|  |  |
| --- | --- |
| **Author Identification Block** | |
| **Author:** | Mr. Lynn Barnett |
| **Student ID:** | \*20360727 |
| **E-Mail:** | [barnettlynn@gmail.com](mailto:aturing@uco.edu) |
| **Course:** | CMSC 2613 – Programming II |
| **CRN:** | 21256, Spring, 2014 |
| **Project:** | p03 |
| **Due:** | February 14, 2014 |
| **Account:** | tt044 |

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

File p03.cpp

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

// Author: Mr. Lynn Barnett

// Student ID: \*20360727

// E-Mail: barnettlynn@gmail.com

// Course: CMSC 2613, Programming II

// CRN: 21256, Spring, 2014

// Project: p03

// Due: February 14, 2014

// Account: tt044

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

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

// Files:

// p03.cpp \*\*\*

// Stack03.h

// Stack03.cpp

// Scan03.h

// Scan03.l

// p03make

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

#include <cstdio>

#include <cstring>

#include "Stack03.h"

#include "Scan03.h"

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

// EXCEPTION MESSAGES

struct CommandLineException

{

CommandLineException(int max, int actual){

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

cout << "A Maximum of " << max << " arguments are permited" << endl;

cout << actual << " arguments were given" << endl;

}

};

struct FileException

{

FileException(const char\* fn)

{

cout << "File " << fn << " could not be opened" << endl;

}

};

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

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

//POSTFIX MANAGER

void post\_fix\_mngr(FILE\* input\_stream, ofstream& output\_stream)

{

Scan L(input\_stream);

Stack03 post\_fix\_stack;

int subtraction\_holder;

int division\_holder;

while(true){

int t = L.Lex();

if (t == 0) break;

switch(t)

{

case INTLIT:

post\_fix\_stack.Push(L.Intlit());

break;

case PLUS:

post\_fix\_stack.Push(post\_fix\_stack.Pop() + post\_fix\_stack.Pop());

break;

case MINUS:

subtraction\_holder = post\_fix\_stack.Pop();

post\_fix\_stack.Push(post\_fix\_stack.Pop() - subtraction\_holder);

break;

case STAR:

post\_fix\_stack.Push(post\_fix\_stack.Pop() \* post\_fix\_stack.Pop());

break;

case SLASH:

division\_holder = post\_fix\_stack.Pop();

post\_fix\_stack.Push(post\_fix\_stack.Pop()/division\_holder);

break;

default:

{

break;

}

}

}

post\_fix\_stack.Print(output\_stream);

}

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

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

//MAIN

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

{

try

{

char input\_file\_name[255], output\_file\_name[255];

switch(argc)

{

case 1:

cout << "Please Enter The Input File Name" << endl;

cout << ": ";

cin >> input\_file\_name;

cout << "Please Enter The Output File Name" << endl;

cout << ": ";

cin >> output\_file\_name;

break;

case 2:

strcpy(input\_file\_name, argv[1]);

cout << "Please Enter The Output File Name" << endl;

cout << ": ";

cin >> output\_file\_name;

break;

case 3:

strcpy(input\_file\_name, argv[1]);

strcpy(output\_file\_name, argv[2]);

break;

default:

throw CommandLineException(2, argc-1);

break;

}

FILE\* input\_stream = fopen(input\_file\_name,"r");

if(!input\_stream) throw FileException(input\_file\_name);

ofstream output\_stream (output\_file\_name);

if(!output\_stream) throw FileException(output\_file\_name);

post\_fix\_mngr(input\_stream, output\_stream);

output\_stream.close();

fclose(input\_stream);

} catch(...)

{

cout << "Something has gone wrong!!!" << endl;

exit(EXIT\_FAILURE);

}

return 0;

}//MAIN

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

{

try

{

char input\_file\_name[255], output\_file\_name[255];

switch(argc)

{

case 1:

cout << "Please Enter The Input File Name" << endl;

cout << ": ";

cin >> input\_file\_name;

cout << "Please Enter The Output File Name" << endl;

cout << ": ";

cin >> output\_file\_name;

break;

case 2:

strcpy(input\_file\_name, argv[1]);

cout << "Please Enter The Output File Name" << endl;

cout << ": ";

cin >> output\_file\_name;

break;

case 3:

strcpy(input\_file\_name, argv[1]);

strcpy(output\_file\_name, argv[2]);

break;

default:

throw CommandLineException(2, argc-1);

break;

}

FILE\* input\_stream = fopen(input\_file\_name,"r");

if(!input\_stream) throw FileException(input\_file\_name);

ofstream output\_stream (output\_file\_name);

if(!output\_stream) throw FileException(output\_file\_name);

post\_fix\_mngr(input\_stream, output\_stream);

output\_stream.close();

fclose(input\_stream);

} catch(...)

{

cout << "Something has gone wrong!!!" << endl;

exit(EXIT\_FAILURE);

}

return 0;

}

File Stack03.h

#ifndef Stack03\_h

#define Stack03\_h

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

// Author: Mr. Lynn Barnett

// Student ID: \*20360727

// E-Mail: barnettlynn@gmail.com

// Course: CMSC 2613, Programming II

// CRN: 21256, Spring, 2014

// Project: p03

// Due: February 14, 2014

// Account: tt044

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

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

// Files:

// p03.cpp

// Stack03.h \*\*\*

// Stack03.cpp

// Scan03.h

// Scan03.l

// p03make

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

#include <fstream> // for ofstream

class Stack03{

struct Element{

Element\* prev;

int value;

Element(Element\*p, int v):prev(p), value(v){}

};

Element\* tos;

void Kill(Element\* e);

public:

Stack03(void);

~Stack03(void);

bool isFull(void);

bool isEmpty(void);

void Push(int t);

int Pop(void);

void Print(std::ofstream& output\_stream);

};

#endif

File Stack03.cpp

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

// Author: Mr. Lynn Barnett

// Student ID: \*20360727

// E-Mail: barnettlynn@gmail.com

// Course: CMSC 2613, Programming II

// CRN: 21256, Spring, 2014

// Project: p03

// Due: February 14, 2014

// Account: tt044

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

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

// Files:

// p03.cpp

// Stack03.h

// Stack03.cpp \*\*\*

// Scan03.h

// Scan03.l

// p03make

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

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

//INCLUDES

#include "Stack03.h"

#include <iostream>

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

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

//EXCEPTIONS

struct StackException {

StackException(const char\* m)

{

std::cout << std::endl;

std::cout << "I am the Stack and I am" << m << ".";

std::cout << std::endl;

}

};

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

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

// CONSTRUCTOR AND DESTRUCTOR

Stack03::Stack03():tos(0){};

Stack03::~Stack03()

{

Kill(tos);

};

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

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

//IS FULL, AND IS EMPTY METHODS

bool Stack03::isFull(void)

{

return false;

};

bool Stack03::isEmpty(void)

{

return tos == 0;

};

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

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

//PUSH AND POP

void Stack03::Push(int t)

{

if(isFull()) throw StackException("full");

Element\* n = new Element(tos, t);

tos=n;

};

int Stack03::Pop(void)

{

if(isEmpty()) throw StackException("empty");

Element\* p = tos;

int t = p->value;

tos = p->prev;

delete p;

return t;

};

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

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

//PRINT

void Stack03::Print(std::ofstream& output\_stream)

{

Element\* iterator = tos;

int i = 0;

while(true)

{

if(i == 0)

{

output\_stream << "Stack[tos]=" << iterator->value << std::endl;

} else{

output\_stream << "Stack[tos-" << i << "]=" << iterator->value << std::endl;

}

iterator = iterator->prev;

if(iterator == 0) break;

i++;

}

};

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

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

//KILL, CALLED BY THE DESTRUCTOR TO DELETE ALL THE ELEMENTS OF THE LIST

void Stack03::Kill(Element\* e)

{

while(e)

{

Element\* p = e;

e = e->prev;

delete p;

}

};

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

File Scan03.h

#ifndef Scan03\_h

#define Scan03\_h 1

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

// File: Scan03.h

// Description:

// Recognizes integers and arithmetic operators for project 3 in

// Programming II.

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

// Author: Thomas R. Turner

// E-Mail: trturner.ucok.edu

// Date: September, 2002

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

// Copyright September, 2002 by Thomas R. Turner

// Do not reproduce without permission from Thomas R. Turner.

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

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

// Standard C and C++ include files

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

#include <cstdio>

#include <fstream>

#include <iostream>

using namespace std;

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

//Token code definitions

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

#define INTLIT 1

#define PLUS 2

#define MINUS 3

#define STAR 4

#define SLASH 5

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

//Function: yylex

//Function yylex is the Scanner. Function yylex returns an integer

//token code as defined above or 0 if end-of-file has been

//reached.

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

#ifdef \_\_cplusplus

extern "C"

#endif

int yylex (void);

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

//Class Scan defines the attributes of a Scanner

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

class Scan {

int tokencode; //Code for the most recent token found

public:

Scan(FILE\* i); //Redirect the input source from the

//keyboard to input file i.

int Lex(void); //Call the scanner yylex and return the code

//found by yylex

int FetchTokenCode(void); //Return the code of the most recent token

void StoreTokenCode(int T); //Store the token code.

char\* FetchSpelling(void); //Return the spelling of the most recent

//token

int Intlit(void); //Return the most recent integer literal

};

#endif

File Scan03.l

%{

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

// File: Scan03.l

// Description:

// Contains the most elementary example use of lex for the purpose of

// building a scanner.

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

// Author: Thomas R. Turner

// E-Mail: trturner@ucok.edu

// Date: September, 2002

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

//Copyright September, 2002 by Thomas R. Turner.

//Do not reproduce without permission from Thomas R. Turner

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

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

// Standard C and C++ Library Include Files

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

#include <cstdio>

#include <iostream>

#include <fstream>

#include <iomanip>

using namespace std;

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

// Application Includes

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

#include "Scan03.h"

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

//Function prototypes

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

int TokenMgr(int T);

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

//Global Variables

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

%}

%%

[ \t\n]+ ;

[+-]?[0-9]+ {

return(TokenMgr(INTLIT));

}

"+" {

return(TokenMgr(PLUS));

}

"-" {

return(TokenMgr(MINUS));

}

"\*" {

return(TokenMgr(STAR));

}

"/" {

return(TokenMgr(SLASH));

}

%%

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

int TokenMgr(int T)

{ return T;

}

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

//Class Scan implementation

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

//Constructor Scan is used to redirect the input file stream from the

//keyboard to input file stream i.

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

Scan::Scan(FILE\* i)

{ yyin=i;

}

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

//Function Lex calls yylex

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

int Scan::Lex(void)

{ return tokencode=yylex();

}

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

//Function FetchSpelling returns a pointer to the spelling of the most

//recent token.

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

char\* Scan::FetchSpelling(void)

{ return (char\*)yytext;

}

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

//Function FetchTokenCode returns the code of the most recent token

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

int Scan::FetchTokenCode(void)

{ return tokencode;

}

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

//Function StoreTokenCode records the most recent token code

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

void Scan::StoreTokenCode(int T)

{ tokencode=T;

}

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

//Function Intlit returns the most recent integer literal

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

int Scan::Intlit(void)

{ int i;

sscanf(yytext,"%d",&i);

return i;

}

//-----------------------End of Lex Definition------------------------

File p03make

#-------------------------------------------------------------------------

# File p03make creates executable file p03.

#-------------------------------------------------------------------------

# Author: Thomas R. Turner

# E-Mail: tturner@ucok.edu

# Date: September, 2002

#-------------------------------------------------------------------------

#object files

#-------------------------------------------------------------------------

obj = p03.o Scan03.o Stack03.o

#-------------------------------------------------------------------------

# Bind p03.o, Scan03.o and Stack03.o

#-------------------------------------------------------------------------

p03: ${obj}

g++ -o p03 ${obj} -ll

#-------------------------------------------------------------------------

# Compile p03.cpp

#-------------------------------------------------------------------------

p03.o: p03.cpp Scan03.h Stack03.h

g++ -g -c p03.cpp

#-------------------------------------------------------------------------

# Compile Scan03.l. First translate the lex specification, then compile

#-------------------------------------------------------------------------

Scan03.o: Scan03.cpp Scan03.h

g++ -g -c Scan03.cpp

Scan03.cpp: Scan03.l Scan03.h

lex Scan03.l

mv lex.yy.c Scan03.cpp

#-------------------------------------------------------------------------

# Compile Stack03.cpp

#-------------------------------------------------------------------------

Stack03.o: Stack03.cpp Stack03.h

g++ -g -c Stack03.cpp