|  |  |
| --- | --- |
| **Author Identification Block** | |
| **Author:** | Chris Graff |
| **Student ID:** | \*20274911 |
| **E-Mail:** | [cgraff@uco.edu](mailto:cgraff@uco.edu) |
| **Course:** | CMSC 2613 – Programming 2 |
| **CRN:** | 21641, Spring 2012 |
| **Project:** | p06 |
| **Due:** | March 2, 2012 |
| **Account:** | tt025 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Scoring Block** | | | |
| **Component** | **Available** | **Earned** | **Explanation** |
| Compilation |  |  |  |
| Submission Instructions | 2 |  |  |
| Author Identification | 1 |  |  |
| Modularity | 3 |  |  |
| Command Line | 3 |  |  |
| Input file | 3 |  |  |
| Output file | 3 |  |  |
| Execution | 10 |  |  |
| **Total** | **25** |  |  |

#include <cstdlib>

#include <iostream>

#include <math.h>

#include <iomanip>

#include <fstream>

#include <string>

#include <string.h>

#include <ios>

#include <limits.h>

#include "List06.h"

#include "Set06.h"

using namespace std;

//----------------------------------------------------------------

//Author: Chris Graff

//StudentID#: \*20274911

//Email: cgraff@uco.edu

//Course: CMSC2613 Programming II

//CRN: 21641, Spring 2012

//Project: p06

//Due: March 2nd, 2012

//Account: tt025

//----------------------------------------------------------------

struct CommandLineException

{

CommandLineException (int max, int actual)

{

cout <<endl <<"Too many command line arguements." <<endl;

cout <<"A maximum of " <<max <<" arguements are permitted." <<endl;

cout <<actual <<" arguements were entered." <<endl;

}

};

struct FileException

{

FileException (char\* filename)

{

cout <<endl <<"File " <<filename <<" could not be opened or doesn't exist" <<endl;

}

};

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

{

try

{

char iFileName1[255], iFileName2[255], iFileName3[255], oFileName[255];

switch (argc)

{

case 1:

cout <<"Enter the input file name 1:";

cin >> iFileName1;

cout <<"Enter the input file name 2:";

cin >> iFileName2;

cout <<"Enter the input file name 3:";

cin >> iFileName3;

cout <<"Enter the output file name:";

cin >> oFileName;

break;

case 2:

strcpy(iFileName1, argv[1]);

cout <<"Enter the input file name 2:";

cin >> iFileName2;

cout <<"Enter the input file name 3:";

cin >> iFileName3;

cout <<"Enter the output file name:";

cin >> oFileName;

break;

case 3:

strcpy(iFileName1, argv[1]);

strcpy(iFileName2, argv[2]);

cout <<"Enter the input file name 3:";

cin >> iFileName3;

cout <<"Enter the output file name:";

cin >> oFileName;

break;

case 4:

strcpy(iFileName1, argv[1]);

strcpy(iFileName2, argv[2]);

strcpy(iFileName3, argv[3]);

cout <<"Enter the output file name:";

cin >> oFileName;

break;

case 5:

strcpy(iFileName1, argv[1]);

strcpy(iFileName2, argv[2]);

strcpy(iFileName3, argv[3]);

strcpy(oFileName, argv[4]);

break;

default:

throw CommandLineException (4, argc-1);

break;

}

ifstream i1(iFileName1);

if (!i1)

throw FileException(iFileName1);

ifstream i2(iFileName2);

if (!i2)

throw FileException(iFileName2);

ifstream i3(iFileName3);

if (!i3)

throw FileException(iFileName3);

ofstream o(oFileName);

if (!o)

throw FileException(oFileName);

Set<int> S1(INT\_MAX, i1);

Set<int> S2(INT\_MAX, i2);

Set<int> S3(INT\_MAX, i3);

Set<int> I(INT\_MAX);

I.Intersection(S1, S2);

Set<int> U(INT\_MAX);

U.Union(S2, S3);

Set<int> D(INT\_MAX);

D.Difference(U, I);

S1.print(o, "set 1=");

S2.print(o, "set 2=");

S3.print(o, "set 3=");

I.print(o, "set I=");

U.print(o, "set U=");

D.print(o, "set D=");

i1.close();

i2.close();

i3.close();

o.close();

}

catch (...)

{

cout <<"Program terminated." <<endl;

exit(EXIT\_FAILURE);

}

}

#ifndef List06\_h

#define List06\_h

#include<iostream>

#include<fstream>

#include<iomanip>

#include<limits>

//----------------------------------------------------------------

//Author: Chris Graff

//StudentID#: \*20274911

//Email: cgraff@uco.edu

//Course: CMSC2613 Programming II

//CRN: 21641, Spring 2012

//Project: p06

//Due: March 2nd, 2012

//Account: tt025

//----------------------------------------------------------------

using namespace std;

template <class T>

class List

{

struct Element

{

Element\* smaller;

T key;

Element\* larger;

Element(Element\* s, T k, Element\* l): smaller(s), key(k), larger(l){}

};

Element\* largest;

Element\* cursor;

void Kill(Element\* e)

{

while(e->key!=MAX)

{

Element\* p=e;

e=e->larger;

delete p;

}

}

void Sentinel()

{

Element\* n = new Element(0, MAX, 0);

cursor=largest=n->smaller=n->larger=n;

}

protected:

T MAX;

public:

List(T m):MAX(m){Sentinel();}

List(T m, istream& i):MAX(m){Sentinel();scan(i);}

~List(){Kill(largest->larger); delete largest;}

void Empty(){Kill(largest->larger); delete largest; Sentinel();}

void insert(T k)

{

Element\* e=largest->larger;

while(k>e->key) e=e->larger;

if(k == e->key) return;

Element\* n=new Element(e->smaller,k,e);

e->smaller->larger=n;

e->smaller=n;

}

void Remove(T k)

{

Element\* e=largest->larger;

while(k > e->key)e=e->larger;

if(k!=e->key) return;

e->smaller->larger = e->larger;

e->larger->smaller = e->smaller;

delete e;

}

void print(ostream& o, const char\* title)

{

o<<endl<<title<<"{";

Element\* e=largest->larger;

for(int a=0;e->key!=MAX;a++)

{

if(a>0) o<<",";

o<<e->key;

e=e->larger;

}

o<<"}"<<endl;

}

void scan(istream& i)

{

for(;;)

{

T k;

i >> k;

if(i.eof()) break;

insert(k);

}

}

void First(){cursor=largest->larger;}

void Next(){if(cursor!=largest) cursor=cursor->larger;}

bool isEol(){return cursor==largest;}

T Key(){if(cursor) return cursor->key;}

bool isMember(T k)

{

Element\* e=largest->larger;

while(k > e->key) e=e->larger;

return k==e->key;

}

T Member(){if(cursor) return cursor->key;}

};

#endif

#ifndef Set06\_h

#define Set06\_h

#include <cstdlib>

#include <fstream>

#include <iostream>

#include <limits>

#include "List06.h"

//----------------------------------------------------------------

//Author: Chris Graff

//StudentID#: \*20274911

//Email: cgraff@uco.edu

//Course: CMSC2613 Programming II

//CRN: 21641, Spring 2012

//Project: p06

//Due: March 2nd, 2012

//Account: tt025

//----------------------------------------------------------------

using namespace std;

template<class T>

class Set:public List<T>

{

public:

Set(T m):List<T>(m){}

Set(T m, istream& i):List<T>(m, i){}

void Union(Set& s1, Set& s2)

{

List<T>::Empty();

for(s1.First();!s1.isEol();s1.Next()){insert(s1.Key());}

for(s2.First();!s2.isEol();s2.Next()){insert(s2.Key());}

}

void Intersection(Set& s1, Set& s2)

{

List<T>::Empty();

for(s1.First();s1.Key()!=List<T>::MAX;s1.Next())

{

if(s2.isMember(s1.Key()))

{

insert(s1.Key());

}

}

}

void Difference(Set& M, Set& S)

{

List<T>::Empty();

for(M.First();!M.isEol();M.Next()) insert(M.Key());

Set<T> I(List<T>::MAX);

I.Intersection(M,S);

for(I.First();!I.isEol();I.Next()) Remove(I.Key());

}

};

#endif