1 Reading The TV Series File

First two shows in the data file

```
Kung Fu: The Legend Continues (1993-1997)
Adventure
http://www.imdb.com/title/tt0103460/
David Carradine
Chris Potter
Richard Anderson
Kim Chan

Matlock (1986-1995)
Mystery
http://www.imdb.com/title/tt0090481/
Andy Griffith
Nancy Stafford
Julie Sommars
Clarence Gilyard Jr.
Kene Holliday
```

Where to start?

Where to start?

What is the data file format?

```
title    (year)
category
IMDB URL
actor name
actor name

blank line(s)

Repeat
```

What if the file format was not regular?

How to copy a file?

1. Character at a time.
2. Line at a time.
3. All at once.

Which of these do you think would be the best starting point for processing the TV Series file? Why?

1.1 Copy: Line at a time

```
#include <fstream>
#include <iostream>
#include <cstdlib>

using namespace std;

int main()
{
    const int MAX_LINE = 128;
    char line[MAX_LINE];

    ifstream fIn;

    fIn.open( "test.dat", ios::in );
    if( !fIn )
    {
       cerr << "Can't open input file" << endl;
       exit( -1 );
    }

    while( fIn.getline( line, MAX_LINE ) )
    {
       cout << line << endl;
    }

    fIn.close();

    return 0;
}
```

1.2 Version 1: ReadTVFile

```
int ReadTVFile()
{
   //ifstream fIn( "tvDB.txt", ios::in );
   ifstream fIn( "tvDB_Test.txt", ios::in );
   if( !fIn )
   {
      cout << "Unable to open \"tvDB\" data file" << endl;
      exit( -1 );
   }

   const int MAX_LINE = 128;
   char line[MAX_LINE];
   char seriesName[MAX_LINE];

   int nSeries = 0;

   while( fIn.getline( line, MAX_LINE ) )
   {
       strcpy( seriesName, line );
       cout << "Series name: " << seriesName << endl;

       fIn.getline( line, MAX_LINE/2 );
       while( strlen(line) > 0 ) {
           cout << "   actor name:  |" << line << "|" << endl;
           fIn.getline( line, MAX_LINE/2 );
       }

       nSeries++;
   }

   fIn.close();

   return nSeries;
}
```

Test file:

```
JAG (1995-2005)
Action
http://www.imdb.com/title/tt0112022/
David James Elliott
Patrick Labyorteaux
Catherine Bell
John M. Jackson
Karri Turner



Matlock (1986-1995)
Mystery
http://www.imdb.com/title/tt0090481/
Andy Griffith
Nancy Stafford
Julie Sommars
Clarence Gilyard Jr.
Kene Holliday

NCIS (2003-2014)
Action
http://www.imdb.com/title/tt0364845/
Michael Weatherly
Pauley Perrette
David McCallum
Mark Harmon
Sean Murray
Cote de Pablo
Brian Dietzen
Rocky Carroll
Lauren Holly
Sasha Alexander
Joe Spano
```

The Flying Nun (1967-1970)
Comedy
http://www.imdb.com/title/tt0061252/
Sally Field
Marge Redmond
Madeleine Sherwood
Alejandro Rey
Shelley Morrison

The Saint (1962-1969)
Action
http://www.imdb.com/title/tt0055701/
Roger Moore
Ivor Dean
Leslie Crawford
Justine Lord

```
% ./a.out
series:  |JAG (1995-2005)|
   actor:  |Action|
   actor:  |http://www.imdb.com/title/tt0112022/|
   actor:  |David James Elliott|
   actor:  |Patrick Labyorteaux|
   actor:  |Catherine Bell|
   actor:  |John M. Jackson|
   actor:  |Karri Turner|
series:  ||
series:  |Matlock (1986-1995)|
   actor:  |Mystery|
   actor:  |http://www.imdb.com/title/tt0090481/|
   actor:  |Andy Griffith|
   actor:  |Nancy Stafford|
   actor:  |Julie Sommars|
   actor:  |Clarence Gilyard Jr.|
   actor:  |Kene Holliday|

    ...

series:  |The Saint (1962-1969)|
   actor:  |Action|
   actor:  |http://www.imdb.com/title/tt0055701/|
   actor:  |Roger Moore|
   actor:  |Ivor Dean|
   actor:  |Leslie Crawford|
   actor:  |Justine Lord|
Number of TV series: 6
```

Issues:

- Year is part of series name.
- No distinction between category, URL, or actor name.
- Extra blank lines not handled properly.
- No actor list.
- No tree.

Add functions to correct these issues.

1.3 Version 2: ReadTVFile

```
/* tvDB2.cpp

   Test code for TV/actor tree programming assignment.

   c++ tvDB2.cpp stringRoutines.cpp

   Bruce M. Bolden
   April 13, 2014
 */

#include <fstream>
#include <iostream>
#include <string.h>
#include <stdlib.h>

#include "stringRoutines.h"

using namespace std;

  // Prototypes
int ReadTVData();

void GetSeriesName( char line[], char seriesName[] );
void GetYears( char line[], int & yStart, int & yEnd );

int main()
{
   int nSeries = 0;

   nSeries = ReadTVData();
   cout << "Number of TV series: " << nSeries << endl;

   return EXIT_SUCCESS;
}
```

```
int ReadTVFile()
{
   ifstream fIn( "tvDB_Test.txt", ios::in );
   if( !fIn )
   {
      cout << "Unable to open \"tvDB\" data file" << endl;
      exit( -1 );
   }

   const int MAX_LINE = 128;
   char line[MAX_LINE];
   char seriesName[MAX_LINE];
   char seriesCategory[MAX_LINE/2];
   char seriesURL[MAX_LINE];
   char actorName[MAX_LINE/2];

   int  yStart, yEnd;

   //StringList actors;

   int nSeries = 0;
```

```
    while( fIn.getline( line, MAX_LINE ) )
    {
        GetSeriesName( line, seriesName );
        GetYears( line, yStart, yEnd );

        fIn.getline( seriesCategory, MAX_LINE/2 );
        fIn.getline( seriesURL, MAX_LINE );

        cout << "series name:  |" << seriesName << "|" << endl;
        cout << "   year start: " << yStart << endl;
        cout << "   year end:   " << yEnd   << endl;
        cout << "   Category:  |" << seriesCategory << "|" << endl;
        cout << "   URL:       |" << seriesURL << "|" << endl;

        fIn.getline( line, MAX_LINE/2 );
        while( strlen(line) > 0 )
        {
            strcpy( actorName, line );
            cout << "   actor name:  |" << actorName << "|" << endl;
            //actors.Add( actorName );

            fIn.getline( line, MAX_LINE/2 );
        }

        nSeries++;
        //tree.AddSeries( seriesName, yStart, yEnd, actors );
    }

    fIn.close();

    return nSeries;
}
```

```
% ./a.out
Entering GetYears()
   yearStart: 4
   yearEnd:   16
   yearString: 1995-2005
   tmpString: 1995
   tmpString: 2005
   year start: 1995
   year end:   2005
Leaving GetYears()
series name:  |JAG|
   year start: 1995
   year end:   2005
   Category:  |Action|
   URL:       |http://www.imdb.com/title/tt0112022/|
Entering GetYears()
   yearStart: 8
   yearEnd:   20
   yearString: 1986-1995
   tmpString: 1986
   tmpString: 1995
   year start: 1986
   year end:   1995
Leaving GetYears()
series name:  |Matlock|
   year start: 1986
   year end:   1995
   Category:  |Mystery|
   URL:       |http://www.imdb.com/title/tt0090481/|

   ...

series name:  |The Saint|
   year start: 1962
   year end:   1969
   Category:  |Action|
   URL:       |http://www.imdb.com/title/tt0055701/|
Number of TV series: 5
```

Issues:

- Still no actor list.
- Still no tree.

Add functions to correct these issues.

1.4 GetSeriesName()

```
void GetSeriesName( char line[], char seriesName[] )
{
   int yearStart;

   yearStart = IndexOf( line, '(' );
   GetSubString( line, 0, yearStart-1, seriesName );
}
```

1.5 GetYears()

```
void GetYears( char line[], int & yStart, int & yEnd )
{
    char tmpString[8];
    char yearString[16];
    int yearStart, yearEnd;

    cout << "Entering GetYears()" << endl;

    yearStart = IndexOf( line, '(' );
    yearEnd   = IndexOf( line, ')' );
    cout << "   yearStart: " << yearStart << endl;
    cout << "   yearEnd:   " << yearEnd   << endl;

    GetSubString( line, yearStart+1, yearEnd-yearStart-1, yearString
    //GetSubString( line, yearStart+1, 11, yearString );
    cout << "   yearString: " << yearString << endl;

    GetSubString( yearString, 0, 4, tmpString );
    cout << "   tmpString: " << tmpString << endl;
    yStart = atoi( tmpString );

    GetSubString( yearString, 7, 4, tmpString );  // 7? not '-'!
    cout << "   tmpString: " << tmpString << endl;
    yEnd = atoi( tmpString );

    cout << "   year start: " << yStart << endl;
    cout << "   year end:   " << yEnd   << endl;

    cout << "Leaving GetYears()" << endl;
}
```

1.6 IndexOf()

```
int IndexOf( char s[], char c )
{
    int i = 0;

    while( s[i] != '\0' && s[i] != c )
       i++;

    return i;
}
```

1.7 GetSubString()

```
void GetSubString( char s[], int start, int end, char res[] )
{
    int i;
    int iRes = 0;    //  index of result string

    //cerr << " In GetSubstring: " << res << endl;
    //cerr << "    start: " << start << endl;
    //cerr << "    end:   " << start+end << endl;

    for( i = start ; i < start+end ; i++ )
    {
       res[iRes++] = s[i];
    }

    res[iRes] = '\0';
    RemoveBlanksFromString( res );

    //cerr << "    Substring: " << res << endl;
    //cerr << " Leaving  GetSubstring: " << res << endl;
}
```

1.8 RemoveBlanksFromString()

```
void RemoveBlanksFromString( char s[] )
{
    int sLen = strlen( s );

    for( int i = sLen ; i >= 0 ; i-- )
    {
        if( isalpha(s[i]) )
           break;
        if( s[i] == ' ' )
           s[i] = '\0';
    }
}
```

1.9 Multiple Blank Lines

Issue:

- Doesn't handle multiple blank lines in the input file.

Add code to correct this issue.

1.9.1 Failed attempt!

```
while( fIn.getline( line, MAX_LINE ) )
{
    GetSeriesName( line, seriesName );

    // other code...

    //  Handle multiple blank lines
    while( strlen(line) == 0 )
    {
       cout << "  blank line!" << endl;
       fIn.getline( line, MAX_LINE );
    }

       // Push back non-blank line
    for( int i = 0 ; i < strlen(line) ; i++ )
       fIn.unget();
       //fIn.putback( line[i] );


    nSeries++;
    //tree.AddSeries( seriesName, yEnd-yStart, actors );
}
```

1.9.2 Better method

```
while( fIn.getline( line, MAX_LINE ) )
{
    if( strlen(line) == 0 )
       continue;

    GetSeriesName( line, seriesName );

    // other code...

    /* Handle multiple blank lines
    while( strlen(line) == 0 )
    {
       cout << "  blank line!" << endl;
       fIn.getline( line, MAX_LINE );
    }

       // Push back non-blank line
    for( int i = 0 ; i < strlen(line) ; i++ )
       fIn.unget();
       //fIn.putback( line[i] );
    */

    nSeries++;
    //tree.AddSeries( seriesName, yEnd-yStart, actors );
}
```

Issues:

- None: ready to construct tree.

```
/*
  Build:  c++ tmdb.cpp llcstr.cpp filmstruct.cpp
*/

#include<iostream>
#include<cstring>
#include<sstream>
#include<fstream>
using namespace std;

#include"llcstr.h"
#include"llcstr.cpp"
#include"filmstruct.cpp"

int main(){

  char test1[] = "a is best", test2[]="be is better", test3[]= "c is

  //Checking strcmp behavior
  cout << endl << "String test: " << endl;
  if (strcmp(test1, test2) > 0)
    cout << test1;
  else if (strcmp(test2, test1) > 0)
    cout << test2;
  else
    cout << "Broken";
  cout << endl;

  //Setting up test film
  film fow;
  fow.title = "Freaking Oscar Winner";
  fow.run_start = 2015;
  fow.run_end = 2016;
  fow.genre = "AWESOME";
  fow.imdb = "tmdb.com";
```

```
fow.cast.PrintNodes();
fow.cast.AddNode(test1);
fow.cast.PrintNodes();
fow.cast.AddNode(test2);
fow.cast.PrintNodes();
fow.cast.AddNode(test3);
fow.cast.PrintNodes();
fow.cast.PrintNodes();
cout << "FOW has a cast of : " << fow.cast.Size() << endl;

cout << endl << endl;
// PrintFilmTitle(fow);

cout << "Adding a fourth movie" << endl;
char test4[]= "d is new";
fow.cast.AddNode(test4);
cout << endl << endl;

cout << "FOW has a cast of : " << fow.cast.Size() << endl;


cout << "Is the cast empty??" << endl;
if (fow.cast.IsEmpty()) cout << "yes" << endl;
else cout << "no" << endl;

cout << "Is \"" << test3 << "\" in the cast??" << endl;
if (fow.cast.InList(test3)) cout << "yes" << endl;
else cout << "no" << endl;

cout << "Made it to this point!" << endl;

fow.cast.PrintNodes();

cout << "SUCCESS!" << endl;
```

```
  PrintFilmCast(fow);
  cout << endl << endl;
  PrintFilmVerbose(fow);

  cout << endl << "STUFF GOES HERE" << endl;
}
```