Projet

Exemple

```
    while (str1[i] != '\0' && j < sl)
    {
        if (str1[i] == str2[j])
            ++j;
        else
        {
            i -= j;
            j = 0;
        }
        ++i;
    }
    return (sl == j);
}

// Classe d'element de liste de string
class String
{
    public:
        friend ile;
        friend FileSystem;
    private:
        String(char ninfo[], String* nnext, String* nprev);
        String();
        String* next;
        String* prev;
        char* info;
};

// Classe fichier. (Cha
```

```
// elle contient un érpertoire: le repertoire racine.
class FileSystem                                    File::File(char nname[], File* nnext)
{
   public:
      FileSystem();
      void Cdd(char gtname[]);
      void InsereRepInCur(char nname[]);
      void InsereFileInCur(char nname[]);
      void SearchName(char sname[]);
      void SearchStr(char sname[]);
      void DispCur();
      void DispRep(Directory* tbd);
      File* OpenFile(char oname[]);
      Directory* root;
      Directory* current;
};

// Constructeur pour les chaines de caracteres: en parametre: (contenu, suivant, precedent)
String::String(char ninfo[], String* nnext, String* nprev)
{
    info = new char[strlen(ninfo)];
    strcpy(info,ninfo);
    next=nnext;
    prev=nprev;
};

// Constructeur pour l'element bidon.
String::String()
{
      info = NULL;
}

// Constructeur de fichier: en parametre (nom, fichier_suivant

{
   str_list = new String;
   str_list->next = str_list;
   str_list->prev = str_list;
   next = nnext;
   name = new char[strlen(nname)];
   strcpy(name,nname);
}

// Constructeur pour lé'élment bidon des listes de fichiers.
File::File()
{
   name = NULL;
}

// Methode pour ajouter une chaine en debut de fichier
void File::AddStringT(char content[])
{
   str_list->next = new String(content,str_list->next,str_list);
   str_list->next->next->prev = str_list->next;
}

// Idem pour ajouter en fin de fichier
void File::AddStringB(char content[])
{
   str_list->prev = new String(content,str_list,str_list->prev);
   str_list->prev->prev->next = str_list->prev;
}
```

```
// Constructeur pour les repertoires: en param: (nom, rep_suivant, rep_parent)
Directory::Directory(char nname[], Directory* nnext, Directory* nparent)
{
   first_file = new File();
   first_file->next = first_file;
   name = new char[strlen(nname)];
   strcpy(name,nname);
   sub = new Directory(this);
   sub->next=sub;
   next = nnext;
   parent = nparent;
}

// Constructeur pour l'element bidon des listes de repertoires.
Directory::Directory(Directory* nparent)
{
   sub = NULL;
   first_file = NULL;
   name = NULL;
   parent = nparent;
}

// Cree un fichier dans le repertoire
void Directory::NewFile(char nname[])
{
   first_file->next = new File(nname,first_file->next);
}

// Cree un sous repertoire dans le repertoire
void Directory::NewDir(char nname[])
{
   sub->next = new Directory(nname, sub->next, this);
}

// Constructeur du systeme de fichier
FileSystem::FileSystem()
{
   root = new Directory("root", NULL, NULL);
   root->next=root;
   current = root;
}

// Va dans le sous reprtoire dont le nom est dans gtname
void FileSystem::Cdd(char gtname[])
{
   Directory* curs = current->sub;
   if (strcmp("..",gtname) != 0)
   {
      curs->name = new char[strlen(gtname)];
      strcpy(curs->name, gtname);
      do {
         curs = curs->next;
      } while(strcmp(curs->name, gtname) != 0);
      delete[] current->sub->name;
      current->sub->name = NULL;
      if (curs != current->sub)
         current = curs;
   }
   else
   {
      current = current->parent;
```

```
    }
}

// Insere un repertoire dans le rep courant.
void FileSystem::InsereRepInCur(char nname[])
{
    current->NewDir(nname);
}

// Affiche le repertoire courant.
void FileSystem::DispCur()
{
    DispRep(current);
}

// Affiche le nom complet d'un repertoire donne
void FileSystem::DispRep(Directory* tbd)
{
    char ret[256] = "";
    int tlen, rlen=0;
    Directory* pos = tbd;
    while(pos != root)
    {
        tlen=strlen(pos->name);
        for (int i=rlen+1; i>=0; --i)
        {
            ret[i + tlen + 1] = ret[i];
        }
        rlen+=tlen;
        strncpy(ret + 1, pos->name, tlen);
        ret[0] = '/';
        pos = pos->parent;
    }
    cout << ret << endl;
}

// Ajoute un fichier dans le repertoire courant
void FileSystem::InsereFileInCur(char nname[])
{
    current->NewFile(nname);
}

// Renvoie l'adresse du fichier nome oname
File* FileSystem::OpenFile(char oname[])
{
    File* pf = current->first_file;
    pf->name = new char[strlen(oname)];
    strcpy(pf->name ,oname);
    do {
        pf = pf->next;
    } while(strcmp(pf->name, oname) != 0);
    delete[] current->first_file->name;
    current->first_file->name = NULL;
    if (pf == current->first_file)
        pf = NULL;
    return pf;
}

// Affiche tous les fichiers dont le nom complet contient sname
void FileSystem::SearchName(char sname[])
{
    Directory* x =                                          o
```

```
    File* ptf;
    while(x != root)
    {
        ptf = x->first_file;
        if (x->first_file != NULL)
        {
            char ret[256] = "/";
            int tlen, rlen=1;
            Directory* pos = x;
            while(pos != root)
            {
                tlen=strlen(pos->name);
                for (int i =rlen+1; i >=0; --i)
                {
                    ret[i + tlen + 1] = ret[i];
                }
                rlen+=tlen+1;
                strncpy(ret + 1, pos->name, tlen);
                ret[0] = '/';
                pos = pos->parent;
            }
            do
            {
                ptf = ptf->next;
                if (ptf->name != NULL)
                    for (int q = 0; q <= strlen(ptf->name); ++q)
                    {
                        ret[rlen+q] = ptf->name[q];
                    }

                if (check_if_inc(ret,sname))
                {
                    cout << ret << endl;
                }
                ret[rlen] = '\0';
            } while(ptf->name != NULL);
        }
        if (x->sub == NULL)
        {
            x = x->parent;
            x = x->next;
        }
        else
        {
            x = x->sub->next;
        }

    }
}

// Affiche tous les fichiers dont le contenu contient sname
void FileSystem::SearchStr(char sname[])
{
    Directory* x = root->sub->next;
    File* ptf;
    while(x != root)
    {
        ptf = x->first_file;
        if (x->first_file != NULL)
        {
            char ret[256] = "/";
            int tlen, rlen=1;
```

```
            Directory* pos = x;
            while(pos != root)
            {
               tlen=strlen(pos->name);
               for (int i =rlen+1; i >=0; --i)
               {
                  ret[i + tlen + 1] = ret[i];
               }
               rlen+=tlen+1;
               strncpy(ret + 1, pos->name, tlen);
               ret[0] = '/';
               pos = pos->parent;
            }
            do
            {
               ptf = ptf->next;

               if (ptf->name != NULL)
               {
                  String* stra = ptf->str_list;
                  for (int q = 0; q <= strlen(ptf->name); ++q)
                  {
                     ret[rlen+q] = ptf->name[q];
                  }
                  do
                  {
                     stra = stra->next;
                     if (stra->info != NULL && check_if_inc(stra->info,sname))
                     {
                        cout << ret << endl;
                     }
                  } while (stra->info != NULL);
                  ret[rlen] = '\0';
               }
            } while(ptf->name != NULL);
         }
         if (x->s!20!!ub == NULL)
         {
            x = x->parent;
            x = x->next;
         }
         else
         {
            x = x->sub->next;
         }

      }
}
```