```cpp
#ifndef CLASSECLIENTE_H_
#define CLASSECLIENTE_H_
#include"ItemPersistencia.h"
#include<string>

//TIPO DE CLASSE 0 = CLIENTE

//*************************************************************************
//                                    Construida e testeda
//*************************************************************************

class Cliente: public ItemPersistencia
{
    private:
        std::string nome;
        long telefone;
        std::string email;
    public:
        Cliente():ItemPersistencia(0){tipoDaClasse=0;};
        Cliente(    unsigned int identificador, const std::string &nome,
                            long telefone, const std::string &email);
        ~Cliente(){};
        void obter( unsigned int &identificador,std::string &nome,
                                    long &telefone,std::string &email)const;
        void atribuir(  unsigned int identificador, const std::string &nome,
                                        long telefone, const std::string &email);
        const std::string desmaterializar();
        //transforma atributos em uma string, esta obtendo os dados dos atributos p
        void materializar(const std::string s);
        //recebe uma string como parametro e armazena nos atributos os respectivos
};

Cliente::Cliente(   unsigned int identificador, const std::string &nome,
                                    long telefone, const std::string &email):
            ItemPersistencia(identificador),
            //recebe um numero(unsigned int) digitado "no programa principal"
            nome(nome),
            telefone(telefone),
            email(email)
{
    tipoDaClasse=0;
    //este tipo seria para identificacao q eh uma pessoa para a classe itemPers
}

void Cliente::obter(    unsigned int &identificador,std::string &nome,
                                            long &telefone,std::string &email)const
{
    identificador = this->identificador;
    nome = this->nome;
    telefone = this->telefone;
    email = this->email;
}

void Cliente::atribuir( unsigned int identificador,const std::string &nome,
                                                long telefone,const std::string &em
{
    this->identificador=identificador;
    this->nome=nome;
    this->telefone=telefone;
    this->email=email;
}

```

```
 63: const std::string Cliente::desmaterializar()
 64: //transformando atributos em string's, linha
 65: //mesma funcao do metodo obter
 66: {
 67:     std::string linha;
 68:     std::string idString;
 69:     for(unsigned int aux=identificador;aux;){
 70:         char letra= (aux%10)+48;
 71:         idString=letra+idString;
 72:         aux=aux/10;
 73:     }
 74:     linha=idString;
 75:     linha +=";";
 76:     linha+=nome;
 77:     linha+=";";
 78:     std::string foneString;
 79:     for(unsigned int aux=telefone;aux;){
 80:         char letra= (aux%10)+48;
 81:         foneString=letra+foneString;
 82:         aux=aux/10;
 83:     }
 84:     linha+=foneString;
 85:     linha+=";";
 86:     linha+=email;
 87:     linha+=";";
 88:     return linha;
 89: }
 90:
 91: void Cliente::materializar(const std::string s)
 92: //transformando string em atributos
 93: //mesma funcao do metodo atribuir
 94: {
 95:     unsigned int aux=0;
 96:     unsigned int pos=0;
 97:     for(;s[pos]!=';';pos++) aux=aux*10+(s[pos]-48);
 98:     identificador=aux;
 99:     std::string strAux="";
100:     for(pos++;s[pos]!=';';pos++) strAux+=s[pos];
101:     nome=strAux;
102:     long aux2=0;
103:     for(pos++;s[pos]!=';';pos++) aux2=aux2*10+(s[pos]-48);
104:     telefone=aux2;
105:     strAux="";
106:     for(pos++;s[pos]!=';';pos++) strAux+=s[pos];
107:     email=strAux;
108: }
109:
110: #endif /*CLASSECLIENTE_H_*/
111:
```