ECE 462
Object-Oriented Programming using C++ and Java

Sound

Yung-Hsiang Lu
yunglu@purdue.edu
Java Sound
javax.sound.midi

Class MidiSystem

java.lang.Object
  javax.sound.midi.MidiSystem

public class MidiSystem
  extends Object

The MidiSystem class provides access to the installed MIDI system resources, including devices such as synthesizers, sequencers, and MIDI input and output ports. A typical simple MIDI application might begin by invoking one or more MidiSystem methods to learn what devices are installed and to obtain the ones needed in that application.
javax.sound.midi

Interface Sequencer

All Superinterfaces:

MidiDevice

public interface Sequencer
extends MidiDevice

A hardware or software device that plays back a MIDI sequence is known as a sequencer. A MIDI sequence contains lists of time-stamped MIDI data, such as might be read from a standard MIDI file. Most sequencers also provide functions for creating and editing sequences.
Sams Teach Yourself Java 2 in 21 Days (2nd Edition) [BARGAIN PRICE] (Paperback)

by Laura Lemay (Author), Rogers Cadenhead (Author)

No customer reviews yet. Be the first.

Available from these sellers.

› 7 used & new available from $7.89

This is a bargain book and quantities are limited. Bargain books are new but could include a small
MIDI
(Musical Instrument Digital Interface)

- instructions for sound synthesizers (sequencers)
- .mid not recording (such as .wav files)
- music has not been played yet
- usually much smaller in file sizes
- may have high quality by good sound cards

```java
File file = new File("betsy.mid");
currentSound = MidiSystem.getSequence(file);
player = MidiSystem.getSequencer();
player.open();
player.setSequence(currentSound);
player.start();
```
public class Main {
    /** *
     * public static void main(String[] args) {
     * // TODO code application logic here
     * Sequence currentSound;
     * Sequencer player;
     * try {
     *     File file = new File("C:\yunglu\teaching\008\betsy.mid");
     *     currentSound = MidiSystem.getSequence(file);
     *     player = MidiSystem.getSequencer();
     *     player.open();
     *     player.setSequence(currentSound);
     *     player.start();
     *     while (player.isRunning()) {
     *         try {
     *             Thread.sleep(10);
     *         } catch (InterruptedException ie) {
     *             System.out.println("Exception "+ ie);
     *         }
     *         player.close();
     *     } catch (Exception ioe) {
     *         System.out.println("Exception "+ ioe);
     *     }
     * }
     * }
     */
}
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static Receiver</td>
<td><code>getReceiver</code>() Obtains a MIDI receiver from an external MIDI port or other default device.</td>
</tr>
<tr>
<td>static Sequence</td>
<td><code>getSequence(File file)</code> Obtains a MIDI sequence from the specified File.</td>
</tr>
<tr>
<td>static Sequence</td>
<td><code>getSequence(InputStream stream)</code> Obtains a MIDI sequence from the specified input stream.</td>
</tr>
<tr>
<td>static Sequence</td>
<td><code>getSequence(URL url)</code> Obtains a MIDI sequence from the specified URL.</td>
</tr>
<tr>
<td>static Sequencer</td>
<td><code>getSequencer()</code> Obtains the default sequencer connected to the default receiver.</td>
</tr>
</tbody>
</table>
Qt Sound
QSound Class Reference

[QtGui module]

The QSound class provides access to the platform audio facilities. More...

#include <QSound>

InheritsQObject.

- List of all members, including inherited members
- Qt 3 support members

Public Functions

- QSound(const QString &filename, QObject *parent = 0)
painter.drawEllipse(bx, by, ballSize, ballSize);

// update ball's position
ballX += ballVX;
bally += ballVY;
if (ballX < 0) {
    ballX = 0;
    ballVX = -ballVX;
    QSound::play("C:\Windows\media\ringin.wav");
}
if (ballX > (ww - ballSize)) {
    ballX = ww - ballSize;
    ballVX = -ballVX;
    QSound::play("C:\Windows\media\ringout.wav");
}
if (bally < 0) {
    bally = 0;
    ballVY = -ballVY;
    QSound::play("C:\Windows\media\tada.wav");
}
if (bally > (wh - ballSize)) {
    bally = wh - ballSize;
    ballVY = -ballVY;
    QSound::play("C:\Windows\media\chord.wav");
}
QString str(QString::number(ballX));
str.append("^");
str.append(QString::number(ballY));
painter.drawText(10, 10, str);
err = QSound::number(ballVY);
Java Play Wave File
import java.io.File;
import javax.sound.sampled.AudioFormat;
import javax.sound.sampled.AudioInputStream;
import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.SourceDataLine;

/**
 * @author yunghu
 */

class Speaker {
    File soundFile;

    public Speaker() {
        soundFile = new File("C:\\yunghu\\teaching\\2008\\train.wav");
        try {
            // create audio input stream to file
            AudioInputStream ais = AudioSystem.getAudioInputStream(soundFile);
            // determine the file's audio format
            AudioFormat format = ais.getFormat();
            System.out.println("Format: "+ format);
            // get a line to play the audio
            DataLine.Info info = new DataLine.Info(SourceDataLine.class, format);
        }
    }
}
// get a line to play the audio
DataLine.Info info = new DataLine.Info(
    SourceDataLine.class, format);
SourceDataLine source = (SourceDataLine) audioSystem.getLine(
    info);
// play the file
source.open(format);
source.start();
int read = 0;
byte[] audioData = new byte[16384];
while (read > -1) {
    read = ais.read(audioData, 0, audioData.length);
    if (read > 0) {
        source.write(audioData, 0, read);
    }
}
source.drain();
source.close();
} catch (Exception exc) {
    System.out.println("Error: " + exc.getMessage());
    exc.printStackTrace();
}
System.exit(0);

public class Main {
```java
while (read > -1) {
    read = ais.read(audioData, 0, audioData.length);
    if (read >= 0) {
        source.write(audioData, 0, read);
    }
}
source.drain();
source.close();
}
```

```java
public class Main {
    public static void main(String[] args) {
        // TODO code application logic here
        Speaker speaker = new Speaker();
    }
}
```
ECE 462
Object-Oriented Programming using C++ and Java

Performance Profile

Yung-Hsiang Lu
yunglu@purdue.edu
Program Performance

• In many cases, performance decides success.
  – # transactions
  – response time for collision detection and avoidance on airplanes
  – multimedia (video, music, voice ...)
  – interactive games
  – handwriting or voice recognition
• A successful software developer must pay attention to performance.
Performance Improvement

• make programs modular ⇒ easier to understand where time is spent
• Keep It Simple (KIS)
• learn better algorithms and use well-known algorithms
• understand problems’ requirements and limitations
• find approximate solutions, if they are good enough
• put your efforts at the right place ⇒ know what matters
Profiling in Netbeans
/*
 * To change this template, choose
 * and open the template in the editor
 */

package performance;

/**
 * @author yunglu
 */

public class Main {

/**
 * @param args the command line arguments
 */

public static void main(String[] args)
{
    // TODO code application logic
    Matrix MA = Matrix.random(5, 4);
    MA.print();
    Matrix MB = Matrix.identity(4);
    MB.print();
}
}
/*
   * To change this template, choose Tools | Templates
   * and open the template in the editor.
   */
   package performance;

   /**
   * @author yunlulu
   */
   public class Main
   {
      /**
       * @param args
       */
      public static void main(String[] args)
      {
         // TODO code addition
         Matrix MA = Matrix.MA;
         MA.print();
         Matrix MB = Matrix.IDENTITY(4);
         MB.print();
      }
   }

   Enable Profiling of Performance

   Profiler is not integrated with project Performance.
   Click OK to perform the integration.

   As part of the integration, the project build script will be modified.
   The original file will be backed up as build-before-profile.xml.

   Note: You can undo the profiler integration at any time by invoking
   Profile | Advanced Commands | Unintegrate Profile from the main menu.
Profiler will now perform an initial calibration of your machine and target JVM.

This calibration needs to be performed the first time you run the profiler to ensure that timing results are accurate when profiling your application. To ensure the calibration data is accurate, please make sure that other applications are not placing a noticeable load on your machine at this time.

You can run the calibration again by choosing "Profiler | Advanced Commands | Run Profiler Calibration"

Warning: If your computer uses dynamic CPU frequency switching, please disable it and do not use it when profiling.
```java
package performance;

public class Main {
  public static void main(String[] arg) {
    // TODO code application logic here
    Matrix MA = Matrix.random(5, 4);
    MA.print();
    Matrix MB = Matrix.identity(4);
    MB.print();
  }
}
```
package performance;

public class Main {
    // TODO code here
    Matrix MA = Matrix.rentity(4);
    MA.print();
    Matrix MB = Matrix.rentity(4);
    MB.print();
}

The calibration was successful.
Click Show Details to see calibration results.

Warning: If your computer uses dynamic CPU frequency switching,
please disable it and rerun calibration as changing the CPU frequency
during profiling would produce inaccurate results.
### Call Tree - Method

<table>
<thead>
<tr>
<th>Method</th>
<th>Time [%]</th>
<th>Time</th>
<th>Invocations</th>
</tr>
</thead>
<tbody>
<tr>
<td>All threads</td>
<td></td>
<td>149 ms</td>
<td>1</td>
</tr>
<tr>
<td>Main</td>
<td></td>
<td>149 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Main.main (String[])</td>
<td></td>
<td>149 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.print ()</td>
<td></td>
<td>145 ms</td>
<td>2</td>
</tr>
<tr>
<td>Self time</td>
<td></td>
<td>4.4 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.random (int, int)</td>
<td></td>
<td>0.164 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.identity (int)</td>
<td></td>
<td>0.174 ms</td>
<td>1</td>
</tr>
</tbody>
</table>

Profiler Agent: **Initializing...**
Profiler Agent: **Options: C:\yumglu\programs\netbeans\NetBeans 6.0\profiler2\lib,5140,10**
Profiler Agent: **Initialized successfully**
BUILD SUCCESSFUL (total time: 1 minute 57 seconds)
package performance;

public class Main {

    public static void main(String[] args) {
        // TODO code application logic here
        Matrix MA = Matrix.random(64, 64);
        // MA.print();
        Matrix MB = Matrix.identity(64);
        // MB.print();
        Matrix MC = MA.add(MB);
    }
}
Include Java Core
<table>
<thead>
<tr>
<th>Call Tree - Method</th>
<th>Time [%]</th>
<th>Time</th>
<th>Invocations</th>
</tr>
</thead>
<tbody>
<tr>
<td>All threads</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>main</td>
<td></td>
<td>75.4 ms (100%)</td>
<td>1</td>
</tr>
<tr>
<td>performance.Main.main(String[])</td>
<td></td>
<td>75.4 ms (100%)</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.random(int, int)</td>
<td></td>
<td>69.0 ms (91.4%)</td>
<td>1</td>
</tr>
<tr>
<td>java.lang.Math.random()</td>
<td></td>
<td>58.5 ms (77.5%)</td>
<td>4033</td>
</tr>
<tr>
<td>Self time</td>
<td></td>
<td>2.33 ms (3.1%)</td>
<td>1</td>
</tr>
<tr>
<td>java.lang.ClassLoader.loadClassTr</td>
<td></td>
<td>0.139 ms (0.2%)</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.&lt;init&gt;(int, int)</td>
<td></td>
<td>0.085 ms (0.1%)</td>
<td>1</td>
</tr>
<tr>
<td>java.lang.ClassLoader.checkPackage</td>
<td></td>
<td>0.030 ms (0%)</td>
<td>1</td>
</tr>
<tr>
<td>Self time</td>
<td></td>
<td>0.908 ms (1.2%)</td>
<td>1</td>
</tr>
<tr>
<td>java.lang.ClassLoader.loadClassInter</td>
<td></td>
<td>5.46 ms (7.2%)</td>
<td>4</td>
</tr>
<tr>
<td>java.lang.ClassLoader.checkPackage</td>
<td></td>
<td>0.091 ms (0.1%)</td>
<td>3</td>
</tr>
</tbody>
</table>
package performance;

public class Main {

    public static void main(String[] args) {
        // TODO code application logic here
        // Matrix MA = Matrix.random(64, 64);
        Matrix MA = Matrix.identity(64);
        MA.print();
        Matrix MB = Matrix.identity(64);
        MB.print();
        Matrix MC = MA.add(MB);
        Matrix MD = MA.subtract(MB);
    }
}
<table>
<thead>
<tr>
<th>Call Tree - Method</th>
<th>Time [%]</th>
<th>Time</th>
<th>Invocations</th>
</tr>
</thead>
<tbody>
<tr>
<td>All threads</td>
<td></td>
<td>10.8 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Main.main (String[])</td>
<td></td>
<td>10.8 ms</td>
<td>1</td>
</tr>
<tr>
<td>java.lang.ClassLoader.loadClass</td>
<td></td>
<td>5.37 ms</td>
<td>4</td>
</tr>
<tr>
<td>performance.Matrix.subtract (perform)</td>
<td></td>
<td>1.97 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.addSubtract</td>
<td></td>
<td>1.73 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.identity (int)</td>
<td></td>
<td>0.860 ms</td>
<td>2</td>
</tr>
<tr>
<td>performance.Matrix.add (performance,)</td>
<td></td>
<td>0.833 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.addSubtract</td>
<td></td>
<td>0.817 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.subtract (null)</td>
<td></td>
<td>0.779 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.subtract (null)</td>
<td></td>
<td>0.000 ms</td>
<td>1</td>
</tr>
<tr>
<td>java.lang.ClassLoader.checkPackage</td>
<td></td>
<td>0.013 ms</td>
<td>1</td>
</tr>
<tr>
<td>java.lang.ClassLoader.checkPackage</td>
<td></td>
<td>0.089 ms</td>
<td>3</td>
</tr>
</tbody>
</table>
package performance;

public class Main {

    public static void main(String[] args) {
        // TODO code application logic here
        Matrix MA = Matrix.identity(640);
        // MA.print();
        Matrix MB = Matrix.identity(640);
        // MB.print();
        Matrix MC = MA.add(MB);
        Matrix MD = MA.subtract(MB);
    }
}
### Call Tree - Method

<table>
<thead>
<tr>
<th>Method</th>
<th>Time (%)</th>
<th>Time</th>
<th>Invocations</th>
</tr>
</thead>
<tbody>
<tr>
<td>All threads</td>
<td></td>
<td>206 ms</td>
<td>1</td>
</tr>
<tr>
<td>main</td>
<td></td>
<td>206 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Main.main(String[])</td>
<td></td>
<td>206 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.identity(int)</td>
<td></td>
<td>115 ms</td>
<td>2</td>
</tr>
<tr>
<td>performance.Matrix.add(performance)</td>
<td></td>
<td>59.6 ms</td>
<td>1</td>
</tr>
<tr>
<td>performance.Matrix.subtract(performance)</td>
<td></td>
<td>23.8 ms</td>
<td>1</td>
</tr>
<tr>
<td>java.lang.ClassLoader.InadClassInter</td>
<td></td>
<td>5.38 ms</td>
<td>4</td>
</tr>
<tr>
<td>Self time</td>
<td></td>
<td>2.54 ms</td>
<td>1</td>
</tr>
<tr>
<td>java.lang.ClassLoader.checkPackage</td>
<td></td>
<td>0.090 ms</td>
<td>3</td>
</tr>
</tbody>
</table>
Netbeans Profiler
This demo shows three of the new profiler features in the NetBeans IDE 6.0: a drill down graph that makes it easier to see which categories your application is spending time in, profiling points that allow you to control the profiler's...
ECE 462
Object-Oriented Programming
using C++ and Java

C++ Performance Profile

Yung-Hsiang Lu
yunglu@purdue.edu
Using gprof
```cpp
#include "matrix.h"

int main(int argc, char * argv [])
{
    Matrix MA = Matrix::random(400, 400);
    Matrix MB = Matrix::identity(400);
    Matrix MC = MA.add(MB);
    Matrix MD = MA.subtract(MA);
    MA.print();
    MB.print();
    MC.print();
    MD.print();
    return 0;
}
```
```cpp
#ifndef MATRIX_H
#define MATRIX_H

class Matrix
{
    private:
        int m_row;
        int m_col;
        double * m_elements;
    Matrix addSubtract(const Matrix BM, int p) const;

    public:
        Matrix(int r, int c);
        virtual ~Matrix();
        // there is memory allocation so we need to implement
        // copy constructor and assignment operator
        static Matrix identity(int r);
        static Matrix random(int r, int c);
        Matrix(const Matrix & morig);
        Matrix & operator = (const Matrix & morig);
        Matrix add(const Matrix BM) const;
        Matrix subtract(const Matrix BM) const;
        void print() const;
};
```
#include <iostream>
#include <time.h>
#include <math.h>
#include "matrix.h"
using namespace std;
Matrix::Matrix(int r, int c) {
    m_row = r;
    m_col = c;
    m_elements = new double[r * c];
    if (m_elements == NULL)
    {
        cout << "Matrix(int r, int c) fail" << endl;
    }
}
Matrix::~Matrix()
{
    if (m_elements)
    {
        delete[] m_elements;
    }
}
```cpp
Matrix::Matrix(const Matrix & morig)
{
    m_row = morig.m_row;
    m_col = morig.m_col;
    int numElements = m_row * m_col;
    m_elements = new double[numElements];
    if (m_elements == NULL)
    {
        cout << "Matrix(const Matrix &) fail" << endl;
    }
    for (int index = 0; index < numElements; index++)
    {
        m_elements[index] = morig.m_elements[index];
    }
}

Matrix & Matrix::operator = (const Matrix & morig)
{
    if (this != & morig)
    {
        delete [] m_elements;
        m_row = morig.m_row;
    }

Matrix & Matrix::operator = (const Matrix & morig)
delete [] m_elements;
m_row = morig.m_row;
m_col = morig.m_col;
int numElements = m_row * m_col;
m_elements = new double[numElements];
if (m_elements == NULL)
{
    cout << "Matrix:: operator = fail" << endl;
}
for (int index = 0; index < numElements; index ++)
{
    m_elements[index] = morig.m_elements[index];
}
return * this;

Matrix Matrix::random(int r, int c)
{
    Matrix AM(r, c);
    int eindex = 0;
Matrix Matrix::identity(int r)
{
    Matrix AM(r, r);
    int eindex = 0;
    for (int rindex = 0; rindex < r; rindex++)
    {
        for (int cindex = 0; cindex < r; cindex++)
        {
            if (rindex == cindex)
            {
                AM.m_elements[eindex] = 1;
            }
            else
            {
                AM.m_elements[eindex] = 0;
            }
            eindex ++;
        }
    }
    return AM;
}
Matrix Matrix::add(const Matrix BM) const
{
    return addSubtract(BM, 1);
}

Matrix Matrix::subtract(const Matrix BM) const
{
    return addSubtract(BM, -1);
}

Matrix Matrix::addSubtract(const Matrix BM, int p) const
{
    if ((m_row != BM.m_row) || (m_col != BM.m_col))
    {
        cout << "plus different dimensions" << endl;
        return Matrix(0, 0);
    }
    Matrix CM(m_row, m_col);
    int eindex = 0;
    for (int rindex = 0; rindex < m_row; rindex++)
    {
        for (int cindex = 0; cindex < m_col; cindex++)
            CM[eindex][cindex] = BM[eindex][cindex];
        eindex++;
    }
    return CM;
}
```cpp
    }
    return CM;
}

void Matrix::print() const
{
    cout << "row, column = " << m_row << " " << m_col << endl;
    int eindex = 0;
    for (int rindex = 0; rindex < m_row; rindex++)
    {
        for (int cindex = 0; cindex < m_col; cindex++)
        {
            cout << m_elements[eindex] << " ";
            eindex ++;
        }
        cout << endl;
    }
```
File Edit Options Buffers Tools Makefile Help

# Makefile for building: cppmatrix
# Generated by qmake (2.01a) (Qt 4.3.0) on: Thu Jun 19 11:35
# Project: cppmatrix.pro
# Template: app
# Command: /home/shay/a/sfwtools/public/qt4.3.0/bin/qmake -u
#          makefile cppmatrix.pro

add -p to the compiler flag

CC = gcc -p
CXX = g++ -p
DEFINES = -DQT_NO_DEBUG -DQT_GUI_LIB -DQT_CORE_LIB -DC
CFLAGS = -m64 -pipe -O2 -Wall -W -D_REENTRANT $(DEFIN
CXXFLAGS = -m64 -pipe -O2 -Wall -W -D_REENTRANT $(DEFIN
INCPATH = -I../../../../../../sfwtools/public/qt4.3.0/mkspec
++-64 -I. -I../../../../../../sfwtools/public/qt4.3.0/include/QtCore
../../sfwtools/public/qt4.3.0/include/QtCoo

Makefile (Makefile) -- 112--Top---------
check -p flag

g++ -p -c -m64 -pipe -O2 -Wall -W -D_REENTRANT -DQT_NO_DEBUG
G -DQT_GUI_LIB -DQT_CORE_LIB -DQT_SHARED -I../../../../../sfwtools/public/qt4.3.0/mkspecs/linux-g++-64 -I -I../../../../sfwtools/public/qt4.3.0/include/QtCore -I../../../../sfwtools/public/qt4.3.0/include/QtGui -I../../../../sfwtools/public/qt4.3.0/include -I -I -I -o main.o main.cc
main.cc:3: warning: unused parameter 'argc'
main.cc:3: warning: unused parameter 'argv'
g++ -p -c -m64 -pipe -O2 -Wall -W -D_REENTRANT -DQT_NO_DEBUG
G -DQT_GUI_LIB -DQT_CORE_LIB -DQT_SHARED -I../../../../../sfwtools/public/qt4.3.0/mkspecs/linux-g++-64 -I -I../../../../sfwtools/public/qt4.3.0/include/QtCore -I../../../../sfwtools/public/qt4.3.0/include/QtGui -I../../../../sfwtools/public/qt4.3.0/include -I -I -I -o matrix.o matrix.cc
g++ -m64 -Wl,-rpath,./home/shay/a/sfwtools/public/qt4.3.0/lib -o cppmatrix main.o matrix.o -L/home/shay/a/sfwtools/public/qt4.3.0/lib -lQtGui -L/home/shay/a/sfwtools/public/qt4.3.0/lib -L/usr/X11R6/lib64 -lpng -lSM -lICE -lpthread -lpthread

YHL C++ Profile
execute the program

```
[(qstruct04) ~/lecturecode/1020/cppmatrix/ ] ./cppmatrix
row, column = 4 4
-356.567  -243.108  341.958  67.7321
-376.531  -217.754  -225.702  -172.67
-25.7591  320.088  475.14  216.214
row, column = 4 4
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
row, column = 4 4
-355.567  -243.108  341.958  67.7321
-376.531  -216.754  -225.702  -172.67
-25.7591  320.088  475.14  217.214
row, column = 4 4
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
[(qstruct04) ~/lecturecode/1020/cppmatrix/ ]
```
473.303 88.3587 -63.7457 -29.1282
-49.5626 -112.3833 313.548 -43.5151
-219.462 -362.018 -230.947 -211.618
row, column = 4 4
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
row, column = 4 4
130.453 -7.64368 -260.949 -207.482
473.303 89.3587 -63.7457 -29.1282
-49.5626 -112.3833 314.548 -43.5151
row, column = 4 4
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
[(qstruct04) ~/lecturecode/1020/cppmatrix/ ] ls
cppmatrix* gmon.out main.o matrix.cc matrix.o
cppmatrix.pro main cc Makefile matrix.h
[(qstruct04) ~/lecturecode/1020/cppmatrix/ ]
gprof ./cppmatrix > profile

<table>
<thead>
<tr>
<th>time</th>
<th>seconds</th>
<th>seconds</th>
<th>calls</th>
<th>ms/call</th>
<th>ms/call</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>66.85</td>
<td>0.02</td>
<td>0.02</td>
<td>4</td>
<td>5.01</td>
<td>5.01</td>
<td>Matri</td>
</tr>
<tr>
<td>33.15</td>
<td>0.03</td>
<td>0.01</td>
<td>2</td>
<td>5.01</td>
<td>5.01</td>
<td>Matri</td>
</tr>
<tr>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>10</td>
<td>0.00</td>
<td>0.00</td>
<td>Matri</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>4</td>
<td>0.00</td>
<td>0.00</td>
<td>Matri</td>
</tr>
<tr>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>global</td>
</tr>
<tr>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>global</td>
</tr>
<tr>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>Matri</td>
</tr>
<tr>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>5.01</td>
<td>Matri</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1 constructors keyed to _ZN6MatrixC2Eii
<table>
<thead>
<tr>
<th>time</th>
<th>seconds</th>
<th>seconds</th>
<th>calls</th>
<th>ms/call</th>
<th>ms/call</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>Matri</td>
</tr>
<tr>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>Matri</td>
</tr>
<tr>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>5.01</td>
<td>Matri</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
#include "matrix.h"

int main(int argc, char * argv [])
{
    Matrix MA = Matrix::random(10000, 10000);
    Matrix MB = Matrix::identity(10000);
    Matrix MC = MA.add(MB);
    Matrix MD = MA.subtract(MA);
    // MA.print();
    // MB.print();
    // MC.print();
    // MD.print();
    return 0;
}
**YHL C++ Profile 19**

<table>
<thead>
<tr>
<th>time</th>
<th>cumulative</th>
<th>self</th>
<th>calls</th>
<th>s/call</th>
<th>total</th>
<th>s/call</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>55.05</td>
<td>5.85</td>
<td>5.85</td>
<td>6</td>
<td>0.97</td>
<td>0.97</td>
<td>Matrix::Matrix(Matrix const&amp;)</td>
<td></td>
</tr>
<tr>
<td>24.17</td>
<td>8.41</td>
<td>2.57</td>
<td>2</td>
<td>1.28</td>
<td>2.26</td>
<td>Matrix::addSubtract(Matrix, int) const</td>
<td></td>
</tr>
<tr>
<td>12.84</td>
<td>9.78</td>
<td>1.39</td>
<td>1</td>
<td>1.36</td>
<td>1.36</td>
<td>Matrix::random(int, int)</td>
<td></td>
</tr>
<tr>
<td>8.22</td>
<td>10.65</td>
<td>0.87</td>
<td>1</td>
<td>0.87</td>
<td>0.87</td>
<td>Matrix::identity(int)</td>
<td></td>
</tr>
<tr>
<td>0.00</td>
<td>10.65</td>
<td>0.00</td>
<td>10</td>
<td>0.00</td>
<td>0.00</td>
<td>Matrix::Matrix()</td>
<td></td>
</tr>
<tr>
<td>0.00</td>
<td>10.65</td>
<td>0.00</td>
<td>4</td>
<td>0.00</td>
<td>0.00</td>
<td>Matrix::Matrix(int, int)</td>
<td></td>
</tr>
<tr>
<td>0.00</td>
<td>10.65</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>global constructors keyed to _ZN6MatrixC2\</td>
<td></td>
</tr>
<tr>
<td>0.00</td>
<td>10.65</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>3.23</td>
<td>Matrix::add(Matrix) const</td>
<td></td>
</tr>
<tr>
<td>0.00</td>
<td>10.65</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>3.23</td>
<td>Matrix::subtract(Matrix) const</td>
<td></td>
</tr>
</tbody>
</table>

- **% cumulative**: a running sum of the number of seconds accounted for by this function and those listed above it.
- **time**: the percentage of the total running time of the program used by this function.
- **self**: the number of seconds accounted for by this function alone. This is the major sort for this listing.
- **calls**: the number of times this function was invoked, if this function is profiled, else blank.
- **s/call**: the average number of milliseconds spent in this function per call, if this function is profiled, else blank.
- **total**: the average number of milliseconds spent in this function. 

---

YHL

C++ Profile
Performance Improvement using Pass by Reference
```cpp
#ifndef MATRIX_H
#define MATRIX_H

class Matrix
{
    private:
        int m_row;
        int m_col;
        double * m_elements;
    Matrix addSubtract(const Matrix & BM , int p) const;

    public:
        Matrix(int r, int c);
        virtual ~Matrix();
        // there is memory allocation so we need to implement
        // copy constructor and assignment operator
        static Matrix identity(int r);
        static Matrix random(int r, int c);
        Matrix(const Matrix & morig);
        Matrix & operator = (const Matrix & morig);
        Matrix add(const Matrix & BM ) const;
        Matrix subtract(const Matrix & BM ) const;
        void print() const;
};
```

YHL C++ Profile 22

Each sample counts as 0.01 seconds.

<table>
<thead>
<tr>
<th>time</th>
<th>cumulative seconds</th>
<th>self seconds</th>
<th>calls</th>
<th>s/call</th>
<th>s/call</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>38.45</td>
<td>2.59</td>
<td>2.59</td>
<td>2</td>
<td>1.29</td>
<td>2.27</td>
<td>Matrix::addSubtract(const Matrix&amp; other, int)</td>
</tr>
<tr>
<td>28.92</td>
<td>4.53</td>
<td>1.93</td>
<td>2</td>
<td>0.97</td>
<td>0.97</td>
<td>Matrix::Matrix(const Matrix&amp; other)</td>
</tr>
<tr>
<td>19.82</td>
<td>5.87</td>
<td>1.33</td>
<td>1</td>
<td>1.33</td>
<td>1.33</td>
<td>Matrix::random(int, int)</td>
</tr>
<tr>
<td>13.12</td>
<td>6.75</td>
<td>0.88</td>
<td>1</td>
<td>0.88</td>
<td>0.88</td>
<td>Matrix::identity(int)</td>
</tr>
<tr>
<td>0.00</td>
<td>6.75</td>
<td>0.00</td>
<td>6</td>
<td>0.00</td>
<td>0.00</td>
<td>Matrix::Matrix()</td>
</tr>
<tr>
<td>0.00</td>
<td>6.75</td>
<td>0.00</td>
<td>4</td>
<td>0.00</td>
<td>0.00</td>
<td>Matrix::Matrix(int, int)</td>
</tr>
<tr>
<td>0.00</td>
<td>6.75</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>0.00</td>
<td>global constructors keyed to _ZN6MatrixC2</td>
</tr>
<tr>
<td>0.00</td>
<td>6.75</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>2.27</td>
<td>Matrix::add(const Matrix&amp; other) const</td>
</tr>
<tr>
<td>0.00</td>
<td>6.75</td>
<td>0.00</td>
<td>1</td>
<td>0.00</td>
<td>2.27</td>
<td>Matrix::subtract(const Matrix&amp; other) const</td>
</tr>
</tbody>
</table>

The percentage of the total running time of the program used by this function.

Cumulative running sum of the number of seconds accounted for by this function and those listed above it.

The number of seconds accounted for by this function alone. This is the major sort for this listing.

The number of times this function was invoked, if this function is profiled, else blank.

The average number of milliseconds spent in this function per call, if this function is profiled, else blank.
ECE 462
Object-Oriented Programming
using C++ and Java

Improve Performance

Yung-Hsiang Lu
yunglu@purdue.edu
Procedure

• find performance bottlenecks
  – use profilers
  – select appropriate data and metrics
• develop a strategy to reduce or remove the bottleneck
  – change a better algorithm
  – don’t do it
  – don’t do it so often
  – don’t do it now
• evaluate the benefits of the strategy
• implement the changes
• repeat the procedure
Don’t Do It Often

X& operator=( const X& other )
{
    if ( this == &other )
    { return *this; }
    // release memory
    // allocate memory
    // copy elements
    return *this;
}

YHL Improve Performance 3

X& operator=( const X& other )
{
    if ( this == &other )
    { return *this; }
    // release memory
    // allocate memory
    if (they have the different amounts of memory)
    {
        // release memory
        // allocate memory
    }
    // copy elements
    return *this;
}
Performance in Interactive Programs

- show progress
- allow other activities simultaneously
- give users options
- remain responsive (by using multiple threads)
- anticipate a user’s actions and pre-compute
#include "matrix.h"
#include <stdlib.h>

int main(int argc, char * argv [])
{
    int row = 1000;
    if (argc > 1)
        { row = atoi(argv[1]); }
    if (row < 1000)
        { row = 1000; }
    Matrix MA = Matrix::identity(row);
    Matrix MB = Matrix::identity(row);
    Matrix MC = MA.add(MB);
    Matrix MD = MA.subtract(MA);
    MC = MA.subtract(MA);
    MD = MA.add(MB);
    MC = MB.subtract(MD);
    MD = MC.add(MA);  
    MA = MA;
    MC = MC;
    MD = MD;
    return 0;
}
Matrix & Matrix::operator = (const Matrix & morig)
{
    if (this != & morig)
    {
        int numElements = morig.m_row * morig.m_col;
        if ((m_row != morig.m_row) ||
            (m_col != morig.m_col))
        {
            delete [] m_elements;
            m_row = morig.m_row;
            m_col = morig.m_col;
            int numElements = m_row * m_col;
            m_elements = new double[numElements];
            if (m_elements -- NULL)
            {
                cout << "Matrix:: operator = fail"
                    << endl;
            }
        }
    for (int index = 0; index < numElements;
                index++)
    {
Compiler Optimization

add \(-O\) to g++ flag
YHL Improve Performance 8

```
[(qstruct04) ~/lecturecode/1020/cppmatrix/ ] more Makefile
CXX = g++
CXXFLAGS = -p -0

cppmatrix: main.o matrix.o
    $(CXX) $(CXXFLAGS) main.o matrix.o -o cppmatrix

main.o: main.cc matrix.h
    $(CXX) -c $(CXXFLAGS) main.cc

matrix.o: matrix.cc matrix.h
    $(CXX) -c $(CXXFLAGS) matrix.cc

clean:
    rm -f *.o cppmatrix

[(qstruct04) ~/lecturecode/1020/cppmatrix/ ]
```
Memory Management in Java
Garbage Collection

Car c1 = new Car("B14937");

c1 = new Car("N86213");

c1 = null;
Garbage Collection

Car c1 = new Car("B14937");
Car c2;
c1 = new Car("N86213");
c2 = c1;
c1 = null;
c2 = null;
Object Creation in Java

- set unneeded objects to null
- reuse objects’ (by changing some attributes), avoid creating short-life objects
- set objects’ attributes to be sufficiently large to prevent frequent resizing
- use static attributes when applicable
- avoid creating intermediate objects

```java
String str = new String("11");
int sval = Integer.parseInt(str);
```

```java
String str = new String("11");
int sval = new Integer(str).intValue();
```
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>numberOfLeadingZeros</code></td>
<td>Returns the number of zero bits preceding the highest-order (&quot;leftmost&quot;) one-bit in the two's complement binary representation of the specified <code>int</code> value.</td>
</tr>
<tr>
<td><code>numberOfTrailingZeros</code></td>
<td>Returns the number of zero bits following the lowest-order (&quot;rightmost&quot;) one-bit in the two's complement binary representation of the specified <code>int</code> value.</td>
</tr>
<tr>
<td><code>parseInt(String s)</code></td>
<td>Parses the string argument as a signed decimal integer.</td>
</tr>
<tr>
<td><code>parseInt(String s, int radix)</code></td>
<td>Parses the string argument as a signed integer in the radix specified by the second argument.</td>
</tr>
<tr>
<td><code>reverse(int i)</code></td>
<td>Returns the value obtained by reversing the order of the bits in the two's complement binary representation of the specified <code>int</code> value.</td>
</tr>
<tr>
<td><code>reverseBytes(int i)</code></td>
<td>Returns the value obtained by reversing the order of the bytes in the two's complement representation of the specified <code>int</code> value.</td>
</tr>
</tbody>
</table>