouter une commande
2- Pour afficher les commandes
3- Pour supprimer une commande
4- Pour afficher le nombres totale des commandes
5- Retour

*** Votre choix :1
Entrez l'identifiant du client : 3

ID Client trouvé :)
Entrez l'identifiant de la catégorie : 2
Entrez l'identifiant du produit dans la catégorie : 3

Produit trouvé dans la catégorie :)
Entrez la quantité : 15

----- NB -----
Commande crée avec succès

***** Gestion des commandes *****
1- Pour ajouter une commande
2- Pour afficher les commandes
3- Pour supprimer une commande
4- Pour afficher le nombres totale des commandes
5- Retour

*** Votre choix :
```

Figure 10 : ajout de la commande

```
Y:\ENICarthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe
Commande crée avec succès

***** Gestion des commandes *****
1- Pour ajouter une commande
2- Pour afficher les commandes
3- Pour supprimer une commande
4- Pour afficher le nombres totale des commandes
5- Retour

*** Votre choix :2

*-----* Toutes les orders *-----*

-----
ID de la commande : 1
Client de la commande :a
Nombres des categories: 1
Montant totale de la commande : 190.00
Nombre des produits dans la commande : 10
-----

-----
ID de la commande : 2
Client de la commande :f
Nombres des categories: 1
Montant totale de la commande : 270.00
Nombre des produits dans la commande : 15
-----

-----
***** Gestion des commandes *****
1- Pour ajouter une commande
2- Pour afficher les commandes
3- Pour supprimer une commande
4- Pour afficher le nombres totale des commandes
5- Retour

*** Votre choix :4

Le nombres totale des commandes est 2
***** Gestion des commandes *****
1- Pour ajouter une commande
2- Pour afficher les commandes
3- Pour supprimer une commande
4- Pour afficher le nombres totale des commandes
5- Retour

*** Votre choix :
```

Figure 11: Affichage de la commande créée et du nbre total du commandes

## 8. Test fichiers :

```
Y:\ENICarthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe

***** MENU PRINCIPALE *****
0- Quitter le programme
1- Gestion des categories
2- Gestion des produits
3- Gestion des clients
4- Gestion des commandes
5- Gestion du tableau RESULTAT
6- Gestion des fichiers

*****

*** Votre choix : 6

***** Gestion des fichiers *****
1- Pour remplir les fichiers
2- Pour afficher les fichiers
3- Pour modifier les fichiers
4- Retour

*** Votre choix :1
Entrez l'identifiant du client : 1
Entrez le nom du client : d
Entrez l'e-mail du client : f
Entrez le nombre d'order du client : 1

Ajouter un client (tapez 0 si vous voulez quitter)
1
Entrez l'identifiant du client : 2
Entrez le nom du client : s
Entrez l'e-mail du client : d
Entrez le nombre d'order du client : 4

Ajouter un client (tapez 0 si vous voulez quitter)
0

***** Gestion des fichiers *****
1- Pour remplir les fichiers
2- Pour afficher les fichiers
3- Pour modifier les fichiers
4- Retour

*** Votre choix :
```

Figure 12 : remplir fichier client et fichier index

```
Y:\ENICarthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECo...
***** Gestion des fichiers *****
1- Pour remplir les fichiers
2- Pour afficher les fichiers
3- Pour modifier les fichiers
4- Retour

*** Votre choix :2

Affichage fichier index client

0
108
Affichage fichier Client
ID du client : 1
Nom du client : d
E-mail du client : f
Nombre des commandes : 1

-----

ID du client : 2
Nom du client : s
E-mail du client : d
Nombre des commandes : 4

-----

***** Gestion des fichiers *****
1- Pour remplir les fichiers
```

Figure 13: affichage fichier client et fichier index

```
Y:\ENICarthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECoo...
***** Gestion des fichiers *****
1- Pour remplir les fichiers
2- Pour afficher les fichiers
3- Pour modifier les fichiers
4- Retour

*** Votre choix :3

Entrez la position du modification :0

Details du client existant :
ID du client : 1
Nom du client : d
E-mail du client : f
Nombre des commandes : 1

-----
Entrez l'identifiant du client : 11
Entrez le nom du client : dd
Entrez l'e-mail du client : ff
Entrez le nombre d'ordre du client : 11

Client modifié avec succès.

***** Gestion des fichiers *****
1- Pour remplir les fichiers
2- Pour afficher les fichiers
3- Pour modifier les fichiers
4- Retour

*** Votre choix :2

Affichage fichier index client

0
108
Affichage fichier Client
ID du client : 11
Nom du client : dd
E-mail du client : ff
Nombre des commandes : 11

-----

ID du client : 2
Nom du client : s
E-mail du client : d
Nombre des commandes : 4
```

Figure 14 : modification d'un client d'une position donnée et l'affichage



## 9. Test tableau dynamique des clients :

```
Y:\ENI\Carthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe

*****

*** Votre choix : 5

***** Gestion du RESULTAT *****
1- Pour remplir le tableau RESULT
2- Pour afficher le tableau RESULT
3- Retour

*** Votre choix :1
remplissage avec succès :)
--- Liste des Commandes ---
Order ID: 1,
Total Amount: 200.00
Order ID: 2,
Total Amount: 400.00
Order ID: 3,
Total Amount: 900.00

***** Gestion du RESULTAT *****
1- Pour remplir le tableau RESULT
2- Pour afficher le tableau RESULT
3- Retour

*** Votre choix :2

--- Commande la plus chère ---
Order ID: 3,
Total Amount: 900.00

***** Gestion du RESULTAT *****
1- Pour remplir le tableau RESULT
2- Pour afficher le tableau RESULT
3- Retour

*** Votre choix :
```

Figure 15 : remplissage et affichage du tableau Resultat

## 10. Menu du programme (Execution):

```
Y:\ENI\Carthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe

***** MENU PRINCIPALE *****
0- Quitter le programme
1- Gestion des categories
2- Gestion des produits
3- Gestion des clients
4- Gestion des commandes
5- Gestion du tableau RESULTAT
6- Gestion des fichiers

*****

*** Votre choix : 1

***** Gestion des categories *****
1- Pour ajouter une catégorie
2- Pour afficher les catégories
3- Pour supprimer une catégorie
4- Pour afficher le nombres totale des categories
5- Retour

*** Votre choix :5

***** MENU PRINCIPALE *****
0- Quitter le programme
1- Gestion des categories
2- Gestion des produits
3- Gestion des clients
4- Gestion des commandes
5- Gestion du tableau RESULTAT
6- Gestion des fichiers

*****

*** Votre choix : 2

***** Gestion des produits *****
1- Pour ajouter un produit
2- Pour afficher un produit
3- Pour supprimer un produit
4- Pour afficher le nombres totale des produits
5- Retour

*** Votre choix :5

***** MENU PRINCIPALE *****
0- Quitter le programme
1- Gestion des categories
2- Gestion des produits
```

```
Y:\ENICarthage\cours\1st Sem\C\Projet_E_Commerce\VersionFinal\ECommerce.exe
***** Gestion des clients *****
1- Pour ajouter un client
2- Pour afficher les clients
3- Pour supprimer un client
4- Pour afficher le nombres totale des clients
5- Retour

*** Votre choix :5

***** MENU PRINCIPALE *****
0- Quitter le programme
1- Gestion des categories
2- Gestion des produits
3- Gestion des clients
4- Gestion des commandes
5- Gestion du tableau RESULTAT
6- Gestion des fichiers

*****

*** Votre choix : 4

***** Gestion des commandes *****
1- Pour ajouter une commande
2- Pour afficher les commandes
3- Pour supprimer une commande
4- Pour afficher le nombres totale des commandes
5- Retour

*** Votre choix :5

***** MENU PRINCIPALE *****
0- Quitter le programme
1- Gestion des categories
2- Gestion des produits
3- Gestion des clients
4- Gestion des commandes
5- Gestion du tableau RESULTAT
6- Gestion des fichiers

*****

*** Votre choix : 6

***** Gestion des fichiers *****
1- Pour remplir les fichiers
2- Pour afficher les fichiers
3- Pour modifier les fichiers
4- Retour
```