**D:\GIT\CPP\source\shape.cpp**

**//**

**// shape.cpp**

**// OOD Assignment Sample**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#include <iostream>**

**#include <fstream>**

**#include <exception>**

**using namespace std;**

**#include "window.h"**

**#include "ParseInput.h"**

**#include "Exceptions.h"**

**ParseInput runProgram;**

**void draw(void)**

**{**

**runProgram.Draw();**

**}**

**// Main Function brings all the components of the solution together**

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

**{**

**ifstream fs;**

**char i[100];**

**// Checks that an input file has been provided**

**if (argc != 2)**

**{**

**cerr << "Correct Usage: shape <filename>" << endl << "Press any key to exit...." << endl;**

**cin >> i;**

**exit(1);**

**}**

**fs.open(argv[1]);**

**// Checks if the file opened successfully**

**if (!fs.is\_open())**

**{**

**cerr << "File Not Found!" << endl << "Press any key to exit...." << endl;**

**cin >> i;**

**exit(1);**

**}**

**// Attempts to run the program, catching any exceptions raised in the process**

**try**

**{**

**fs >> runProgram;**

**}**

**catch (FormatException& e)**

**{**

**cerr << e.what() << endl;**

**}**

**catch (BracketsError2& e)**

**{**

**cerr << e.what() << endl;**

**}**

**catch (BracketsError& e)**

**{**

**cerr << e.what() << endl ;**

**}**

**// Closes the input file**

**fs.close();**

**// Displays output**

**window w(argc,argv);**

**}**

**D:\GIT\CPP\source\Exceptions.h**

**//**

**// Exceptions.h**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#include <string>**

**#include <exception>**

**#include <sstream>**

**#include <iostream>**

**using namespace std;**

**// Exception for catching incorrect sizes or commands such as 'FORWRAD 1' or 'FORWARD a'**

**class FormatException:public exception{**

**public:**

**FormatException(string m):msg("ERROR: Unrecognised instruction " + m){}**

**FormatException():msg("ERROR: Incorrect command size parameter"){}**

**~FormatException() throw(){};**

**const char\* what(){return msg.c\_str();}**

**private:**

**string msg;**

**};**

**// Exception for a repeat loop that isn't ended (missing ']' character)**

**class BracketsError2:public exception{**

**public:**

**BracketsError2():msg("ERROR: Unterminated Repeat Loop (missing ])"){}**

**~BracketsError2() throw(){};**

**const char\* what(){return msg.c\_str();}**

**private:**

**string msg;**

**};**

**// Exception for a repeat loop that isn't started (missing '[' character)**

**class BracketsError:public exception{**

**public:**

**BracketsError():msg("ERROR: Repeat Loop not started (missing [)"){}**

**~BracketsError() throw(){};**

**const char\* what(){return msg.c\_str();}**

**private:**

**string msg;**

**};**

**D:\GIT\CPP\source\ParseInput.h**

**//**

**// ParseInput.h**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#ifndef CPP\_ParseInput\_h**

**#define CPP\_ParseInput\_h**

**#include <iostream>**

**#include <vector>**

**#include <fstream>**

**#include <string>**

**#include "Instruction.h"**

**using namespace std;**

**// This class reads in the input program from the command line**

**class ParseInput**

**{**

**public:**

**ParseInput(); // Constructor**

**~ParseInput(); // Destructor**

**void Draw(); // Virtual Draw function**

**void SetBrackets(bool b){brackets = b;} // Allows write access to brackets member variable**

**friend ifstream& operator>>(ifstream& is, ParseInput& pi); // Reads the input program**

**private:**

**std::vector<Instruction \*> CommandList; // Vector type allows dynamic growth of the command list**

**bool brackets; // Flag to indicate if enclosing brackets are balanced.**

**};**

**#endif**

**D:\GIT\CPP\source\ParseInput.cpp**

**//**

**// ParseInput.cpp**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#include <iostream>**

**#include <fstream>**

**#include <string>**

**#include <exception>**

**#include "Exceptions.h"**

**#include "Forward.h"**

**#include "Jump.h"**

**#include "Repeat.h"**

**#include "Rotate.h"**

**using namespace std;**

**// Default Constructor**

**ParseInput::ParseInput()**

**{**

**brackets = true;**

**}**

**// Default Destructor**

**ParseInput::~ParseInput()**

**{**

**CommandList.clear();**

**}**

**// Virtual Draw function**

**void ParseInput::Draw()**

**{**

**// Iterates through the command list vector of Instruction classes and executes their Draw() method**

**for (std::vector<Instruction\*>::iterator it=CommandList.begin(); it != CommandList.end(); it++) {**

**(\*it)->Draw();**

**}**

**}**

**// Reads the input file and stores as a ParseInput class**

**ifstream& operator>>(ifstream& is, ParseInput& pi)**

**{**

**string s;**

**double sz;**

**// Loops as long as the input string isn't an end of file indicator**

**while (!is.eof())**

**{**

**is >> s;**

**// Instantiates a new class depending on which Instruction keyword is read in**

**// If the keyword is not recognised then a FormatException is thrown.**

**if (s == "FORWARD")**

**{**

**if (is >> sz)**

**{**

**Forward \*p\_f = new Forward(sz);**

**Instruction \*i1 = p\_f;**

**pi.CommandList.push\_back(i1);**

**}**

**else**

**{**

**throw FormatException();**

**}**

**}**

**else if (s == "JUMP")**

**{**

**if (is >> sz)**

**{**

**Jump \*p\_j = new Jump(sz);**

**Instruction \*i2 = p\_j;**

**pi.CommandList.push\_back(i2);**

**}**

**else**

**{**

**throw FormatException();**

**}**

**}**

**else if (s == "LEFT")**

**{**

**if (is >> sz)**

**{**

**Rotate \*p\_l = new Rotate(sz);**

**Instruction \*i3 = p\_l;**

**pi.CommandList.push\_back(i3);**

**}**

**else**

**{**

**throw FormatException();**

**}**

**}**

**else if (s == "RIGHT")**

**{**

**if (is >> sz)**

**{**

**Rotate \*p\_r = new Rotate(-sz);**

**Instruction \*i4 = p\_r;**

**pi.CommandList.push\_back(i4);**

**}**

**else**

**{**

**throw FormatException();**

**}**

**}**

**else if (s == "]")**

**{**

**pi.brackets = true;**

**is >> ws;**

**return is;**

**}**

**else if (s == "REPEAT")**

**{**

**if (is >> sz)**

**{**

**Repeat \*p\_rp = new Repeat(sz);**

**(\*p\_rp).SetBrackets(false);**

**is >> \*p\_rp;**

**Instruction \*i2 = p\_rp;**

**pi.CommandList.push\_back(i2);**

**}**

**else**

**{**

**throw FormatException();**

**}**

**}**

**else**

**{**

**throw FormatException(s);**

**}**

**is >> ws;**

**}**

**// If the brackets are not balanced at the end of the input, an exception is thrown**

**if (pi.brackets == false)**

**{**

**throw BracketsError2();**

**}**

**return is;**

**}**

**D:\GIT\CPP\source\Instruction.h**

**//**

**// Instruction.h**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#ifndef CPP\_Instruction\_h**

**#define CPP\_Instruction\_h**

**#include <iostream>**

**// Base class for the Instructions**

**// All of the instruction classes - Jump, Repeat, Rotate and Forward**

**// Publically inherit from this class**

**class Instruction**

**{**

**public:**

**Instruction(){} // Default Constructor**

**Instruction(double size); // Constructor**

**~Instruction(){} // Destructor**

**float GetSize(); // public interface to read the size member variable**

**// Pure virtual Draw function - must be defined in each derived class**

**// Provides run-time polymorphism, allowing the program to handle any input sequence**

**virtual void Draw() = 0;**

**protected:**

**// size is the parameter of the instruction i.e. -**

**// degrees to rotate, number of times to repeat, distance to draw or jump**

**double size;**

**};**

**#endif**

**D:\GIT\CPP\source\Forward.h**

**//**

**// Forward.h**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#ifndef CPP\_Forward\_h**

**#define CPP\_Forward\_h**

**#include "Instruction.h"**

**// Inherits publically from Instruction**

**class Forward: public Instruction**

**{**

**public:**

**Forward(double s); // Constructor**

**void Draw(); // Virtual Draw function**

**~Forward(){} // Destructor**

**private:**

**Forward() {} // Default Constructor**

**};**

**#endif**

**D:\GIT\CPP\source\Forward.cpp**

**//**

**// Forward.cpp**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#include "Forward.h"**

**#ifdef \_WIN32**

**#include<Windows.h>**

**#include <GL/glut.h> // The GL Utility Toolkit (Glut) Header**

**#endif**

**#ifdef \_\_APPLE\_\_**

**#include <GLUT/glut.h>**

**#endif**

**#ifdef \_\_linux\_\_**

**#include <GL/glut.h>**

**#endif**

**// Constructor**

**Forward::Forward(double s)**

**{**

**//size is a member variable of the instruction base class**

**size = s;**

**}**

**void Forward::Draw()**

**{**

**// Start drawing a line**

**glBegin(GL\_LINE\_STRIP);**

**glVertex3f(0, 0, 0);**

**glVertex3f(size,0, 0);**

**glEnd();**

**glTranslatef(size, 0, 0); // Move cursor to end of line**

**// End the drawing of a line**

**}**

**D:\GIT\CPP\source\Jump.h**

**//**

**// Jump.h**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#ifndef CPP\_Jump\_h**

**#define CPP\_Jump\_h**

**#include "Instruction.h"**

**// Inherits publically from Instruction**

**class Jump: public Instruction**

**{**

**public:**

**Jump(double s); // Constructor**

**void Draw(); // Virtual Draw function**

**~Jump(){} // Destructor**

**private:**

**Jump(){} // Default Constructor**

**};**

**#endif**

**D:\GIT\CPP\source\Jump.cpp**

**//**

**// Jump.cpp**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#include "Jump.h"**

**#ifdef \_WIN32**

**#include<Windows.h>**

**#include <GL/glut.h> // The GL Utility Toolkit (Glut) Header**

**#endif**

**#ifdef \_\_APPLE\_\_**

**#include <GLUT/glut.h>**

**#endif**

**#ifdef \_\_linux\_\_**

**#include <GL/glut.h>**

**#endif**

**// Constructor**

**Jump::Jump(double s)**

**{**

**//size is a member variable of the instruction base class**

**size = s;**

**}**

**void Jump::Draw()**

**{**

**// Jump**

**glTranslatef(size, 0, 0);**

**// End Jump**

**}**

**D:\GIT\CPP\source\Rotate.h**

**//**

**// Rotate.h**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#ifndef CPP\_Rotate\_h**

**#define CPP\_Rotate\_h**

**#include "Instruction.h"**

**// Inherits publically from Instruction**

**class Rotate: public Instruction**

**{**

**public:**

**Rotate(double s); // Constructor**

**void Draw(); // Virtual Draw function**

**~Rotate(){} // Destructor**

**private:**

**Rotate(){} // Default Constructor**

**};**

**D:\GIT\CPP\source\Rotate.cpp**

**//**

**// Rotate.cpp**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#include "Rotate.h"**

**#ifdef \_WIN32**

**#include<Windows.h>**

**#include <GL/glut.h> // The GL Utility Toolkit (Glut) Header**

**#endif**

**#ifdef \_\_APPLE\_\_**

**#include <GLUT/glut.h>**

**#endif**

**#ifdef \_\_linux\_\_**

**#include <GL/glut.h>**

**#endif**

**// Constructor**

**Rotate::Rotate(double s)**

**{**

**//size is a member variable of the instruction base class**

**size = s;**

**}**

**// Virtual Draw function**

**// The "Left" and "Right" commands are both represented by this class.**

**// Only one is initialised with -size.**

**void Rotate::Draw()**

**{**

**// Rotate cursor by size degrees clockwise**

**glRotatef(size, 0, 0, 1);**

**// End rotate**

**}**

**#endif**

**D:\GIT\CPP\source\Repeat.h**

**//**

**// Repeat.h**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#ifndef CPP\_Repeat\_h**

**#define CPP\_Repeat\_h**

**#include "ParseInput.h"**

**// Inherits publically from Instruction**

**class Repeat: public Instruction**

**{**

**public:**

**Repeat(){} // Default Constructor**

**void Draw(); // Virtual Draw function**

**Repeat(double sz); // Constructor**

**~Repeat(){} // Destructor**

**friend ifstream& operator>>(ifstream& is, Repeat& r); // Input operator**

**void SetBrackets(bool b){RepeatProgram.SetBrackets(b);} // Allows write access to brackets member variable**

**private:**

**// Repeat class has a parse input class as a member variable**

**// The 'repeat' command is treated as a sub program within the LOGO program**

**ParseInput RepeatProgram;**

**};**

**D:\GIT\CPP\source\Repeat.cpp**

**//**

**// Repeat.cpp**

**// CPP**

**//**

**// Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**// Copyright (c) 2012 Oli Davis, James Sinclair and Craig Lord. All rights reserved.**

**//**

**#include "Repeat.h"**

**#include "Exceptions.h"**

**#ifdef \_WIN32**

**#include<Windows.h>**

**#include <GL/glut.h> // The GL Utility Toolkit (Glut) Header**

**#endif**

**#ifdef \_\_APPLE\_\_**

**#include <GLUT/glut.h>**

**#endif**

**#ifdef \_\_linux\_\_**

**#include <GL/glut.h>**

**#endif**

**// Constructor**

**Repeat::Repeat(double sz)**

**{**

**//size is a member variable of the instruction base class**

**size = sz;**

**}**

**// Virtual Draw function**

**void Repeat::Draw()**

**{**

**for (int i(0); i< size;i++)**

**{**

**// Executes the ParseInput class for the specified number (size) of repeats**

**RepeatProgram.Draw();**

**}**

**}**

**// Reads the sub program to be stored in the ParseInput class**

**ifstream& operator>>(ifstream& is, Repeat& r)**

**{**

**char c;**

**is >> ws;**

**if (is.peek() == '[' )**

**{**

**is.get(c);**

**}**

**else**

**{**

**throw BracketsError();**

**{}**

**}**

**is >> r.RepeatProgram;**

**return is;**

**}**

**#endif**

**D:\GIT\CPP\source\window.h (included for completeness)**

**#ifndef \_\_WINDOW\_H\_\_**

**#define \_\_WINDOW\_H\_\_**

**#ifdef \_WIN32**

**#include<Windows.h>**

**#include <GL/glut.h> // The GL Utility Toolkit (Glut) Header**

**#endif**

**#ifdef \_\_APPLE\_\_**

**#include <GLUT/glut.h>**

**#endif**

**#ifdef \_\_linux\_\_**

**#include <GL/glut.h>**

**#endif**

**#include <stdlib.h>**

**static void draw(void);**

**class window {**

**public:**

**window(int argc, char\*\* argv);**

**~window(){};**

**static void reshape(int w,int h);**

**static void keyboard ( unsigned char key, int x, int y );**

**static void display();**

**};**

**window::window(int argc, char\*\* argv)**

**{**

**glutInit( &argc, argv );**

**glutInitWindowSize ( 500, 500 );**

**glutInitDisplayMode ( GLUT\_RGB | GLUT\_DOUBLE );**

**glutCreateWindow ( "OOD assignment" );**

**glEnable(GL\_DEPTH\_TEST);**

**glDepthFunc(GL\_LEQUAL);**

**glShadeModel(GL\_SMOOTH);**

**glClearColor(0.0f, 0.0f, 0.0f, 0.0f);**

**glEnable(GL\_COLOR\_MATERIAL );**

**glHint(GL\_PERSPECTIVE\_CORRECTION\_HINT, GL\_NICEST);**

**glutReshapeFunc ( reshape );**

**glutDisplayFunc ( display );**

**glutKeyboardFunc ( keyboard );**

**glutMainLoop ( );**

**}**

**void window::reshape ( int w, int h )**

**{**

**glViewport ( 0, 0, w, h );**

**glMatrixMode ( GL\_PROJECTION );**

**glLoadIdentity ( );**

**if ( h==0 )**

**gluPerspective ( 80, ( float ) w, 1.0, 5000.0 );**

**else**

**gluPerspective ( 80, ( float ) w / ( float ) h, 1.0, 5000.0 );**

**glMatrixMode ( GL\_MODELVIEW );**

**glLoadIdentity ( );**

**glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);**

**}**

**void window::keyboard( unsigned char key, int x, int y )**

**{**

**switch ( key ) {**

**case 27:**

**exit ( 0 );**

**break;**

**default:**

**break;**

**}**

**}**

**void window::display()**

**{**

**glLoadIdentity();**

**glTranslatef(0.0f,0.0f,-6.0f);**

**glColor3f(1,0,0);**

**draw();**

**glutSwapBuffers ( );**

**glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);**

**}**

**#endif /\* \_\_WINDOW\_H\_\_ \*/**

**D:\GIT\CPP\source\Makefile**

**# Makefile for C++ Assignment**

**# Created by Oli Davis, James Sinclair and Craig Lord on 30/11/2012.**

**# Working for Ubuntu 12.04 (06/12/12)**

**CC=g++**

**CFLAGS=-lGL -lglut -lGLU -lm**

**shape: Exceptions.h shape.o ParseInput.o Instruction.h Forward.o Jump.o Rotate.o Repeat.o window.h**

**$(CC) $(CFLAGS) shape.o ParseInput.o Forward.o Jump.o Rotate.o Repeat.o -o shape**

**shape.o: Exceptions.h shape.cpp ParseInput.h window.h**

**$(CC) $(CFLAGS) -c shape.cpp**

**ParseInput.o: ParseInput.cpp Exceptions.h Instruction.h Forward.h Jump.h Rotate.h Repeat.h**

**$(CC) -c ParseInput.cpp**

**Forward.o: Forward.cpp Forward.h Instruction.h**

**$(CC) -c Forward.cpp**

**Jump.o: Jump.cpp Jump.h Instruction.h**

**$(CC) -c Jump.cpp**

**Rotate.o: Rotate.cpp Rotate.h Instruction.h**

**$(CC) -c Rotate.cpp**

**Repeat.o: Repeat.cpp Repeat.h ParseInput.h**

**$(CC) -c Repeat.cpp**

**clean:**

**rm -rf \*o shape**