DOCUMENTATION OFFICIEL DU LOGICIEL

"BricoConstruction"

Documentation partie GESTION ARTICLE :

LES FONCTIONS VÉRIFIENT LA VALIDITÉ DE CHAQUE VALEUR POUR CHAQUE ENTRÉE !

1 - FONCTION GLOBALE :

La fonction globale est la fonction initiale du logiciel. Elle s'affiche lors du lancement de la partie gestion magasin. Elle nous permet de choisir dans quelle catégorie de l'application nous souhaitons utiliser et qui eux-mêmes redirige vers d'autres fonctions utiles à leur spécificité.

En appuyant sur '0', l'utilisateur est redirigé vers la partie Responsable des articles. Dans ce menu il y a plusieurs options possibles.

En tant que Responsable, l'utilisateur à accès à tout un panel de fonctions qu'il peut choisir en utilisant les touches **"1","2","3","4", et "9"**

En appuyant sur '1', l'utilisateur est redirigé vers la partie client des articles. Dans ce menu il y a plusieurs options possibles.

En tant que client, l'utilisateur à accès à tout un panel de fonctions qu'il peut choisir en utilisant les touches **"1","2","3" et "9".**

Nous détaillerons les fonctions catégorie par catégorie dans la suite de cette documentation.

2 - FONCTION AJOUT ARTICLE (RESPONSABLE) :

```
void AjoutArticle(void)
{
	int ref;
	float poid,volume,prix;
	FILE *flot;
	flot = fopen("articles.txt","a");
	if (flot == NULL)
		{
			printf("Problème avec la création du fichier\n");
			exit(1);
		}
	printf("Saisir la référence de l'article (-1 pour arrêter): ");
	scanf("%d",&ref);
	while (ref < -1)
	{
		printf("Erreur ; Le numéro doit être positif (ou -1 pour arrêter); retapez : ");
		scanf("%d",&ref);
	}
	while (ref >= 0)
	{
		printf("Saisir le poid(kg): ");
		scanf("%f",&poid);
		while (poid < 0)
		{
			printf("Erreur ; Le poid doit être positif ; retapez : ");
			scanf("%f",&poid);
		}

		printf("Saisir le volume: ");
		scanf("%f",&volume);
		while (volume < 0)
		{
			printf("Erreur ; Le volume doit être positif ; retapez : ");
			scanf("%f",&volume);
		}

		printf("Saisir le prix: ");
		scanf("%f",&prix);
		while (prix < 0)
		{
			printf("Erreur ; Le prix doit être positif ; retapez : ");
			scanf("%f",&prix);
		}

		fprintf(flot,"%d\t%.3f\t%.2f\t%.2f\n",ref,poid,volume,prix);
		printf("Saisir la référence de l'article (-1 pour arrêter) : ");
		scanf("%d",&ref);
		while (ref < -1)
		{
			printf("Erreur ; Le numéro doit être positif (ou -1 pour arrêter); retapez : ");
			scanf("%d",&ref);
		}
	}
	fclose(flot);
	printf("Fin de l'ajout.\n");
}
```

La fonction ajout article est la première fonction que nous rencontrons dans la partie responsable, elle va nous permettre d'ajouter un article dans la liste des articles disponible dans le magasin.

Elle va ouvrir le fichier et demander à l'utilisateur de saisir les informations à propos de l'article (la référence, le poids, le volume et le prix).

Une fois entrée, la fonction fermera le fichier et renverra à la fonction globale responsable.

3 - FONCTION AFFICHAGE FICHIER (RESPONSABLE) :

```
int TableArticle(int Tref[], float Tpoid[],float Tvolume[],float Tprix[])
{
	int ref,i=0;
	float poid,volume,prix;

	FILE *flot;
	flot = fopen("articles.txt","r");
	if (flot == NULL)
		{
			printf("Problème avec la lecture du fichier\n");
			exit(1);
		}

	fscanf(flot,"%d %f %f %f",&ref,&poid,&volume,&prix);
	while(feof(flot) == 0)
	{
		Tref[i] = ref;
		Tpoid[i] = poid;
		Tvolume[i] = volume;
		Tprix[i] = prix;
		i++;
		fscanf(flot,"%d %f %f %f",&ref,&poid,&volume,&prix);
	}
	fclose(flot);
	return i;
}

void AffichTable(int Tref[], float Tpoid[],float Tvolume[],float Tprix[],int n)
{
	int i;
	printf("référence\tpoids(kg)\tvolume(l)\tprix Unitaire\n");
	for (i=0;i < n;i++)
	{
		printf("%d\t\t%.2f\t\t\t%.3f\t\t\t%.2f\n",Tref[i],Tpoid[i],Tvolume[i],Tprix[i]);
	}
}
```

La fonction d'affichage est la deuxième fonction présente dans la partie responsable, elle va nous permettre d'afficher l'entièreté du fichier contenant les informations à propos des articles en chargeant le fichier de données dans des tableaux grâce à une fonction annexe.

Puis, elle va afficher ce qu'il a précédemment chargé sur le terminal avant de renvoyer à la fonction globale responsable .

4 - FONCTION AFFICHAGE ARTICLE (RESPONSABLE) :

La fonction d'affichage d'article est la troisième fonction présente dans la partie responsable, elle fonctionne de la même manière que la fonction affichage fichier grâce au tableau. Mais contrairement à cette dernière elle va demander à l'utilisateur l'article à saisir et vas seulement afficher la ligne contenue dans le tableau qui correspond au numéro de l'article recherché,

avant de renvoyer à la fonction globale responsable

```c
void AffichArticle(int Tref[], float Tpoid[],float Tvolume[],float Tprix[],int n)
{
	int refR,pos;
	printf("Afficher quel article ? ");
	scanf("%d",&refR);
	pos = posRef(Tref,refR,n);
	if (pos == -1)
	{
		printf("L'article n'existe pas dans la base de donnée.");
	}
	else
	{
		printf("%d\t\t%.2f\t\t%.3f\t\t%.2f\n",Tref[pos],Tpoid[pos],Tvolume[pos],Tprix[pos]);
	}
}
```

5 - FONCTION MODIFICATION ARTICLE (RESPONSABLE) :

La fonction de modification est la quatrième fonction de la catégorie responsable et va permettre de modifier un article déjà existant.

Les tableaux sont envoyés dans cette fonction lors de son lancement, l'utilisateur est ensuite invité à entrer la valeur de la référence de l'article à modifier, dès lors qu'il est trouvé dans les tableaux, l'usager devra rentrer les nouvelles valeurs de poids, de volume et de prix qui écraseront les précédentes.

La fonction renvoie ensuite à la fonction globale responsable.

```c
void ModifArticle(int Tref[], float Tpoid[],float Tvolume[],float Tprix[],int n)
{
	int refR;
	printf("Saisir la référence de l'article a modifier: ");
	scanf("%d",&refR);
	while (refR < -1)
	{
		printf("Erreur ; Le numéro doit être positif; retapez : ");
		scanf("%d",&refR);
	}
	int i;
	for (i=0;i < n;i++)
	{
	if (Tref[i] == refR)
		{
		printf("Saisir le nouveau poid(kg): ");
		scanf("%f",&Tpoid[i]);
		while (Tpoid[i] < 0)
		{
			printf("Erreur ; Le poid doit être positif ; retapez : ");
			scanf("%f",&Tpoid[i]);
		}

		printf("Saisir le nouveau volume: ");
		scanf("%f",&Tvolume[i]);
		while (Tvolume[i] < 0)
		{
			printf("Erreur ; Le volume doit être positif ; retapez : ");
			scanf("%f",&Tvolume[i]);
		}

		printf("Saisir le nouveau prix: ");
		scanf("%f",&Tprix[i]);
		while (Tprix[i] < 0)
		{
			printf("Erreur ; Le prix doit être positif ; retapez : ");
			scanf("%f",&Tprix[i]);
		}
		}
	}
}
```

6 - FONCTION SUPPRESSION ARTICLE (RESPONSABLE) :

```c
int SupprArticle(int Tref[], float Tpoid[],float Tvolume[],float Tprix[],int n)
{
	int refR,pos;
	printf("Saisir la référence de l'article a supprimer: ");
	scanf("%d",&refR);
	while (refR < -1)
	{
		printf("Erreur ; Le numéro doit être positif; retapez : ");
		scanf("%d",&refR);
	}
	pos = posRef(Tref,refR,n);
	if (pos == -1)
	{
		printf("L'article n'existe pas dans la base de donnée.");
		return n;
	}
	else
	{
		int i;
		for (i=pos;i<=n;i++)
		{
			Tref[i] = Tref[i+1];
			Tpoid[i] = Tpoid[i+1];
			Tvolume[i] = Tvolume[i+1];
			Tprix[i] = Tprix[i+1];
		}
		return n-=1;
	}
}
```

La fonction suppression est la cinquième et dernière fonction de l'outil responsable. De la même manière que la modification.

Les tableaux sont donnés à la fonction et l'utilisateur est invité à entrer la référence de l'article à supprimer et grâce à un système de décalage à gauche, l'article à supprimer passera en dernière position et la taille logique **sera diminuée de 1** la supprimant ainsi.

Nous retournons ensuite à la fonction globale responsable.

7 - FONCTION AJOUT ARTICLE AU PANIER (CLIENT) :

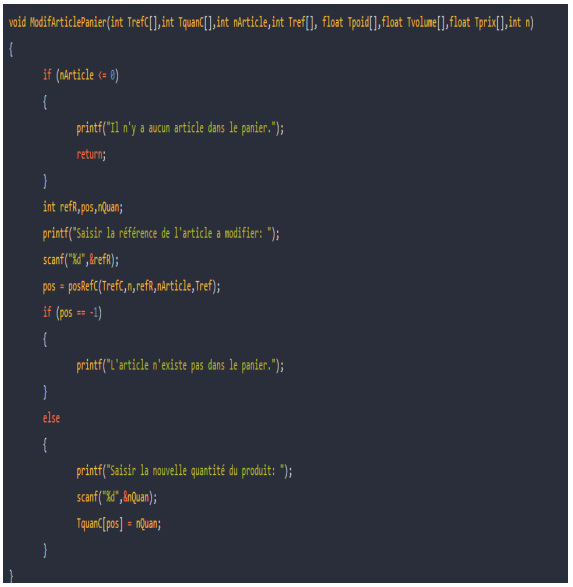
```
int ajoutPanier(int TrefC[],int TquanC[],int nArticle,int Tref[], int n)
{
	int refP,code,quantite,pos;
	printf("Saisir la référence du produit à ajouter au panier: ");
	scanf("%d",&refP);

	pos = posRefC(TrefC,n,refP,nArticle,Tref);
	if (pos != -1)
	{
		printf("L'article existe déja dans le panier.");
		return nArticle;
	}

	code = posRef(Tref,refP,n);
	while (code == -1)
	{
		printf("Erreur ; Cette référence n'existe pas dans la base de donnée; retapez : ");
		scanf("%d",&refP);
		code = posRef(Tref,refP,n);
	}

	printf("Saisir la quantité souhaité pour le produit: ");
	scanf("%d",&quantite);
	while (quantite < 1)
	{
		printf("Erreur ; La quantité peut seulement être positive; retapez : ");
		scanf("%d",&quantite);
	}
	printf("\nRéférence : %d\n",refP);
	printf("Quantité : %d\n",quantite);
	TrefC[nArticle] = refP;
	TquanC[nArticle] = quantite;
	return nArticle +=1;
}
```

La fonction ajout article au panier est la première fonction (hors saisie d'informations voiture) de la partie cliente. Lors de son appel on lui donne les tableaux composant le panier, l'utilisateur est invité à entrer la référence de l'article qu'il souhaite.

La fonction ajout article va ensuite appeler la fonction posrec pour déterminer la position de l'article dans le tableau pour ajouter la quantité de ce dernier (demander à l'utilisateur) dans le panier.

La fonction renvoie ensuite à la fonction globale cliente.

8 - FONCTION MODIFIER ARTICLE AU PANIER (CLIENT) :



```
void ModifArticlePanier(int TrefC[],int TquanC[],int nArticle,int Tref[], float Tpoid[],float Tvolume[],float Tprix[],int n)
{
	if (nArticle <= 0)
	{
		printf("Il n'y a aucun article dans le panier.");
		return;
	}
	int refR,pos,nQuan;
	printf("Saisir la référence de l'article a modifier: ");
	scanf("%d",&refR);
	pos = posRefC(TrefC,n,refR,nArticle,Tref);
	if (pos == -1)
	{
		printf("L'article n'existe pas dans le panier.");
	}
	else
	{
		printf("Saisir la nouvelle quantité du produit: ");
		scanf("%d",&nQuan);
		TquanC[pos] = nQuan;
	}
}
```

La fonction ajout modifier article du panier est la deuxième fonction de la partie cliente. Lors de son appel on lui donne les tableaux. Elle va nous permettre de modifier la quantité d'un article.

Elle fonctionne de la même manière que la fonction d'ajout, en appelant la fonction posrec afin de retourner la position de l'article dont on souhaite modifier la quantité dans le tableau.

Une fois la quantité changée, la fonction renvoie à la fonction globale cliente.

9 - FONCTION SUPPRESSION ARTICLE AU PANIER (CLIENT) :

La fonction suppression carte est la troisième option sélectionnable dans la fonction de la partie cliente, elle va nous permettre de supprimer un article du panier dans les tableaux.

Grâce à un système de décalage à gauche, nous allons écraser la valeur que nous souhaitons supprimer en partant de la valeur de la position de la référence que l'utilisateur du programme va donner. (Toujours grâce à posrec)

Une fois la suppression bien effectuée, le programme renvoie à la fonction globale cliente.

```
int SupprArticlePanier(int TrefC[],int TquanC[],int nArticle,int Tref[], float Tpoid[],float Tvolume[],float Tprix[],int n)
{
	if (nArticle <= 0)
	{
		printf("Il n'y a aucun article dans le panier.");
		return nArticle;
	}
	int refR,pos;
	printf("Saisir la référence de l'article a supprimer: ");
	scanf("%d",&refR);
	pos = posRefC(TrefC,n,refR,nArticle,Tref);
	if (pos == -1)
	{
		printf("L'article n'existe pas dans le panier.");
	}
	else
	{
		TrefC[pos] = TrefC[pos+1];
		TquanC[pos] = TquanC[pos+1];
		return nArticle - 1;
	}
}
```

10 - FONCTION AFFICHAGE ARTICLE PANIER (CLIENT) :

La fonction d'affichage des articles du panier est la dernière (et la plus longue) fonction du programme client.

Lorsque l'utilisateur appelle cette fonction, elle va d'elle-même récupérer le contenu du panier des tableaux dans des variables pour effectuer les calculs nécessaires (tel que les charges pour déterminer si elles sont autorisées ou non).

Après avoir afficher tous les contenus, elle renvoie à la fonction globale cliente.

```
void affichPanier(int TrefC[],int TquanC[],int nArticle,int Tref[], float Tpoid[],float Tvolume[],float Tprix[],float chargeV, float volV,int n)
{
	printf("\nRéf\tQté\tPoids\tVol\tPrixU\tPoidsTot\tVolTot\tPrixTot\tCagnotte\n");
	int i, pos,cagnotte;
	float prixtot,volR,chargeR;
	int cfinal=0;
	float prixfinal=0,volfinal=0,chargefinal=0;
	for (i=0;i <nArticle;i++)
	{
		pos = posRef(Tref,TrefC[i],n);
		prixtot=Tprix[pos]*TquanC[i];
		cagnotte=prixtot*0.1;
		prixfinal+=prixtot;
		cfinal+=cagnotte;
		volfinal+=Tvolume[pos]*TquanC[i];
		chargefinal+=Tpoid[pos]*TquanC[i];
		printf("%d\t%d\t%.1f\t%.1f\t%.1f\t%.1f\t%.1f\t%d\n",TrefC[i],TquanC[i],Tpoid[pos],Tvolume[pos],Tprix[pos],Tpoid[pos]*TquanC[i],Tvolume[p
os]*TquanC[i],prixtot,cagnotte);
	}
	printf("\n\t\t\t\t\tPrix total à payer:\t%.0f euros\n",prixfinal);
	printf("\t\t\t\t\tCagnotte totale:\t%d euros\n",cfinal);

	volR = volV - volfinal;
	printf("Volume utilisé:  %.0f litres\n",volfinal);
	printf("Volume restant: ");
	if (volR < 0)
	{
		printf("Attention dépassement de la charge autorisée de ");
		volR *= -1;
	}
	printf("%.0f litres\n",volR);


	chargeR = chargeV - chargefinal;
	printf("Charge Actuelle: %.0f kg\n",chargeFinal);
	printf("Charge restante: ");
	if (chargeR < 0)
	{
		printf("Attention dépassement de la charge autorisée de ");
		chargeR *= -1;
	}
	printf("%.0f kg\n",chargeR);
}
```

EXÉCUTIONS ET *JEUX D'ESSAIS* DE LA GESTION ARTICLE (NON EXHAUSTIF) :

```
Que voulez vous faire ? (afficher[0]/afficher article[1]/ajouter[2]/modifier[3]/supprimer[4]/quitter[9]) 0
référence       poids(kg)       volume(l)       prix Unitaire
101             2.25            2.500           15.75
233             10.00           15.000          32.00
272             6.50            10.000          29.99
66              33.00           11.000          22.00
345             30.00           29.000          10.00
```

Affichage des articles.

```
Que faire ? (afficher[0]/afficher article[1]/ajouter[2]/modifier[3]/supprimer[4]/quitter[9]) 1
Afficher quel article ? 101
101             2.25            2.500           15.75
```

Affichage d'un article.

```
Que voulez vous faire ? (afficher[0]/afficher article[1]/ajouter[2]/modifier[3]/supprimer[4]/quitter[9]) 1
Afficher quel article ? 7438987432
L'article n'existe pas dans la base de donnée.
```

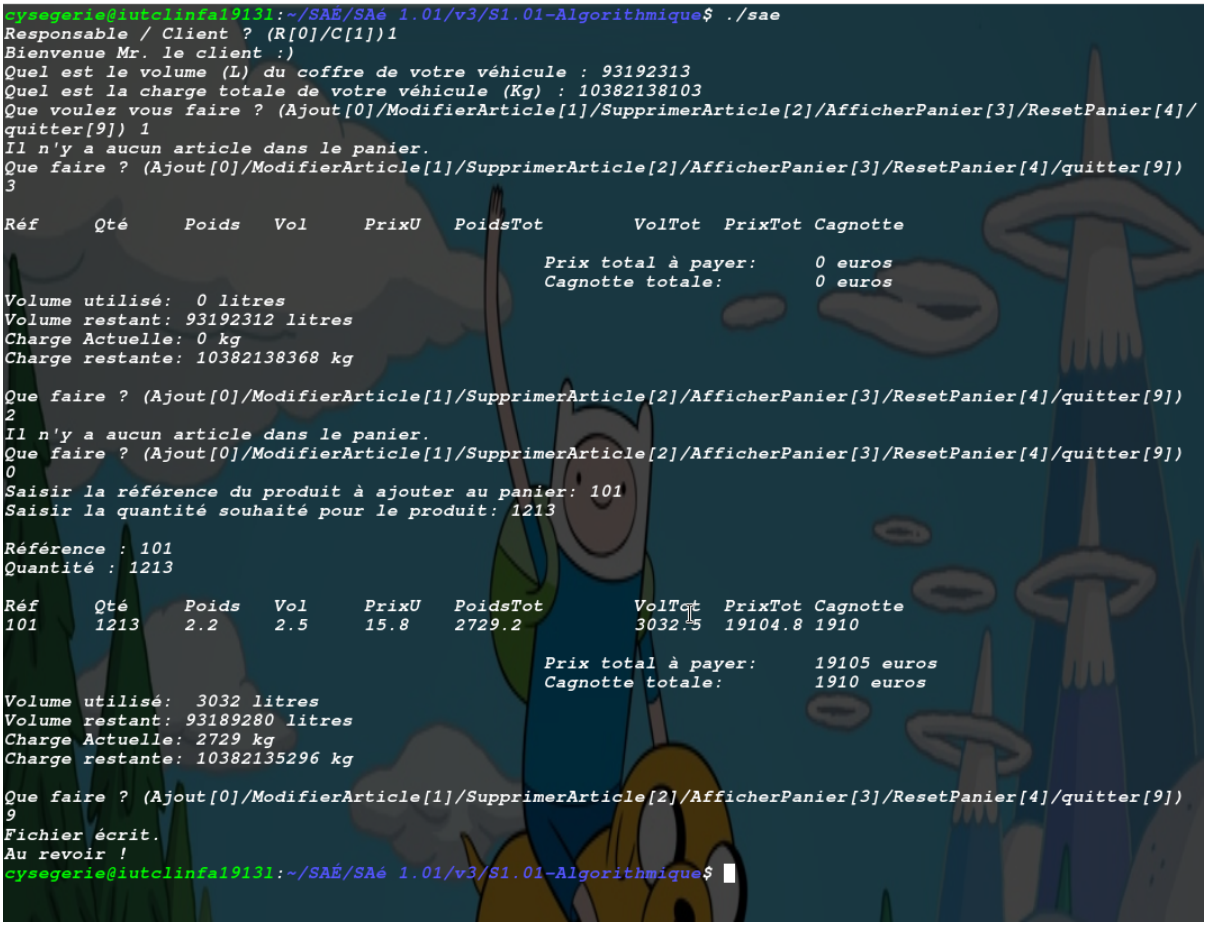
Affichage d'un article dans le cas où l'article n'existe pas

```
Que faire ? (afficher[0]/afficher article[1]/ajouter[2]/modifier[3]/supprimer[4]/quitter[9]) 2
Saisir la référence de l'article (-1 pour arrêter): 564
Saisir le poid(kg): 38
Saisir le volume: 20
Saisir le prix: 17.9
Saisir la référence de l'article (-1 pour arrêter): -1
Fin de l'ajout.

Que faire ? (afficher[0]/afficher article[1]/ajouter[2]/modifier[3]/supprimer[4]/quitter[9]) 0
référence       poids(kg)       volume(l)       prix Unitaire
101             2.25            2.500           15.75
233             10.00           15.000          32.00
272             6.50            10.000          29.99
66              33.00           11.000          22.00
345             30.00           29.000          10.00
564             38.00           20.000          17.90

Que faire ? (afficher[0]/afficher article[1]/ajouter[2]/modifier[3]/supprimer[4]/quitter[9])
```

Ajout d'un article puis affichage.

```
Que voulez vous faire ? (afficher[0]/afficher article[1]/ajouter[2]/modifier[3]/supprimer[4]/quitter[9]) 2
Saisir la référence de l'article (-1 pour arrêter): -3872938
Erreur ; Le numéro doit être positif (ou -1 pour arrêter); retapez : 564
Saisir le poid(kg): -37
Erreur ; Le poid doit être positif ; retapez : -38
Erreur ; Le poid doit être positif ; retapez : 38
Saisir le volume: -20
Erreur ; Le volume doit être positif ; retapez : -20
Erreur ; Le volume doit être positif ; retapez : 20
Saisir le prix: -18
Erreur ; Le prix doit être positif ; retapez : -18
Erreur ; Le prix doit être positif ; retapez : 17.9
Saisir la référence de l'article (-1 pour arrêter): -1
Fin de l'ajout.

Que faire ? (afficher[0]/afficher article[1]/ajouter[2]/modifier[3]/supprimer[4]/quitter[9])
```

Ajout erroné d'articles (valeurs négatives).



Ajout d'articles en grande quantité.

Documentation partie GESTION FICHIER CLIENT :

LA FONCTION VERIFIE LA VALIDITE DE CHAQUE VALEUR POUR CHAQUE ENTRÉE !

A CHAQUE ERREUR, UN CODE ERREUR EST RENVOYÉ DANS GLOBALE !

1 - FONCTION GLOBALE :

```
void modifclientglobal (void)
{
        int n,choix,tmax=100,coderreur;
        int tabNC [100]={0}, tabsuspension [100]={0};
        float tabCA6 [100]={0};

        coderreur=creationFichier();

        n=tableauxcharge(tabNC,tabCA6,tabsuspension,tmax);
        if (n==-1 || n==-2)
        {
                printf("tableau non chargée, erreur\n");
                return;
        }
        printf("Bienvenue dans l'application de modulation client .\nque souhaitez vous faire aujourd'hui ?\n");

        while(choix!=9)
        {
                printf("\n");
                printf("Appuyez sur '1' pour ajouter un client,\nsur '2' pour changer l'état de suspension d'une carte client,\nsur '3' pour supprimer un clien
,\nsur '4' pour afficher le fichier client,\nsur '5' pour chercher un seul client dans le fichier,\nOu sur '9' pour enregistrer et sortir du programme !\n");
                scanf("%d",&choix);
                while (choix!=1 && choix!=2 && choix!=3 && choix!=4  && choix!=5 && choix!=9)
                {
                        printf("Entrée incorrect, veuillez ressaisir.\n");
                        scanf("%d",&choix);
                }
                if (choix==1)
                {
                        coderreur=ajoutClient(tabNC,tabCA6,tabsuspension,&n,tmax);
                }

                if (choix==2)
                {
                        coderreur=suspensioncarte(tabNC,tabCA6,tabsuspension,&n,tmax);
                }

                if (choix==3)
                {
                        coderreur=supressionclient(tabNC,tabCA6,tabsuspension,&n,tmax);
                }

                if (choix==4)
                {
                        coderreur=affichageall(tabNC,tabCA6,tabsuspension,&n);
                }
                if (choix==5)
                {
                        coderreur=affichagesolo(tabNC,tabCA6,tabsuspension,&n);
                }
        }
    coderreur=enregistrement(tabNC,tabCA6,tabsuspension,&n);
        return;
```

La fonction globale est la fonction qui s'affiche lors du démarrage de la partie gestion fichier client. C'est un menu qui va nous permettre de choisir l'action à effectuer dans le fichier client en choisissant parmi 6 options qui représentent 6 autres fonctions présentes dans le code.

(Les différentes entrées possibles sont : "1","2","3","4","5" et "9", dans le cas où l'entrée ne correspondrait pas la fonction vous demandera de retaper.)

A chaque fin d'opérations nous revenons à chaque fois dans cette fonction avec un rappel des entrées possibles pour choisir les options, à son lancement elle initialise les tableaux et met les valeurs du fichier dedans.

2 - FONCTION AJOUT CLIENT :

```
int ajoutClient (int tabNC[],float tabCA6[],int tabsuspension[],int *tlogique,int tmax)

{
        int numeroclient,suspension,compteur=0;
        float cagnotte;

        if (*tlogique+1>tmax)
        {
                printf("tableau trop petit\n");
                return -1;
        }

        printf("ajout d'un client au programme de fidélité...\nquel est le numéro de ce client ?\n'999' pour quitter.\n");
        scanf("%d",&numeroclient);
        if (numeroclient==999)
        {
                printf("retour au menu global\n");
                return 0;
        }
        while (numeroclient<0)
        {
                printf("le numéro du client ne peut être négatif , réessayez\n'999' pour quitter.\n");
            scanf("%d",&numeroclient);
                if (numeroclient==999)
                {
                        printf("retour au menu global\n");
                        return 0;
                }
        }

        while (compteur<*tlogique)
        {
                if (tabNC[compteur]==numeroclient)
                {
                        while(tabNC[compteur]==numeroclient)
                        {
                                printf("le numéro client existe déjà ,veuillez ressaisir\n");
                            scanf("%d",&numeroclient);
                        }
                }
                compteur=compteur+1;
        }

        *tlogique=*tlogique+1;
        tabNC[*tlogique]=numeroclient;
        tabCA6[*tlogique]=0.00;
```

La fonction ajout client est la première option sélectionnable dans la fonction globale, elle va nous permettre de rajouter un client à notre fichier.

Elle fonctionne par le biais de tableaux : une fois appelé, l'utilisateur rentre le numéro de client qu'il veut ajouter au fichier dans le terminal, la fonction s'occupera de vérifier en parcourant les tableaux si le numéro existe déjà.

Nous serons ensuite ramenés à la fonction globale.

Après que la fonction est augmentée la taille logique grâce au pointeur “tlogique”.

3 - FONCTION SUSPENSION CARTE :

La fonction suspension carte est la deuxième option sélectionnable dans la fonction globale, elle va nous permettre de changer l’état de suspension d’une carte (0 ou 1).

Elle fonctionne elle aussi grâce aux tableaux : la fonction globale en l’appelant va lui envoyer les tableaux et leur taille logique.

L’utilisateur est invité à taper le numéro de la carte client qu’il souhaite suspendre / dé-suspendre.

La fonction parcourra les tableaux pour trouver l’emplacement de la carte avec les valeurs de sa cagnotte et de l’état de suspension associé (si le numéro client recherché n’est pas trouvé, la fonction retourne une erreur et repart dans la fonction globale)

```
int suspensioncarte (int tabNC[],float tabCAG[],int tabsuspension[],int *tlogique,int tmax)

{

    int nouvellecarte,suspension=2,compteur=0,compteurCAG,recherche;

printf("quel carte souhaitez vous suspendre ou désuspendre (entrez numéro client)\n'999' pour quitter.\n");
scanf("%d",&recherche);
    if (recherche==999)
        {
            printf("retour au menu global\n");
            return 0;
        }
while (recherche<0)
{
    printf("numéro non valide , veuillez réessayez!\n");
    scanf("%d",&recherche);
        if (recherche==999)
        {
            printf("retour au menu global\n");
            return 0;
        }
}

for (compteur=0;compteur<=*tlogique;compteur++)
{
    if (tabNC[compteur]==recherche)
    {
        compteurCAG=compteur;
        printf("l'état de la carte de ce client est %d\n",tabsuspension[compteur]);
    printf("modifiez l'état (0 pour non suspendu) (1 pour suspendu)\n");
    scanf("%d",&suspension);
    while (suspension!=1 && suspension !=0)
    {
        printf("état de suspension non valide , retapez.\n");
            scanf("%d",&suspension);
    }
    if (tabsuspension[compteur]==suspension)
    {
        printf("état de suspension est déja de %d\n",suspension);
        return -4;
    }
    tabsuspension[compteur]=suspension;
    }
}

if (suspension!=0 && suspension!=1)
{
    printf("le numéro client n'existe pas\n");
    return -7;
}
```

L’état de suspension de la carte est rappelé à l’utilisateur qui est invité à entrer la nouvelle valeur de la suspension.

Si le nouvel état est égal à l’ancienne, un message s’affiche, rien n’est modifié et la fonction renvoie à la fonction globale.

```
if (tabsuspension[compteurCAG]==1)
{
        if (*tlogique+1>tmax)
        {
            printf("tableau trop petit\n");
            return -1;
        }

    *tlogique=*tlogique+1;

        printf("création d'une nouvelle carte pour le client %d\n",tabNC[compteurCAG]);
        printf("numéro de la nouvelle carte ?\n");
        scanf("%d",&nouvellecarte);
        while (nouvellecarte<0)
        {
            printf("ne peut être négatif\n");
            scanf("%d",&nouvellecarte);
        }
        for (compteur=0;compteur<=*tlogique;compteur++)

            while (tabNC[compteur]==nouvellecarte)
            {
                printf("le client existe déja, ressaisissez\n");
                scanf("%d",&nouvellecarte);
                while(nouvellecarte<0)
                {
                    printf("le numéro ne peut être négatif , réésayez.\n");
                    scanf("%d",&nouvellecarte);
                }
            }

        tabNC[*tlogique]=nouvellecarte;
        tabCAG[*tlogique]=tabCAG[compteurCAG];
        tabsuspension[*tlogique]=0;

        printf("état de suspension modifié et nouvelle carte crée !\n");
        return 0;
    }
    printf("état de suspension modifié !\n");
    return 0;
```

DANS LE CAS OU SUSPENSION = 0 ;
Changement de l’état dans le tableau.

DANS LE CAS OU SUSPENSION = 1 ;
Changement de l’état dans le tableau, l’utilisateur est invité à créer une nouvelle carte pour le client, la fonction vérifie si la place est suffisante dans les tableaux grâce à la taille max “tmax” initialement donné par la fonction globale.

La fonction retourne à la fonction globale.

4 - FONCTION SUPPRESSION CARTE :

```c
int supressionclient (int tabNC[],float tabCAG[],int tabsuspension[],int *tlogique,int tmax)
{

	int compteur,recherche,nt=1;
	printf("\n");
	printf("quel est le numéro du client que vous souhaitez supprimer ?\n'999' pour quitter.\n");
	scanf("%d",&recherche);
	if (recherche==999)
	{
		printf("retour au menu global\n");
		return 0;
	}
	while(recherche<0)
	{
		printf("ne peut être négatif\n'999' pour quitter.\n");
		scanf("%d",&recherche);
		if (recherche==999)
		{
			printf("retour au menu global\n");
			return 0;

		}
	}

	for (compteur=0;compteur<=*tlogique;compteur++)
		if (tabNC[compteur]==recherche)
		{
			while (compteur<=*tlogique)
			{
				tabNC[compteur]=tabNC[compteur+1];
				tabCAG[compteur]=tabCAG[compteur+1];
				tabsuspension[compteur]=tabsuspension[compteur+1];
				compteur=compteur+1;
				nt=0;

			}
			*tlogique=*tlogique-1;
		}

	if (nt!=0)
	{
		printf("le numéro client n'existe pas\n");
		return -4;
	}
	printf("client supprimé.\n");
	return 0;
```

La fonction suppression carte est la troisième option sélectionnable dans la fonction globale, elle va nous permettre de supprimer une carte.

La fonction globale passe les tableaux et leur taille logique à la fonction.

De la même manière que la fonction précédente on va demander à l'utilisateur quel client veut-il supprimer. La fonction parcourra ensuite les tableaux jusqu'à cette valeur, la décalera à la toute fin du fichier (en décalant toutes les valeurs à gauche), et, diminue la taille logique de 1, supprimant ainsi cette valeur.

Si la valeur n'existe pas la fonction renvoie à la fonction globale.

Retour à la fonction globale une fois la suppression bien effectuée.

5 - FONCTION AFFICHAGE COMPLET :

La fonction affichage complet est la quatrième option de la fonction globale, elle permet d'afficher dans le terminal le contenu des tableaux.

La fonction une fois appelée va grâce aux tableaux donné par la fonction global écrire les fonctions sous forme de colonne. Une fois les tableaux entièrement parcourus, le fichier se ferme et renvoie à la fonction globale.

```c
int affichageall (int tabNC[],float tabCAG[],int tabsuspension[],int *tlogique)
{
	int compteur;
	printf("\n");
	for (compteur=0;compteur<=*tlogique;compteur++)
		printf("%d\t%.2f\t%d\n",tabNC[compteur],tabCAG[compteur],tabsuspension[compteur]);
	return 0;
}
```

6 - FONCTION AFFICHAGE SPÉCIFIQUE :

La fonction affichage spécifique est la cinquième et dernière option de la fonction globale, elle permet d'afficher dans le terminal les informations d'un client recherché.

Elle fonctionne exactement de la même façon que la fonction affichage complet qui écrivait les tableaux ligne par ligne, sauf qu'en demandant à l'utilisateur d'entrer un numéro à rechercher. On va seulement afficher la ligne ou le numéro client est égale au numéro client recherché.

```
int affichagesolo (int tabNC[],float tabCAG[],int tabsuspension[],int *tlogique)
{
	int rechercher,compteur=0;
	float cagnotte;

	printf("\n");
	printf("quel client recherchez vous ?\n'999' pour quitter.\n");
	scanf("%d",&rechercher);
	if (rechercher==999)
	{
		printf("retour au menu global\n");
		return 0;
	}
	while(rechercher<0)
	{
		printf("ne peut être négatif\n'999' pour quitter.\n");
		scanf("%d",&rechercher);
		if (rechercher==999)
		{
			printf("retour au menu global\n");
			return 0;
		}

	}
	printf("\n");

	while (compteur<*tlogique)
	{
		if (tabNC[compteur]==rechercher)
		{
			printf("%d\t%.2f\t%d\n",tabNC[compteur],tabCAG[compteur],tabsuspension[compteur]);
			return 0;
		}
		compteur=compteur+1;
	}
	printf("ce client n'existe pas\n");
	return 0;
}
```

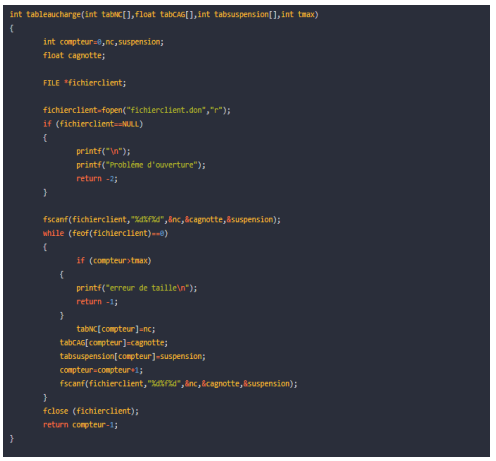
Si elle ne trouve pas la valeur pour numéro client, elle renvoie une erreur. (Ajouté post screen)
Et retour à la fonction globale.

7 - FONCTION CHARGEMENT TABLEAUX :

La fonction de chargement tableaux est vital pour l'utilisation de notre programme, elle va permettre de parcourir le fichier donné pour stocker chacune des lignes dans les tableaux.

Tableaux dont nous allons modifier le contenu pour éditer le fichier.

Renvoie de la valeur et retour à la fonction globale.



```
int tableaucharge(int tabNC[],float tabCAG[],int tabsuspension[],int tmax)
{
	int compteur=0,nc,suspension;
	float cagnotte;

	FILE *fichierclient;

	fichierclient=fopen("Fichierclient.don","r");
	if (fichierclient==NULL)
	{
		printf("\n");
		printf("Problème d'ouverture");
		return -2;
	}

	fscanf(fichierclient,"%d%f%d",&nc,&cagnotte,&suspension);
	while (feof(fichierclient)==0)
	{
		if (compteur>tmax)
	    {
		printf("erreur de taille\n");
		return -1;
	    }
		tabNC[compteur]=nc;
	    tabCAG[compteur]=cagnotte;
	    tabsuspension[compteur]=suspension;
	    compteur=compteur+1;
	    fscanf(fichierclient,"%d%f%d",&nc,&cagnotte,&suspension);
	}
	fclose (fichierclient);
	return compteur-1;
}
```

8 - FONCTION CREATION FICHIER / VÉRIFICATION FICHIER :

La fonction création / vérification fichier et la deuxième et dernière fonction non visible du programme. Elle permet à chaque lancement du programme gestion fichier client de vérifier la présence du fichier et si non le créer.

```
int creationfichier(void)
{
	FILE *flot;
	flot=fopen("fichierclient.don","a");
	fclose(flot);
	return 0;

}
```

On va demander à la fonction de rajouter du vide au fichier, s'il existe cela ne va rien modifier, s'il n’existe pas cela va le créer.

Retour à la fonction globale.

EXÉCUTIONS ET *JEUX D'ESSAIS* DE LA GESTION DU FICHIER CLIENT :

```
Bienvenue dans l'application de modulation client .
que souhaitez vous faire aujourd'hui ?

Appuyez sur '1' pour ajouter un client,
sur '2' pour changer l'état de suspension d'une carte client,
sur '3' pour supprimer un client ,
sur '4' pour afficher le fichier client,
sur '5' pour chercher un seul client dans le fichier,
Ou sur '9' pour enregistrer et sortir du programme !
1
ajout d'un client au programme de fidélité...
quel est le numéro de ce client ?
'999' pour quitter.
142
client ajouté au programme de fidélité !
```

Exécution correcte de la première fonction. (Ajout client)

```
Appuyez sur '1' pour ajouter un client,
sur '2' pour changer l'état de suspension d'une carte client,
sur '3' pour supprimer un client ,
sur '4' pour afficher le fichier client,
sur '5' pour chercher un seul client dans le fichier,
Ou sur '9' pour enregistrer et sortir du programme !
2
quel carte souhaitez vous suspendre ou désuspendre (entrez numéro client)
'999' pour quitter.
142
l'état de la carte de ce client est 0
modifiez l'état (0 pour non suspendu) (1 pour suspendu)
1
création d'une nouvelle carte pour le client 142
numéro de la nouvelle carte ?
13133
état de suspension modifié et nouvelle carte crée !
```

Exécution correcte de la seconde fonction. (Suspension carte)

```
Appuyez sur '1' pour ajouter un client,
sur '2' pour changer l'état de suspension d'une carte client,
sur '3' pour supprimer un client ,
sur '4' pour afficher le fichier client,
sur '5' pour chercher un seul client dans le fichier,
Ou sur '9' pour enregistrer et sortir du programme !
3

quel est le numéro du client que vous souhaitez supprimer ?
'999' pour quitter.
13133
client supprimé.
```

Exécution correcte de la troisième fonction. (Suppression carte)

```
Appuyez sur '1' pour ajouter un client,
sur '2' pour changer l'état de suspension d'une carte client,
sur '3' pour supprimer un client ,
sur '4' pour afficher le fichier client,
sur '5' pour chercher un seul client dans le fichier,
Ou sur '9' pour enregistrer et sortir du programme !
4

344     0.00    0
12      0.00    0
42      0.00    0
414     0.00    0
142     0.00    1
```

Exécution correcte de la quatrième fonction. (Affichage fichier client)

```
Appuyez sur '1' pour ajouter un client,
sur '2' pour changer l'état de suspension d'une carte client,
sur '3' pour supprimer un client ,
sur '4' pour afficher le fichier client,
sur '5' pour chercher un seul client dans le fichier,
Ou sur '9' pour enregistrer et sortir du programme !
5

quel client recherchez vous ?
'999' pour quitter.
42

42      0.00    0
```

Exécution correcte de la cinquième fonction. (Affichage client spécifique)

```
Appuyez sur '1' pour ajouter un client,
sur '2' pour changer l'état de suspension d'une carte client,
sur '3' pour supprimer un client ,
sur '4' pour afficher le fichier client,
sur '5' pour chercher un seul client dans le fichier,
Ou sur '9' pour enregistrer et sortir du programme !
1
ajout d'un client au programme de fidélité...
quel est le numéro de ce client ?
'999' pour quitter.
-4
le numéro du client ne peut être négatif , réessayez
'999' pour quitter.
344
le numéro client existe déja ,veuillez ressaissir
94
client ajouté au programme de fidélité !
```

Exécution erronée de la première fonction. (Valeur incorrect)

```
Appuyez sur '1' pour ajouter un client,
sur '2' pour changer l'état de suspension d'une carte client,
sur '3' pour supprimer un client ,
sur '4' pour afficher le fichier client,
sur '5' pour chercher un seul client dans le fichier,
Ou sur '9' pour enregistrer et sortir du programme !
2
quel carte souhaitez vous suspendre ou désuspendre (entrez numéro client)
'999' pour quitter.
141244
le numéro client n'existe pas
```

Exécution erronée de la seconde fonction. (Valeur incorrect client)

```
Appuyez sur '1' pour ajouter un client,
sur '2' pour changer l'état de suspension d'une carte client,
sur '3' pour supprimer un client ,
sur '4' pour afficher le fichier client,
sur '5' pour chercher un seul client dans le fichier,
Ou sur '9' pour enregistrer et sortir du programme !
2
quel carte souhaitez vous suspendre ou désuspendre (entrez numéro client)
'999' pour quitter.
12
l'état de la carte de ce client est 0
modifiez l'état (0 pour non suspendu) (1 pour suspendu)
0
état de suspension est déja de 0
```

Exécution erronée de la seconde fonction. (Valeur incorrect suspension)

```
Appuyez sur '1' pour ajouter un client,
sur '2' pour changer l'état de suspension d'une carte client,
sur '3' pour supprimer un client ,
sur '4' pour afficher le fichier client,
sur '5' pour chercher un seul client dans le fichier,
Ou sur '9' pour enregistrer et sortir du programme !
2
quel carte souhaitez vous suspendre ou désuspendre (entrez numéro client)
'999' pour quitter.
-3
numéro non valide , veuillez réessayez
12
l'état de la carte de ce client est 0
modifiez l'état (0 pour non suspendu) (1 pour suspendu)
53
état de suspension non valide , retapez.
1
création d'une nouvelle carte pour le client 12
numéro de la nouvelle carte ?
12
le client existe déja, ressaisissez
42
le client existe déja, ressaisissez
424
état de suspension modifié et nouvelle carte crée !
```

Exécution erronée de la deuxième fonctions (toutes les erreurs de valeurs possibles)

```
Appuyez sur '1' pour ajouter un client,
sur '2' pour changer l'état de suspension d'une carte client,
sur '3' pour supprimer un client ,
sur '4' pour afficher le fichier client,
sur '5' pour chercher un seul client dans le fichier,
Ou sur '9' pour enregistrer et sortir du programme !
3

quel est le numéro du client que vous souhaitez supprimer ?
'999' pour quitter.
-42
ne peut étre négatif
'999' pour quitter.
1313
le numéro client n'existe pas
```

Exécution erronée de la troisième fonction. (Toutes les valeurs incorrect)

```
Appuyez sur '1' pour ajouter un client,
sur '2' pour changer l'état de suspension d'une carte client,
sur '3' pour supprimer un client ,
sur '4' pour afficher le fichier client,
sur '5' pour chercher un seul client dans le fichier,
Ou sur '9' pour enregistrer et sortir du programme !
5

quel client recherchez vous ?
'999' pour quitter.
31313

ce client n'existe pas
```

Exécution erronée de la cinquième fonction. (Valeur incorrect)

TRACES D'EXÉCUTIONS (NON EXHAUSTIVE)

void affichPanier(int TrefC[],int TquanC[],int nArticle,int Tref[], float Tpoid[],float Tvolume[],float Tprix[],float chargeV, float volV,int n, float cagnotteC)

Lignes \ Variables	TrefC	TquanC	nArticle	Tref	Tpoid	Tvolume	Tprix	chargeV	volV	cagnotteC	n	i	pos	cagnotte	prixtot	volR	chargeR	cfinal	prixfinal	volfinal	chargefinal	Commentaire
1	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	x	x	x	x	x	x	x	x	x	x	Initialisation des variables de la fonction
3	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	x	x	x	x	x	x	x	x	x	x	Affichage : Réf Qté Poids Vol PrixU PoidsTot VolTot PrixTot
4	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	?	?	x	x	x	x	x	x	x	x	
5	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	?	?	?	?	?	?	x	x	x	x	Initialisation des variables dans la fonction
6	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	?	?	?	?	?	?	0	x	x	x	
7	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	?	?	?	?	?	?	0	0	0	0	
8	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	?	?	?	?	?	0	0	0	0	On rentre dans la boucle for qui permettra d'Afficher les articles.
10	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	?	?	?	?	0	0	0	0	Affectation à pos de la position de la référence
11	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	?	53,25	?	?	0	0	0	0	Calcule du prix total d'un article (Prix * Quantité)
12	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	?	?	0	0	0	0	Calcule de la cagnotte (prix total d'un article *0.1)
13	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	?	?	0	53,25	0	0	Ajouts du prix total d'un article au prix final à payer
14	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	?	?	0	53,25	0	0	Ajouts de la cagnotte de l'article à la cagnotte final
15	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	?	?	0	53,25	7.5	0	Ajouts du volume de l'article * quantité au volume final
16	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	?	?	0	53,25	7.5	6.75	Ajouts du poid de l'article * quantité au poid final
17	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	?	?	0	53,25	7.5	6.75	Affichage : 101 3 2.25 2.5 15.75 6.75 7.5 53.90 5.325
18	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	?	?	0	53,25	7.5	6.75	Fin de la boucle
19	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	?	?	0	53,25	7.5	6.75	Affichage : Prix total à payer: 53.25 euros
20	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	?	?	0	53,25	7.5	6.75	Affichage : Cagnotte totale: 5.32 euros
22	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	542,5	?	0	53,25	7.5	6.75	Calcule du volume restant (Volume de la voiture - volume final)
23	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	542,5	?	0	53,25	7.5	6.75	Affichage : Volume utilisé: 7.5 litres
24	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	542,5	?	0	53,25	7.5	6.75	Affichage : Volume restant:
25	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	542,5	?	0	53,25	7.5	6.75	On ne rentre pas dans la boucle car le vol restant est supérieur à 0.
30	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	542,5	?	0	53,25	7.5	6.75	Suite de l'affichage volume restant : 542.5
32	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	542,5	543,25	0	53,25	7.5	6.75	Calcule du volume restant (Volume max de la voiture - volume final)
33	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	542,5	543,25	0	53,25	7.5	6.75	Affichage : Charge Actuelle: 6.75 kg
34	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	542,5	543,25	0	53,25	7.5	6.75	Affichage : Charge restante:
35	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	542,5	543,25	0	53,25	7.5	6.75	On ne rentre pas dans la boucle car la charge restante est supérieur à 0.
40	[101]	[3]	1	[101,233,272,66,345]	[2.25,10,6.5,33,30]	[2.5,15,10,11,29]	[15.75,32,29.99,22,10]	410,0	450,0	30,4	?	0	0	5,325	53,25	542,5	543,25	0	53,25	7.5	6.75	Suite de l'affichage charge restante : 543.25

Trace d'exécution de la fonction affichPanier.

void AjoutsArticle(void)

Lignes \ Variables	ref	poid	volume	prix	flot	Commentaire
1	Init	X	X	X	NULL	
2	?	Init	Init	Init	NULL	
3	?	?	?	?	fopen (append)	
5-8	?	?	?	?	If	
11	Scanf 101	?	?	?	OPEN	Saisi de la référence de l'article
12	While 101 < 0	?	?	?	OPEN	Vérification si ref est positif
15	Scanf (NULL)	?	?	?	OPEN	└ Saisie de la référence de l'article
17	While 101>=0	?	?	?	OPEN	Vérification si ref est supérieur ou égal à 0
20	101	Scanf 2.5	?	?	OPEN	Saisie du poid
21	101	While 2.5	?	?	OPEN	Vérification si le poid est inférieur à 0
24	101	Scanf (NU	?	?	OPEN	└ Sasi du poid
28	101	2.5	Scanf 2	?	OPEN	Saisi du volume
29	101	2.5	While 2<0	?	OPEN	Vérification si le volume est inférieur à 0
32	101	2.5	Scanf (NU	?	OPEN	└ Saisi du volume
36	101	2.5	2	Scanf 5.25	OPEN	Saisi du prix
37	101	2.5	2	While 5.25 < 0	OPEN	Vérification si le prix est inférieur à 0
38	101	2.5	2	Scanf (NULL)	OPEN	Saisi du prix
43	101	2.5	2	5.25	fprintf	Ajouts des variables ref/poids/volume/prix dans
49	Scanf	?	?	?	OPEN	Saisi référence (relance la boucle)
52					fclose	Fermeture du fichier si reférence = -1

Information			
	Integer		
	Float		
	FILE		
	Ajouts		
	Modification		Commentaire
	Supression		
	Check If/While		Utilisation

Trace d'exécution de la fonction AjoutsArticle.