



REPUBLIQUE DU BENIN

\*\*\*

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE  
SCIENTIFIQUE (MESRS)

\*\*\*

UNIVERSITÉ D'ABOMEY-CALAVI (UAC)

\*\*\*

ÉCOLE POLYTECHNIQUE D'ABOMEY-CALAVI (EPAC)

Unité d'Enseignement : Langage C++

Filière : Génie Informatique et Télécoms 3<sup>e</sup> année

## RAPPORT DE PROJET : GESTION D'UN REPERTOIRE TELEPHONIQUE

Réalisé par :

AHOUCANDJINO Bill D. M.

Sous la direction de :

M. TOLODE Gérard

Année académique: 2020-2021

# TABLE DES MATIERES

<i>ENONCE.....</i>	<i>3</i>
<i>ANALYSE DU PROJET.....</i>	<i>3</i>
<i>STRUCTURATION DU PROGRAMME.....</i>	<i>3</i>
<i>RECAPITULATIF DES FONCTIONS.....</i>	<i>4</i>
<i>DEROULEMENT ET CAPTURE A L'EXECUTION.....</i>	<i>5</i>
<i>FONCTIONS C++ ET BIBLIOTHEQUES UTILISEES.....</i>	<i>11</i>
<i>DIFFICULTES RENCONTREES ET SOLUTIONS APPORTEES.....</i>	<i>13</i>
<i>A SAVOIR AVANT L'EXECUTION DU PROGRAMME.....</i>	<i>17</i>
<i>OBJECTIVATION.....</i>	<i>17</i>
<i>TABLE DES ILLUSTRATIONS .....</i>	<i>18</i>

## ENONCE

Ecrire un programme C++ qui dispose d'un menu permettant de gérer le répertoire téléphonique d'un utilisateur.

## ANALYSE DU PROJET

Il s'agit d'implémenter un exemple de répertoire téléphonique semblable à ceux de nos téléphones. Nous nous baserons sur les méthodes standards des dits répertoires à savoir : lister tous les contacts du répertoire, ajouter, rechercher, modifier et/ou supprimer un contact. Nous disposerons d'un fichier.txt qui servira de base de données (les contacts du répertoire) auquel nous aurons accès tout au long de l'une quelconque des opérations parmi celles disponibles que l'utilisateur voudra effectuer.

## STRUCTURATION DU PROGRAMME

Notre programme est structuré en trois (03) grandes parties : le *main.cpp* (fonction principale), le *fichierSource.cpp* contenant la définition des fonctions secondaires éventuelles et le *fichier.h* (fichier entête) contenant le prototype des fonctions secondaires. Cette structuration du programme le rend moins lourd en apparence et facilite la lecture et la compréhension pour ceux qui voudront s'y intéresser.

Analysons pas à pas le contenu du programme :

### 1. Le main.cpp

Il s'agit de notre fonction principale. Nous l'avons nommée *Repertoire.cpp*. Elle ne contient que très peu de lignes (ce qui est l'objectif de notre structuration). Dans ce fichier, nous avons déclaré les variables qui seront considérées comme globales, soit plus simplement, utilisables par les autres fonctions. Elle porte en entête, en plus des bibliothèques standards, une inclusion du *fichier.h* appelé ici *FonctionsMenu.h*.

A l'exécution, *Repertoire.cpp* fait appel à une fonction que nous nommons ici *Menu()* qui se chargera de rediriger l'utilisateur vers les fonctions correspondantes aux opérations qu'il aurait choisi effectuer. Ensuite, à la fin de l'exécution de l'opération choisie, elle demande à ce dernier s'il voudrait effectuer d'autres opérations. Si oui, elle le redirige vers *Menu()*. Une quelconque autre réponse autre

qu'un « OUI » est considéré comme un « NON » : le programme est arrêté donc *sortie du répertoire*.

## 2. Le fichierSource.cpp

Nommé **Menu.cpp**, il contient la définition de toutes les fonctions secondaires. Nous avons, à l'étude de ce projet, cinq (05) fonctions secondaires primordiales (à la demande du chargé du cours) et de quelques fonctions secondaires optionnelles :

### ***RECAPITULATIF DES FONCTIONS***

Fonctions	Prototypes	Notions utilisées
<b>Fonctions secondaires primordiales :</b> elles représentent les opérations incontournables que nous pouvons effectuer dans un répertoire téléphonique.	bool Menu () ;	Structure conditionnelle Switch
	bool AjouterContact () ;	Gestion de fichier.txt (notion de flux), structures
	void RechercherContact (string indice) ;	Gestion de fichier.txt, tableau dynamique
	bool ModifierContact (std ::string indice);	Gestion de fichier.txt (notion de flux), tableau dynamique
	bool SupprimerContact (std::string);	Gestion de fichier.txt (notion de flux), tableau dynamique
	void AfficherTousMesContacts();	Gestion de fichier.txt (notion de flux)
	void ReorganiseRepertoire() ;	Tableaux dynamiques
	void ParOrdreAlphabetique() ;	Tableaux dynamiques
	int GerErreur(int ) ;	Vérification d'une saisie
	bool OptionsAvancees() ;	Structure conditionnelle Switch
	bool ParametrerSecurite() ;	Gestion de fichier.pdf (notion de flux), tableau dynamique

	bool SupprimerLesDoublons() ;	Gestion de fichier.txt (notion de flux), tableau dynamique
	void ViderLeRepertoire() ;	Gestion de fichier.txt (notion de flux), tableau dynamique
<b>Fonctions secondaires optionnelles</b>	bool MotDePasse() ;	Gestion de fichier.pdf (notion de flux), tableau dynamique
	bool VerifierIdentite() ;	Gestion de fichier.txt (notion de flux), tableau dynamique

### 3. Le fichier.h

Nous l'avons nommé FonctionsMenu.h. Il contient le prototype de toutes les fonctions de Menu.cpp et la bibliothèque string du fait qu'en arguments de certaines fonctions, nous avons des variables plus complexes comme les *string*.

## ***DEROULEMENT ET CAPTURE A L'EXECUTION***

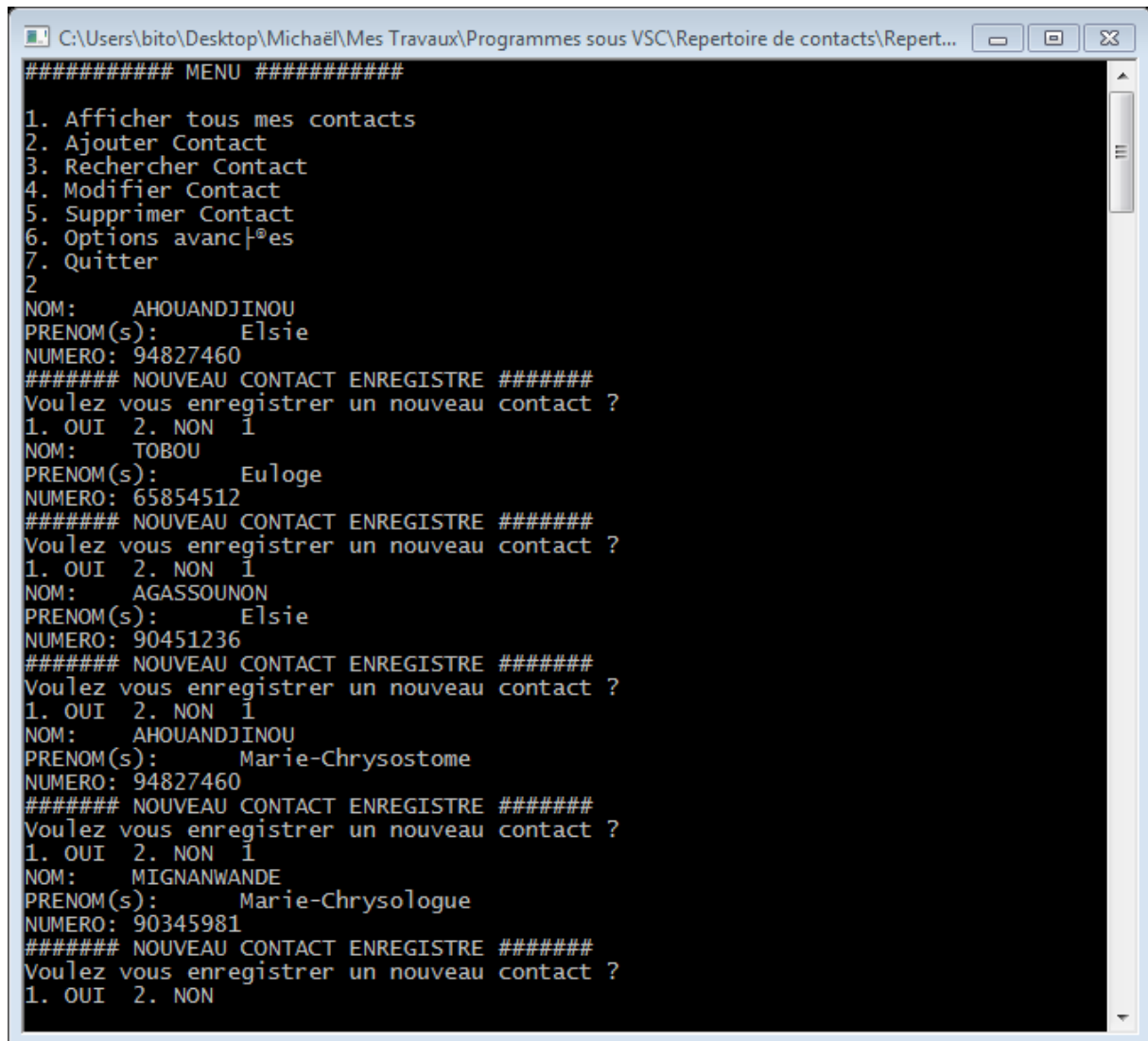
L'utilisateur lance le programme. Le *main.cpp* fait appel à la fonction *Menu()* du fichier *Menu.cpp*. Cette fonction propose six (07) choix à l'utilisateur : afficher tous ses contacts, ajouter un contact, rechercher un contact, modifier un contact, supprimer un contact, des options avancées (que nous verrons plus loin) et quitter le répertoire.

- Quitter le répertoire

Il ne s'agit pas ici d'une fonction. Il est simplement demandé à l'utilisateur de confirmer sa sortie du répertoire. A sa réponse positive, l'exécution du programme s'estompe. A une quelconque autre réponse, on suppose qu'il voudrait révoquer son choix de sortir du répertoire. Alors, la fonction *Menu()* renvoie une réponse au *main()* pour signaler qu'elle peut demander ou redemander à l'utilisateur s'il voudrait effectuer une autre opération. A sa réponse négative, le programme s'estompe.

- Ajouter un contact

Au choix de cette opération, il lui est proposé d'entrer le nom, le(s) prénom(s) et le numéro téléphonique du contact qu'il veut ajouter. Ces données sont récupérées grâce à une structure nommée **Contact** et sauvegardées dans un fichier texte en guise de répertoire. A la fin, elle renvoie à la fonction appelante un booléen : **true** si l'ajout est bien fait ou **false** si non, problème lié à l'accès au répertoire.



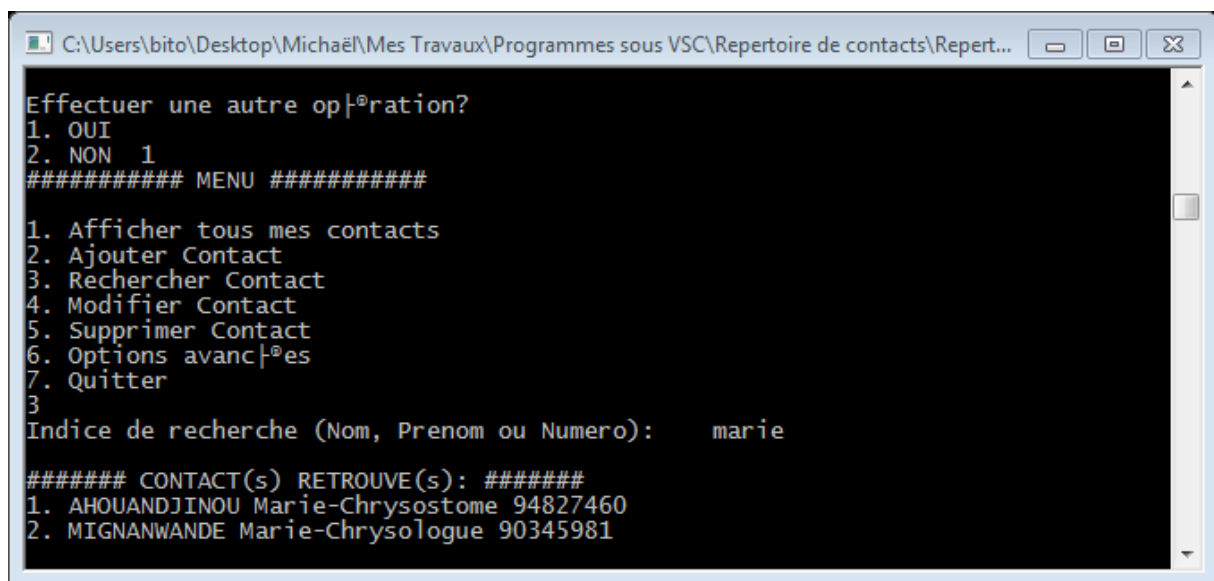
```
C:\Users\bito\Desktop\Michaël\Mes Travaux\Programmes sous VSC\Repertoire de contacts\Repert...
##### MENU #####
1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avancées
7. Quitter
2
NOM:      AHOUANDJINO
PRENOM(s): Elsie
NUMERO: 94827460
##### NOUVEAU CONTACT ENREGISTRE #####
Voulez vous enregistrer un nouveau contact ?
1. OUI 2. NON 1
NOM:      TOBOU
PRENOM(s): Euloge
NUMERO: 65854512
##### NOUVEAU CONTACT ENREGISTRE #####
Voulez vous enregistrer un nouveau contact ?
1. OUI 2. NON 1
NOM:      AGASSOUNON
PRENOM(s): Elsie
NUMERO: 90451236
##### NOUVEAU CONTACT ENREGISTRE #####
Voulez vous enregistrer un nouveau contact ?
1. OUI 2. NON 1
NOM:      AHOUANDJINO
PRENOM(s): Marie-Chrysostome
NUMERO: 94827460
##### NOUVEAU CONTACT ENREGISTRE #####
Voulez vous enregistrer un nouveau contact ?
1. OUI 2. NON 1
NOM:      MIGNANWANDE
PRENOM(s): Marie-Chrysologue
NUMERO: 90345981
##### NOUVEAU CONTACT ENREGISTRE #####
Voulez vous enregistrer un nouveau contact ?
1. OUI 2. NON
```

**Figure 1 : Ajout de quelques contacts**

- Rechercher un contact

Au choix de cette opération, il est demandé à l'utilisateur d'entrer un indice qui permettra à la fonction appelée d'effectuer la recherche dans le répertoire (ici notre fichier.txt). Une fois fait, cette fonction parcourt ligne par ligne le répertoire, à la recherche d'une ligne contenant l'indice donné par l'utilisateur. Dès qu'elle en trouve, elle range le résultat de ses recherches dans un tableau déclaré globalement donc

accessible à toutes les fonctions. Le tableau étant initialement vide, s'il le demeure, alors la pêche aux contacts aurait été mauvaise : contact recherché inexistant ou problème d'accès au répertoire. S'il n'est plus vide après la recherche, la fonction appelante pourra afficher le fruit de ses recherches en accédant au tableau appelé ici **TableauContact**. Au niveau de cette fonction, il est également géré le problème de la casse. A l'image des répertoires de contacts dont nous disposons, l'utilisateur pourrait entrer un indice de recherche en **MAJUSCULE** alors qu'il aurait enregistré le contact avec des lettres **minuscules** (et inversement). Cette fonction retrouve quand même les contacts recherchés s'ils existent et les range dans **TableauContact**.



```

C:\Users\bito\Desktop\Michaël\Mes Travaux\Programmes sous VSC\Repertoire de contacts\Repert...
Effectuer une autre op@ration?
1. OUI
2. NON 1
##### MENU #####
1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avanc@es
7. Quitter
3
Indice de recherche (Nom, Prenom ou Numero):  marie

##### CONTACT(s) RETROUVE(s): #####
1. AHOUANDJINOU Marie-Chrysostome 94827460
2. MIGNANWANDE Marie-Chrysologue 90345981
  
```

**Figure 2 :** Recherche de contacts contenant « marie » et affichage de toutes les occurrences trouvées (remarquons le problème de casse géré)

- Modifier un contact

Nous établissons ici une relation d'utilisation de la fonction **RechercherContact()** par la fonction **ModifierContact()**. Cette relation est justifiée : l'utilisateur demande à modifier un contact en donnant un indice pour retrouver celui-ci. Il aurait été inconvenant de demander à la fonction **ModifierContact()** de rechercher lui-même le contact à modifier alors qu'il y a déjà une fonction toute faite pour ce travailler. Alors notre fonction en cours l'appelle. En réponse, cette dernière modifie au besoin le tableau devant contenir le fruit de ses recherches ou le maintient vide. La fonction appelante vérifie l'état (rempli ou non) du tableau. S'il n'est pas rempli, le contact à modifier n'existe pas dans le répertoire. S'il est rempli, il est affiché à l'écran les contacts retrouvés et demandé à l'utilisateur de choisir celui qu'il voudrait

modifier. Une fois fait, on lui demande maintenant d'entrer la chaîne à modifier puis la nouvelle chaîne remplaçante. Etant donné qu'il s'agit d'une opération délicate, une confirmation de modification lui est demandée. Puis la modification est étendue au répertoire. Tout comme la fonction précédente, **ModifierContact()** gère le problème lié à la casse.

```

##### MENU #####
1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avancées
7. Quitter
4
Indice de recherche du contact à modifier (Nom, Prénom ou Numéro):    elsie
1. AGASSOUNON Elsie 90451236
2. AHOUANDJINOU Elsie 94827460
Choisir l'indice du contact à modifier:                                1
Donnée à modifier (NOM, PRENOM, NUMERO):                            Elsie
Nouvelle Donnée:              Brunelle
Ce contact sera modifié. Confirmer?
1.OUI  2.NON  1

##### MODIFICATION REUSSIE ! #####

Modifier un autre contact?
1. OUI  2. NON  2
Effectuer une autre opération?
1. OUI
2. NON  1
##### MENU #####
1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avancées
7. Quitter
1

##### TOUS VOS CONTACTS: #####
AGASSOUNON Brunelle 90451236
AHOUANDJINOU Elsie 94827460
AHOUANDJINOU Marie-Chrysostome 94827460
MIGNANWANDE Marie-Chrysologue 90345981
TOBOU Euloge 65854512
#####

```

**Figure 3 :** Modification d'un contact (le contact initial était « AGASSOUNON Elsie 90451236 » puis nous avons remplacé « Elsie » par « Brunelle ». Ensuite nous affichons les contacts du répertoire pour faire remarquer la modification)

- Supprimer un contact



La démarche est semblable à celle de la fonction précédente : **SupprimerContact()** demande à **RechercherContact()** de vérifier l'existence du ou des contacts à supprimer à partir d'un indice de recherche et de les ranger dans un tableau qui lui est accessible. Ensuite, elle vérifie l'état du tableau et demande, si non vide, à l'utilisateur d'entrer l'indice du contact à supprimer après avoir affiché à l'écran le contenu non vide du **TableauContact**. Une confirmation de suppression lui est demandée puisqu'il n'aura plus de retour possible. Puis la suppression est étendue au répertoire.

```

C:\Users\bito\Desktop\Michaël\Mes Travaux\Programmes sous VSC\Repertoire de contacts\Repert...
Effectuer une autre op ration?
1. OUI
2. NON 1
##### MENU #####
1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avanc es
7. Quitter
5
Indice de recherche du contact    supprimer (Nom, Prenom ou Numero):   ahouan
1. AHOUANDJINOU Elsie 94827460
2. AHOUANDJINOU Marie-Chrysostome 94827460
Choisir l'indice du contact    supprimer:      2
Ce contact sera supprim  . Confirmer?
1. OUI 2. NON 1
##### SUPPRESSION REUSSIE ! #####

Supprimer un autre contact?
1. OUI 2. NON 2
Effectuer une autre op ration?
1. OUI
2. NON 1
##### MENU #####
1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avanc es
7. Quitter
1

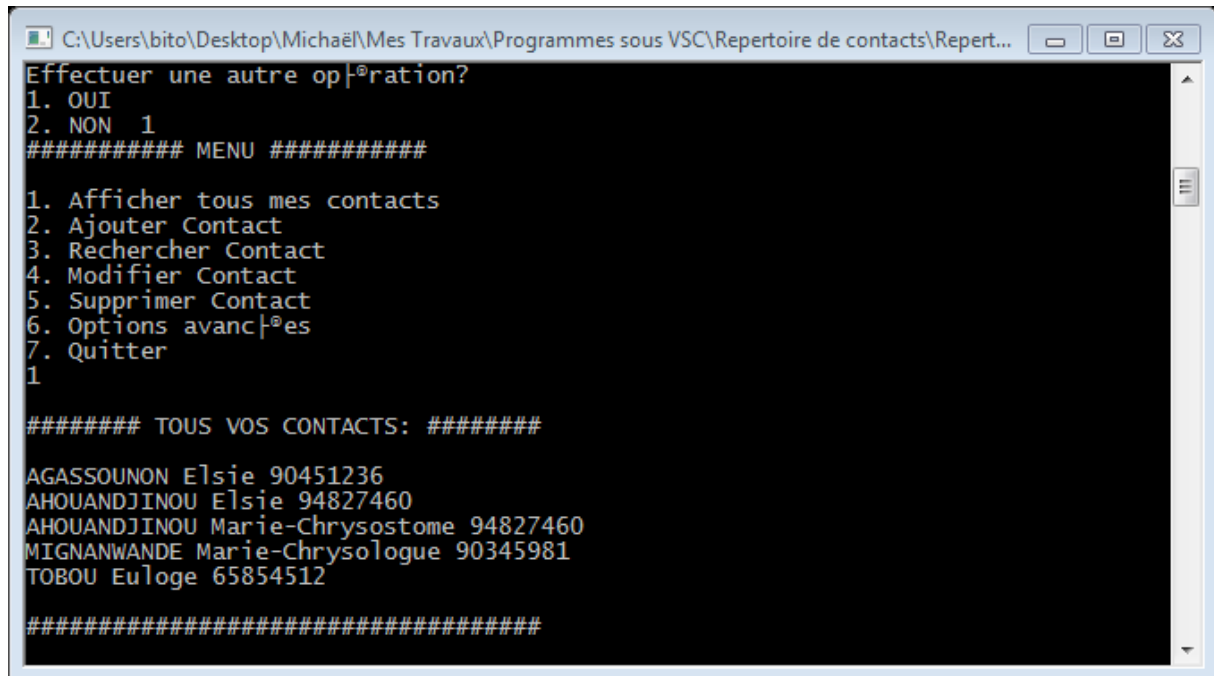
##### TOUS VOS CONTACTS: #####
AGASSOUNON Brunelle 90451236
AHOUANDJINOU Elsie 94827460
MIGNANWANDE Marie-Chrysologue 90345981
TOBOU Euloge 65854512
#####

```

**Figure 4** : Suppression d'un contact (Nous avons supprim   ici le contact « AHOUANDJINOU Marie-Chrysostome 94827460 » apr  s une confirmation de suppression comme d  crit plus haut, puis nous affichons le contenu du r  pertoire afin de faire remarquer que le contact n'y est plus)

- Afficher tous mes contacts

Il s'agit d'une fonction toute simple qui affichera, à la demande de l'utilisateur, tout le contenu de son répertoire.



```

C:\Users\bito\Desktop\Michaël\Mes Travaux\Programmes sous VSC\Repertoire de contacts\Repert...
Effectuer une autre op ration?
1. OUI
2. NON 1
##### MENU #####
1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avanc es
7. Quitter
1
##### TOUS VOS CONTACTS: #####
AGASSOUNON Elsie 90451236
AHOUANDJINOU Elsie 94827460
AHOUANDJINOU Marie-Chrysostome 94827460
MIGNANWANDE Marie-Chrysologue 90345981
TOBOU Euloge 65854512
#####

```

**Figure 5 :** Affichage de tous les contacts initialement enregistr s (ils sont rang s par ordre alphab tique)

```

C:\Users\bito\Desktop\Michaël\Mes Travaux\Programmes sous VSC\Repertoire de contacts\Repert...
Effectuer une autre op ration?
1. OUI
2. NON 1
##### MENU #####
1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avanc es
7. Quitter
2
NOM: DAGA
PRENOM(s): Eunicia
NUMERO: 94112236
##### NOUVEAU CONTACT ENREGISTRE #####
Voulez vous enregistrer un nouveau contact ?
1. OUI 2. NON 2
Effectuer une autre op ration?
1. OUI
2. NON 1
##### MENU #####
1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avanc es
7. Quitter
1
##### TOUS VOS CONTACTS: #####
AGASSOUNON Brunelle 90451236
AHOUANDJINOU Elsie 94827460
DAGA Eunicia 94112236
MIGNANWANDE Marie-Chrysologue 90345981
TOBOU Euloge 65854512
#####

```

**Figure 6 :** Affichage de tous les contacts apr s ex cution de certaines op rations

- R organiser le r pertoire

Apr s une suppression de contact, il est cr   par d faut une ligne vide dans le r pertoire. Cette fonction se chargera de r organiser le r pertoire pour qu'il n'ait pas un aspect trop peu accueillant.

## ***FONCTIONS C++ ET BIBLIOTHEQUES UTILISEES***

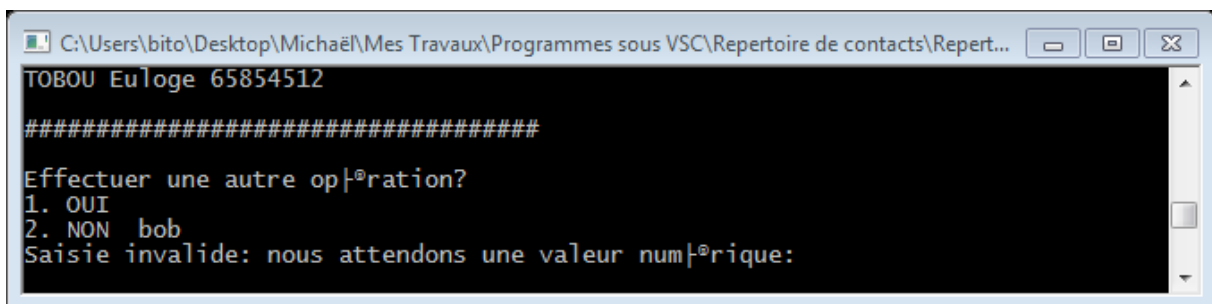
Fonctions	Biblioth�que	R�le
cin.ignore()	Iostream	Ignorer le contenu du buffer lorsqu'on voudra utiliser un getline()

<b>cin.clear()</b>	Iostream	Nettoyer les traces d'une quelconque saisie au clavier (nettoyer le contenu d'une variable)
<b>tolower()</b>	Cctype	Transforme les MAJUSCULES en minuscules
<b>toupper()</b>	Cctype	Transforme les minuscules en MAJUSCULES
<b>length()</b>	String	Donne la taille d'une chaîne de caractères
<b>replace()</b>	String	Remplace le contenu d'une chaîne de caractères par une autre spécifiée
<b>erase()</b>	String	Efface le contenu spécifié d'une chaîne de caractère
<b>find()</b>	String	Recherche une chaîne de caractères dans une autre
<b>size()</b>	Vector	Donne la taille d'un tableau dynamique
<b>empty()</b>	vector	Vérifie si un tableau est vide
<b>clear()</b>	vector	Supprime toute les cases d'un tableau
<b>push_back()</b>	vector	Ajoute une case au tableau
<b>ofstream</b>	fstream	Permet la déclaration d'un flux sur un fichier à ouvrir en écriture
<b>ifstream</b>	fstream	Permet la déclaration d'un flux sur un fichier à ouvrir en lecture

<code>close()</code>	Fstream	Ferme le fichier ouvert avant qu'il ne soit fait par défaut
----------------------	---------	---

## ***DIFFICULTES RENCONTREES ET SOLUTIONS APPORTEES***

- Bug du programme lors d'une saisie alphanumérique alors qu'il est attendu une saisie strictement numérique : pour régler ce problème, nous avons créé une fonction de prototype *int GerErreur(int)* qui est appelée à chaque fois qu'une saisie numérique est attendue. La fonction se charge de vérifier la saisie de l'utilisateur et l'oblige (à sept (07) essais au plus) à corriger sa saisie si elle est incorrecte. A sept (07) erreurs consécutives, le programme est délibérément arrêté avec l'instruction *exit(1)*.

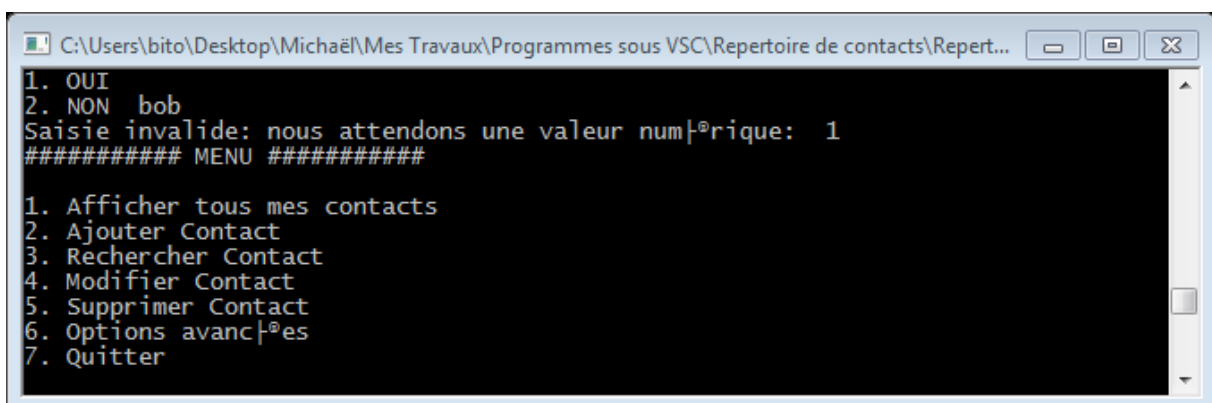


```

C:\Users\bito\Desktop\Michaël\Mes Travaux\Programmes sous VSC\Repertoire de contacts\Repert...
TOBOU Euloge 65854512
#####
Effectuer une autre op ration?
1. OUI
2. NON bob
Saisie invalide: nous attendons une valeur num rique:

```

**Figure 7 :** test d'erreur (le programme attend une valeur numérique mais nous entrons une valeur alphanumérique. Au lieu d'assister à un bug, le programme demande calmement d'entrer une valeur numérique)



```

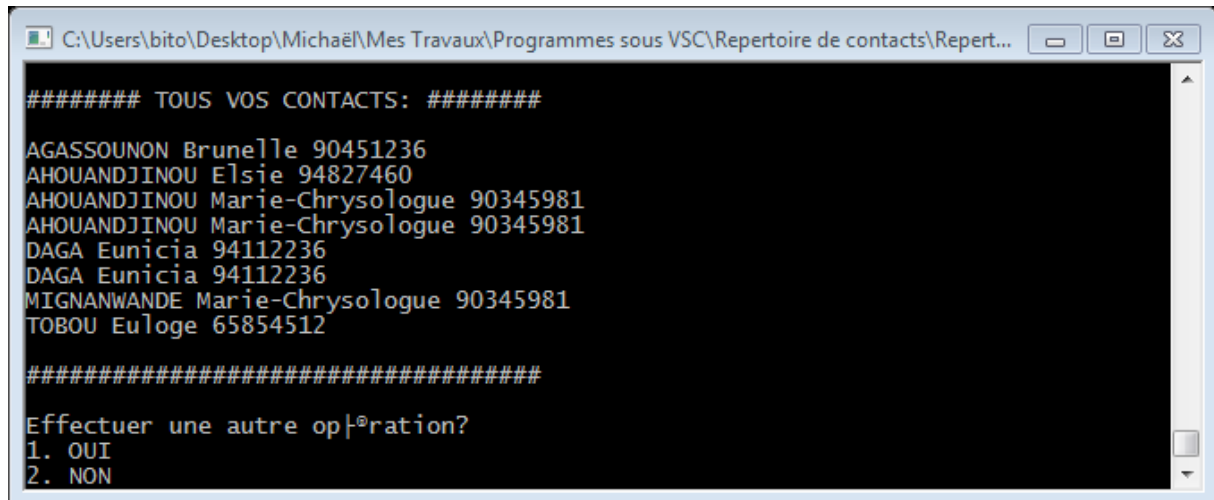
C:\Users\bito\Desktop\Michaël\Mes Travaux\Programmes sous VSC\Repertoire de contacts\Repert...
1. OUI
2. NON bob
Saisie invalide: nous attendons une valeur num rique: 1
##### MENU #####
1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avanc es
7. Quitter

```

**Figure 8 :** Reprise normale du programme après une saisie correcte (dès que la valeur numérique est entrée, l'exécution du programme reprend normalement)

- Il peut arriver que l'utilisateur enregistre plusieurs fois le même contact sans s'en rendre compte. Alors, une fonction toute faite, *bool*

*SupprimerLesDoublons()* lui permettra de supprimer d'un coup les doublons de contacts et de ne conserver, de ce fait, qu'une occurrence de chacun d'eux.



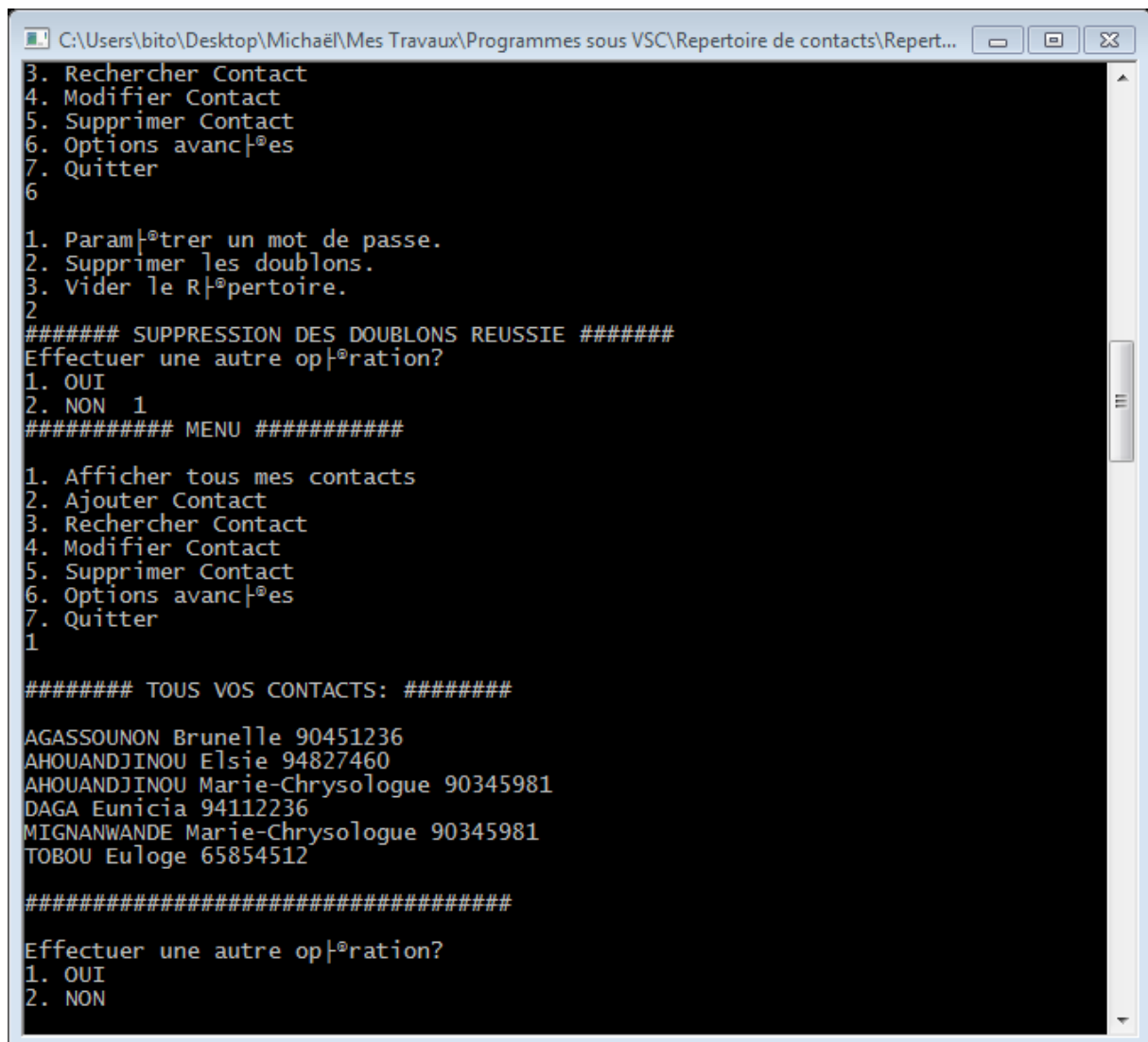
```
C:\Users\bito\Desktop\Michaël\Mes Travaux\Programmes sous VSC\Repertoire de contacts\Repert...

##### TOUS VOS CONTACTS: #####
AGASSOUNON Brunelle 90451236
AHOUANDJINOU Elsie 94827460
AHOUANDJINOU Marie-Chrysologue 90345981
AHOUANDJINOU Marie-Chrysologue 90345981
DAGA Eunicia 94112236
DAGA Eunicia 94112236
MIGNANWANDE Marie-Chrysologue 90345981
TOBOU Euloge 65854512

#####

Effectuer une autre op ration?
1. OUI
2. NON
```

**Figure 9 : Enregistrement de plusieurs contacts identiques**



```
C:\Users\bito\Desktop\Michaël\Mes Travaux\Programmes sous VSC\Repertoire de contacts\Repert...

3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avanc es
7. Quitter
6

1. Param trer un mot de passe.
2. Supprimer les doublons.
3. Vider le R pertoire.
2
##### SUPPRESSION DES DOUBLONS REUSSIE #####
Effectuer une autre op ration?
1. OUI
2. NON 1
##### MENU #####

1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avanc es
7. Quitter
1

##### TOUS VOS CONTACTS: #####
AGASSOUNON Brune lle 90451236
AHOUANDJINOU Elsie 94827460
AHOUANDJINOU Marie-Chrysologue 90345981
DAGA Eunicia 94112236
MIGNANWANDE Marie-Chrysologue 90345981
TOBOU Euloge 65854512

#####

Effectuer une autre op ration?
1. OUI
2. NON
```

**Figure 10 : Suppression des doublons pr c demment ins r s**

- Au cours des divers tests du programme, nous avons eu besoin de nettoyer complètement le répertoire. Comme il aurait été plus pratique de le faire sans avoir à quitter le programme, nous avons alors pensé à une fonction qui permettrait de combler ce manque. Il s'agit de la fonction de prototype ***void** **ViderLeRepertoire()***. Etant donné qu'il s'agit d'une opération assez délicate, nous avons conditionné son exécution à un mot de passe, d'où la fonction ***bool** **MotDePasse(string)*** qui servira à vérifier la justesse du « mot de passe » entré par l'utilisateur. A une première utilisation de cette commande, on demande à l'utilisateur de paramétrer ses données de sécurité, ce qui nous conduit à la fonction ***bool** **ParametrerSecurite()***. En cas de mot de passe oublié, la fonction ***bool** **VerifierIdentite()*** se charge de vérifier l'identité de l'utilisateur légitime. Si la vérification est positive, l'utilisateur devra reparamétrer ses données de sécurité.

```
C:\Users\bito\Desktop\Michaël\Mes Travaux\Programmes sous VSC\Repertoire de contacts\Repert...
6. Options avancées
7. Quitter
6

1. Paramétrer un mot de passe.
2. Supprimer les doublons.
3. Vider le Répertoire.
3
Pour effectuer cette opération, vous aurez besoin de configurer les données de
sécurité.
##### CONFIGURATION DE SECURITE #####

Renseigner les champs.
##### Mot De Passe #####
Entrer Nouveau Mot De Passe: Goyaves7
Confirmer Mot De Passe: Goyaves7

##### Questions de sécurité en cas de mot de passe oublié #####
Nom de votre chien ? Dack
Votre ville d'origine ? Québec
La couleur de votre téléphone ? Bleu ciel
Nombre d'essai(s) restant(s): 3

1. Entrer le mot de passe.
2. Mot de passe oublié. Goyaves7
Saisie invalide: nous attendons une valeur numérique: 1
Mot de passe: Goyaves7
Accès Autorisé !
Le répertoire sera vidé. Voulez vous continuer ?
1. OUI 2. NON 1
##### REPERTOIRE VIDE #####
Effectuer une autre opération?
1. OUI
2. NON 1
##### MENU #####

1. Afficher tous mes contacts
2. Ajouter Contact
3. Rechercher Contact
4. Modifier Contact
5. Supprimer Contact
6. Options avancées
7. Quitter
1

##### TOUS VOS CONTACTS: #####

#####
```

**Figure 11** : test de la suppression générale : Vider le répertoire (Pour une première utilisation, configuration de sécurité demandée, puis, demande de mot de passe avant la suppression totale de tous les contacts du répertoire. Remarquez qu'à l'affichage, nous n'avons plus de contacts dans notre répertoire)

- Quand un nouveau contact est ajouté, il est mis à la fin du répertoire. Dès que le répertoire est un peu bondé, l'utilisateur, en affichant tous ses contacts, pourrait avoir du mal à se retrouver. Alors, nous nous sommes chargés de faire trier et ranger automatiquement les contacts par ordre alphabétique, en utilisant la fonction *void ParOrdreAlphabetique()*



## ***A SAVOIR AVANT L'EXECUTION DU PROGRAMME***

Il serait fastidieux de demander à l'utilisateur à chaque lancement du programme de bien vouloir entrer l'adresse absolue ou relative de son fichier.txt (en guise de répertoire), s'il existe déjà, ou l'emplacement où il voudra le mettre.

Par défaut, un chemin relatif est utilisé pour spécifier l'emplacement du répertoire de contacts et du fichier privé qui contiendra les données de sécurité.

S'il s'agit d'un usage permanent et s'il le désire, l'utilisateur devra modifier une fois pour toute une ligne du code dans le fichier Repertoire.cpp (le main.cpp). Il s'agit de la ligne 9 où il y a par défaut :

```
string adresse("Contacts.txt"),adresseP("personnel.pdf");
```

Si la ligne n'est pas modifiée, le chemin par défaut est maintenu.

## ***OBJECTIVATION***

Il nous a été demandé d'écrire un C++ qui permettra à un utilisateur de gérer un répertoire téléphonique. Les difficultés rencontrées sont diverses mais ont pu être au mieux surmontées grâce à l'ajout d'autres fonctions autres que les standards attendues. Toutefois, ce programme est ouvert à une quelconque amélioration.

## **TABLE DES ILLUSTRATIONS**

Figure 1 : Ajout de quelques contacts .....	6
Figure 2 : Recherche de contacts contenant « marie » et affichage de toutes les occurrences trouvées.....	7
Figure 3 : Modification d'un contact.....	8
Figure 4 : Suppression d'un contact .....	9
Figure 5 : Affichage de tous les contacts initialement enregistrés .....	10
Figure 6 : Affichage de tous les contacts après exécution de certaines opérations .....	11
Figure 7 : test d'erreur.....	13
Figure 8 : Reprise normale du programme après une saisie correcte .....	13
Figure 9 : Enregistrement de plusieurs contacts identiques .....	14
Figure 10 : Suppression des doublons précédemment insérés .....	14
Figure 11 : test de la suppression générale : Vider le répertoire.....	16