```
#include <cstdlib>
#include <cstdio>
#include <stdlib.h>
#include <stdio.h>
#include <iostream>
#include <string>
#include <fstream>
#include <sys/types.h>
#ifdef WIN32
#include <windows.h>
#include <winsock2.h>
#include <commctrl.h>
#else
#include <sys/socket.h>
#include <pthread.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#endif
#include <unistd.h>
#include <memory.h>
#include <signal.h>

using namespace std;

void consultation(void);
void transfert(void);

int main(int argc, char* argv[])
{

    int choix=-1;
    while(1)
    {
        printf("Quelle operation voulez-vous effectuer ?\n");
        printf("- 0 : Quitter l'application\n");
        printf("- 1 : Consulter votre solde\n");
        printf("- 2 : Effectuer un virement\n");
        scanf("%i",&choix);

        if(choix == 0) // on quitte
                return 0;
```

```
        else if(choix == 1) // consultation
             consultation();

        else if(choix == 2) // virement
             transfert();

        system("PAUSE");  // attente d'une pression d'une touche du clavier
        system("CLS");   // effacement de l'ecran
    }


    return 0;

}


void consultation(void)
{

// code pour activer l'API socket de Windows
        WSADATA         wsaData;
        int             res;
        if((res = WSAStartup(MAKEWORD(2,0), &wsaData)) != 0)       // initialisation de l'API socket sous
Windows
                printf("Impossible d'initialiser l'API Winsock 2.0\n");
// ! code pour activer l'API socket de Windows



// code pour creer un socket de type TCP
  int sock;
  if ((sock=socket(AF_INET,SOCK_STREAM,0)) == -1)
  {
    perror("Creation de socket impossible");
    return;
  }
// ! code pour creer un socket de type TCP
```

```c
// configuration de l'adresse socket
  struct sockaddr_in adresse_socket;
  int longueur=sizeof(struct sockaddr_in);
  int  port=5666; // port de communication
  char *adresse = "127.0.0.1"; // adresse IP du serveur

   memset(&adresse_socket,0x0,sizeof(adresse_socket)); // on met toute la structure adresse_socket a zero

   adresse_socket.sin_family = AF_INET;
   adresse_socket.sin_port = htons(port);
   adresse_socket.sin_addr.S_un.S_addr = inet_addr(adresse);
// ! configuration de l'adresse socket



// demande de connexion au serveur
   if((res = connect(sock,(struct sockaddr *) &adresse_socket, sizeof(adresse_socket))) != 0)
                printf("Impossible de se connecter !\n");
// ! demande de connexion au serveur



// enregistrement des informations
        char    nom[100];
        char    prenom[100];
        char    idcompte[100];
        char    mdp[100];

        // remplir les champs pour la consultation
        printf("\nVous avez demande a consulter le solde d'un compte\n\n");

        printf("Quel est l'identifiant du compte que vous voulez-consulter ?\n");
        scanf("%s",&idcompte);
        if(strlen(idcompte) != 7)
        {
           printf("erreur, id doit faire 7 caracteres\n");
        }
        printf("Quel est le nom associe au compte ?\n");
        scanf("%s",&nom);
        printf("Quel est le prenom associe au compte ?\n");
        scanf("%s",&prenom);
```

```
        printf("Quel est le mot de passe du compte ?\n");
        scanf("%s",&mdp);
        if(strlen(mdp) != 5)
        {
           printf("erreur, id doit faire 5 caracteres\n");
        }
// ! enregistrement des informations



// envoie des donnees au serveur
        if(send(sock,idcompte,sizeof(idcompte),0) != sizeof(idcompte))
                printf("Echec de l'envoie des données idcompte !\n");
        if(send(sock,nom,sizeof(nom),0) != sizeof(nom))
                printf("Echec de l'envoie des données nom !\n");
        if(send(sock,prenom,sizeof(prenom),0) != sizeof(prenom))
                printf("Echec de l'envoie des données prenom !\n");
        if(send(sock,mdp,sizeof(mdp),0) != sizeof(mdp))
                printf("Echec de l'envoie des données mdp !\n");
// ! envoie des donnees au serveur



// reception des donnees
        int     receive=0; // nb octets recus
        char    etat[100];
        char    date[10];
        char    heure[10];
        char    solde[100];

        if((receive = recv(sock,etat,2,0)) != 2)
                printf("Echec de reception des donnees !\n");
        if(etat[0] == '1')
                printf("Echec identification !\n");
        else
        {
                if((receive = recv(sock,date,sizeof(date),0)) != sizeof(date))
                            printf("Echec de reception des donnees date !\n");
                date[receive] = '\0';// place à la fin du tableau le caractère nulle
                if((receive = recv(sock,heure,sizeof(heure),0)) != sizeof(heure))
                            printf("Echec de reception des donnees heure !\n");
```

```
                heure[receive] = '\0';// place à la fin du tableau le caractère nulle
                if((receive = recv(sock,solde,100,0)) != sizeof(solde))
                        printf("Echec de reception des donnees solde !\n");
                solde[receive] = '\0';// place à la fin du tableau le caractère nulle

                printf("le solde au %s a %s est de %s euros\n",date,heure,solde);          // on afficher le
contenu du buffer
        }
// ! reception des donnees



// fermeture du socket
        shutdown(sock,2);
        close(sock); // ferme le socket
// ! fermeture du socket


        return; // EXIT_SUCCESS
}


void transfert(void)
{
// code pour activer l'API socket de Windows
        WSADATA         wsaData;
        int             res;
        if((res = WSAStartup(MAKEWORD(2,0), &wsaData)) != 0)       // initialisation de l'API socket sous
Windows
                printf("Impossible d'initialiser l'API Winsock 2.0\n");
// ! code pour activer l'API socket de Windows



// code pour creer un socket de type TCP
  int sock;
  if ((sock=socket(AF_INET,SOCK_STREAM,0)) == -1)
  {
    perror("Creation de socket impossible");
    return;
  }
```

```c
// ! code pour creer un socket de type TCP



// configuration de l'adresse socket
  struct sockaddr_in adresse_socket;
  int longueur=sizeof(struct sockaddr_in);
  int  port=5999; // port de communication
  char *adresse = "127.0.0.1"; // adresse IP du serveur

   memset(&adresse_socket,0x0,sizeof(adresse_socket)); // on met toute la structure adresse_socket a zero

   adresse_socket.sin_family = AF_INET;
   adresse_socket.sin_port = htons(port);
   adresse_socket.sin_addr.S_un.S_addr = inet_addr(adresse);
// ! configuration de l'adresse socket



// demande de connexion au serveur
   if((res = connect(sock,(struct sockaddr *) &adresse_socket, sizeof(adresse_socket))) != 0)
                printf("Impossible de se connecter !\n");
// ! demande de connexion au serveur



// enregistrement des informations
        char    nomA[100],nomB[100];
        char    prenomA[100],prenomB[100];
        char    somme[100];
        char    idcompteA[100],idcompteB[100];
        char    mdpA[100],mdpB[100];

        // remplir les champs pour la consultation
        printf("\nVous avez demande a consulter le solde d'un compte\n\n");

        printf("Quel est l'identifiant du compte A dont vous voulez retirer de l'argent ?\n");
        scanf("%s",&idcompteA);
        if(strlen(idcompteA) != 7)
        {
           printf("erreur, id doit faire 7 caracteres\n");
```

```
        }
        printf("Quel est le nom associe au compte A ?\n");
        scanf("%s",&nomA);
        printf("Quel est le prenom associe au compte A ?\n");
        scanf("%s",&prenomA);
        printf("Quel est le mot de passe du compte A ?\n");
        scanf("%s",&mdpA);
        if(strlen(mdpA) != 5)
        {
           printf("erreur, id doit faire 5 caracteres\n");
        }

        printf("\nQuel est l'identifiant du compte B dont vous voulez effectuer un virement ?\n");
        scanf("%s",&idcompteB);
        if(strlen(idcompteB) != 7)
        {
           printf("erreur, id doit faire 7 caracteres\n");
        }
        printf("Quel est le nom associe au compte B ?\n");
        scanf("%s",&nomB);
        printf("Quel est le prenom associe au compte B ?\n");
        scanf("%s",&prenomB);
        printf("Quel est le mot de passe du compte B ?\n");
        scanf("%s",&mdpB);
        if(strlen(mdpB) != 5)
        {
           printf("erreur, id doit faire 5 caracteres\n");
        }

        printf("\nQuelle est la somme a virer ?\n");
        scanf("%s",&somme);
        if(somme[0] == '-')
        {
           printf("erreur, somme doit etre positif\n");
           printf("en consequence, virement de 0 euros\n");
           somme[0]='0';somme[1]='\0';
        }
// ! enregistrement des informations
```

```
// envoie des donnees au serveur
        if(send(sock,idcompteA,sizeof(idcompteA),0) != sizeof(idcompteA))
                printf("Echec de l'envoie des donnees idcompteA !\n");
        if(send(sock,nomA,sizeof(nomA),0) !=sizeof(nomA))
                printf("Echec de l'envoie des donnees nomA !\n");
        if(send(sock,prenomA,sizeof(prenomA),0) != sizeof(prenomA))
                printf("Echec de l'envoie des donnees prenomA !\n");
        if(send(sock,mdpA,sizeof(mdpA),0) != sizeof(mdpA))
                printf("Echec de l'envoie des donnees mdpA !\n");

        if(send(sock,idcompteB,sizeof(idcompteB),0) != sizeof(idcompteB))
                printf("Echec de l'envoie des donnees idcompteB !\n");
        if(send(sock,nomB,sizeof(nomB),0) != sizeof(nomB))
                printf("Echec de l'envoie des donnees nomB !\n");
        if(send(sock,prenomB,sizeof(prenomB),0) != sizeof(prenomB))
                printf("Echec de l'envoie des donnees prenomB !\n");
        if(send(sock,mdpB,sizeof(mdpB),0) != sizeof(mdpB))
                printf("Echec de l'envoie des donnees mdpB !\n");

        if(send(sock,somme,sizeof(somme),0) != sizeof(somme))
                printf("Echec de l'envoie des donnees somme !\n");
// ! envoie des donnees au serveur



// reception des donnees
        int     receive=0; // nb octets recus
        char    etatA[100],etatB[100],etat[100];
        char    date[10];
        char    heure[10];
        char    soldeA[100],soldeB[100];

        if((receive = recv(sock,etatA,2,0)) != 2)
                printf("Echec de reception des donnees etat !\n");
        if((receive = recv(sock,etatB,2,0)) != 2)
                printf("Echec de reception des donnees etat !\n");
        if(etatA[0] == '1' || etatB[0] == '1')
                printf("Echec identification !\n");
        else
        {
                if((receive = recv(sock,etat,2,0)) != 2)
```

```
                        printf("Echec de reception de la donnee etat !\n");
                if(etat[0] == '1')
                        printf("Virement impossible car solde A negatif !\n");

                if((receive = recv(sock,date,sizeof(date),0)) != sizeof(date)) // fonction bloquante en
attente d'un message
                        printf("Echec de reception des donnees date !\n");
                date[receive] = '\0';// place à la fin du tableau le caractère nul
                if((receive = recv(sock,heure,sizeof(heure),0)) != sizeof(heure))
                        printf("Echec de reception des donnees heure !\n");
                heure[receive] = '\0';// place à la fin du tableau le caractère nul
                if((receive = recv(sock,soldeA,sizeof(soldeA),0)) != sizeof(soldeA))
                        printf("Echec de reception des donnees soldeA !\n");
                soldeA[receive] = '\0';// place à la fin du tableau le caractère nul
                if((receive = recv(sock,soldeB,sizeof(soldeB),0)) != sizeof(soldeB))
                        printf("Echec de reception des donnees soldeB !\n");
                soldeB[receive] = '\0';// place à la fin du tableau le caractère nul
                if(etat[0] != '1')
                        printf("le virement a ete effectue le %s a %s\n",date,heure);
                printf("le nouveau solde du compte A est %s euros\n",soldeA);
                printf("le nouveau solde du compte B est %s euros\n",soldeB);
        }
// ! reception des donnees



// fermeture du socket
        shutdown(sock,2);
        close(sock); // ferme le socket
// ! fermeture du socket



        return; // EXIT_SUCCESS
}
```