CS22510 Assignment - Runners and Riders

Tom Leaman (thl5)

March 22, 2013

Contents

1 Event Creator

1.1 Source code

1.1.1 main.cpp

```
/*
 * main.cpp
 *
 * Tom Leaman (thl5@aber.ac.uk)
 */

#include <fstream>
#include <iostream>
#include <string>
#include <vector>

int show_menu() {
  using namespace std;

  cout << endl;
  cout << "Menu" << endl;
  cout << endl;
  cout << "\t1. Create new event" << endl;
  cout << "\t2. Add entrant" << endl;
  cout << "\t3. Add course" << endl;
  cout << "\t4. Quit" << endl;

  cout << endl;
  cout << ">> ";
  int result;
  cin >> result;
  return result;
}

void write_event(std::string filename, std::string name, std::string
   date, std::string time) {
  using namespace std;

  ofstream file;
  file.open(filename.c_str());

  file << name << "\n";
  file << date << "\n";
  file << time << "\n";

  file.close();
}

void create_event() {
  using namespace std;

  cout << "Please enter event name" << endl;
  cout << ">> ";
  string name;
  cin.ignore();
  getline(cin, name);

  cout << "Please enter date" << endl;
  cout << ">> ";
  string date;
  cin.ignore();
  getline(cin, date);

  cout << "Please enter start time" << endl;
  cout << ">> ";
  string time;
  cin >> time;

  cout << "Please enter file name to save data" << endl;
  cout << ">> ";
  string filename;
  cin >> filename;

  write_event(filename, name, date, time);
}

void write_entrant(std::string filename, int id, char course,
   std::string name) {
  using namespace std;

  ofstream file;
  file.open(filename.c_str(), ios::app);

  file << id << " " << course << " " << name << "\n";

  file.close();
}

void add_entrant() {
  using namespace std;

  cout << "Please enter entrant id" << endl;
  cout << ">> ";
  int id;
  cin >> id;

  cout << "Please enter course id" << endl;
  cout << ">> ";
  char course;
  cin >> course;

  cout << "Please enter entrant name" << endl;
  cout << ">> ";
  string name;
  cin.ignore();
  getline(cin, name);

  cout << "Please enter entrant file" << endl;
  cout << ">> ";
  string filename;
  cin >> filename;

  write_entrant(filename, id, course, name);
}

void write_course(std::string filename, char id, int num_nodes,
   std::vector<int> nodes) {
```

```
  using namespace std;

  ofstream file;
  file.open(filename.c_str(), ios::app);

  file << id << " " << num_nodes;
  for (int i = 0; i < num_nodes; i++) {
    file << " " << nodes[i];
  }
  file << "\n";

  file.close();
}

void add_course() {
  using namespace std;

  cout << "Please enter course id" << endl;
  cout << ">> ";
  char id;
  cin >> id;

  cout << "Please enter the number of nodes" << endl;
  cout << ">> ";
  int num_nodes;
  cin >> num_nodes;

  vector<int> nodes(num_nodes);
  for (int i = 0; i < num_nodes; i++) {
    cout << "Please enter node " << (i+1) << endl;
    cout << ">> ";
    cin >> nodes[i];
  }

  cout << "Please enter course file" << endl;
  cout << ">> ";
  string filename;
  cin >> filename;

  write_course(filename, id, num_nodes, nodes);
}

int main(int argc, char* argv[]) {
  using namespace std;

  cout << "Event Creator" << endl;

  bool running = true;
  while (running) {
    int input = show_menu();
    switch (input) {
      case 1:
        create_event();
        break;
      case 2:
        add_entrant();
        break;
      case 3:
        add_course();
        break;
      case 4:
        running = false;
        break;
      default:
        // invalid option
        // do nothing
        break;
    }
  }

  return 0;
}
```

1.1.2 Makefile

```
CFLAGS=-g -Wall

clean:
  rm -rf main
```

1.2 Compiler output

```
tom@twoflower:~/cs22510-assignment/event_creator $ make main
g++     main.cpp   -o main
```

1.3 Example usage

```
tom@twoflower:~/cs22510-assignment/event_creator $ ./main
Event Creator

Menu

  1. Create new event
  2. Add entrant
  3. Add course
  4. Quit

>> 1
Please enter event name
>> 100 Metre Dash
Please enter date
>> 23rd March 2013
Please enter start time
>> 10:00
Please enter file name to save data
>> name.txt

Menu

  1. Create new event
  2. Add entrant
  3. Add course
  4. Quit

>> 2
Please enter entrant id
>> 1
Please enter course id
>> A
```

```
Please enter entrant name
>> Alan Freeman
Please enter entrant file
>> entrants.txt

Menu

  1. Create new event
  2. Add entrant
  3. Add course
  4. Quit

>> 2
Please enter entrant id
>> 2
Please enter course id
>> A
Please enter entrant name
>> Pete Murray
Please enter entrant file
>> entrants.txt

Menu

  1. Create new event
  2. Add entrant
  3. Add course
  4. Quit

>> 2
Please enter entrant id
>> 3
Please enter course id
>> A
Please enter entrant name
>> David Jacobs
Please enter entrant file
>> entrants.txt

Menu

  1. Create new event
  2. Add entrant
  3. Add course
  4. Quit

>> 2
Please enter entrant id
>> 4
Please enter course id
>> B
Please enter entrant name
>> Samantha Juste
Please enter entrant file
>> entrants.txt

Menu

  1. Create new event
  2. Add entrant
  3. Add course
  4. Quit

>> 2
Please enter entrant id
>> 5
Please enter course id
>> B
Please enter entrant name
>> Simon Dee
Please enter entrant file
>> entrants.txt

Menu

  1. Create new event
  2. Add entrant
  3. Add course
  4. Quit

>> 3
Please enter course id
>> A
Please enter the number of nodes
>> 8
Please enter node 1
>> 1
Please enter node 2
>> 2
Please enter node 3
>> 3
Please enter node 4
>> 9
Please enter node 5
>> 12
Please enter node 6
>> 13
Please enter node 7
>> 2
Please enter node 8
>> 1
Please enter course file
>> courses.txt

Menu

  1. Create new event
  2. Add entrant
  3. Add course
  4. Quit

>> 3
Please enter course id
>> B
Please enter the number of nodes
>> 11
Please enter node 1
>> 1
```

```
Please enter node 2
>> 2
Please enter node 3
>> 3
Please enter node 4
>> 9
Please enter node 5
>> 8
Please enter node 6
>> 10
Please enter node 7
>> 11
Please enter node 8
>> 12
Please enter node 9
>> 13
Please enter node 10
>> 2
Please enter node 11
>> 1
Please enter course file
>> courses.txt

Menu

  1. Create new event
  2. Add entrant
  3. Add course
  4. Quit

>> 4
```

1.4 Generated files

1.4.1 name.txt

```
100 Metre Dash
3rd March 2013
10:00
```

1.4.2 entrants.txt

```
1 A Alan Freeman
2 A Pete Murray
3 A David Jacobs
4 B Samantha Juste
5 B Simon Dee
```

1.4.3 courses.txt

```
A 8 1 2 3 9 12 13 2 1
B 11 1 2 3 9 8 10 11 12 13 2 1
```

2 Checkpoint Manager

2.1 Source code

2.1.1 event/Course.java

```java
package event;

import event.node.Node;

import java.util.List;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class Course {

  private char id;
  private List<Node> nodes;

  public Course(char id, List<Node> nodes) {
    this.id = id;
    this.nodes = nodes;
  }

  public char getId() {
    return id;
  }

  public List<Node> getNodes() {
    return nodes;
  }

  public Node getLastNode() {
    return nodes.get(nodes.size() - 1);
  }

}
```

2.1.2 event/Entrant.java

```
package event;

import event.node.Node;
import event.node.JunctionNode;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class Entrant {

  public static final Status NOT_STARTED = Status.NOT_STARTED;
  public static final Status RUNNING = Status.RUNNING;
  public static final Status STOPPED = Status.STOPPED;
  public static final Status FINISHED = Status.FINISHED;
  public static final Status DISQUALIFIED = Status.DISQUALIFIED;

  private int id;
  private Course course;
  private String name;

  private Node currentNode;
  private Status status;

  public Entrant(int id, Course course, String name) {
    this.id = id;
    this.course = course;
    this.name = name;

    currentNode = null;
    status = NOT_STARTED;
  }

  public int getId() {
    return id;
  }

  public Course getCourse() {
    return course;
  }

  public String getName() {
    return name;
  }

  public Status getStatus() {
    return status;
  }

  public void setStatus(Status status) {
    this.status = status;
  }

  /**
   *  returns the next !junction node or null if one cannot be found
   */
  public Node getNextCheckpoint() {
    int currI = course.getNodes().indexOf(currentNode);
    if (currI == -1) return getCourse().getNodes().get(0); // clearly
       hasn't started yet
    for (int i = currI + 1; i < course.getNodes().size(); i++) {
      if (!(course.getNodes().get(i) instanceof JunctionNode))
        return course.getNodes().get(i);
    }
    return null;
  }

  public void updateLocation(Node node) {
    currentNode = node;
  }

  private enum Status {
    NOT_STARTED,
    RUNNING,
    STOPPED,
    FINISHED,
    DISQUALIFIED
  }

}
```

2.1.3 event/Event.java

```
package event;

import event.node.Node;

import java.util.List;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class Event {

  private List<Node> nodes;
  private List<Entrant> entrants;

  public Event(List<Node> nodes, List<Entrant> entrants) {
    this.nodes = nodes;
    this.entrants = entrants;
  }

  public Node getNode(int id) {
    for (Node n : nodes) {
      if (n.getId() == id) return n;
    }
    return null;
  }

  public List<Node> getNodes() {
    return nodes;
  }

  public Entrant getEntrant(int id) {
    for (Entrant e : entrants) {
      if (e.getId() == id) return e;
    }
    return null;
  }

  public List<Entrant> getEntrants() {
    return entrants;
  }

}
```

2.1.4 event/node/Node.java

```
package event.node;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public abstract class Node {

  private int id;

  public Node(int id) {
    this.id = id;
  }

  public int getId() {
    return id;
  }

}
```

2.1.5 event/node/CheckpointNode.java

```
package event.node;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class CheckpointNode extends Node {

  public CheckpointNode(int id) {
    super(id);
  }

}
```

2.1.6 event/node/JunctionNode.java

```
package event.node;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class JunctionNode extends Node {

  public JunctionNode(int id) {
    super(id);
  }

}
```

2.1.7 event/node/MedicalCheckpointNode.java

```
package event.node;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class MedicalCheckpointNode extends Node {

  public MedicalCheckpointNode(int id) {
    super(id);
  }

}
```

2.1.8 event/update/ArrivalUpdate.java

```
package event.update;

import event.Entrant;
import event.node.Node;
import util.Time;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class ArrivalUpdate extends Update {

  public ArrivalUpdate(Node node, Entrant entrant, Time time) {
    super(node, entrant, time);
  }

  @Override
  public char getType() {
    return 'A';
  }

  @Override
  public void execute() {
    getEntrant().updateLocation(getNode());
    getEntrant().setStatus(Entrant.STOPPED);
  }

}
```

2.1.9 event/update/DepartureUpdate.java

```java
package event.update;

import event.Entrant;
import event.node.Node;
import util.Time;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class DepartureUpdate extends Update {

  public DepartureUpdate(Node node, Entrant entrant, Time time) {
    super(node, entrant, time);
  }

  @Override
  public char getType() {
    return 'D';
  }

  @Override
  public void execute() {
    getEntrant().setStatus(Entrant.RUNNING);
  }

}
```

2.1.10 event/update/ExcludedUpdate.java

```
package event.update;

import event.Entrant;
import event.node.Node;
import util.Time;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class ExcludedUpdate extends Update {

  public ExcludedUpdate(Node node, Entrant entrant, Time time) {
    super(node, entrant, time);
  }

  @Override
  public char getType() {
    return 'E';
  }

  @Override
  public void execute() {
    getEntrant().setStatus(Entrant.DISQUALIFIED);
  }

}
```

2.1.11 event/update/InvalidUpdate.java

```
package event.update;

import event.Entrant;
import event.node.Node;
import util.Time;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class InvalidUpdate extends Update {

  public InvalidUpdate(Node node, Entrant entrant, Time time) {
    super(node, entrant, time);
  }

  @Override
  public char getType() {
    return 'I';
  }

  @Override
  public void execute() {
    getEntrant().updateLocation(getNode());
    getEntrant().setStatus(Entrant.DISQUALIFIED);
  }

}
```

2.1.12 event/update/TimeUpdate.java

```java
package event.update;

import event.Entrant;
import event.node.Node;
import util.Time;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class TimeUpdate extends Update {

  public TimeUpdate(Node node, Entrant entrant, Time time) {
    super(node, entrant, time);
  }

  @Override
  public char getType() {
    return 'T';
  }

  @Override
  public void execute() {
    getEntrant().setStatus(Entrant.RUNNING);
    getEntrant().updateLocation(getNode());
    if (getNode() == getEntrant().getCourse().getLastNode())
      getEntrant().setStatus(Entrant.FINISHED);
  }

}
```

2.1.13 event/update/Update.java

```
package event.update;

import event.Entrant;
import event.node.Node;
import util.Time;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public abstract class Update {

  private Node node;
  private Entrant entrant;
  private Time time;

  public Update(Node node, Entrant entrant, Time time) {
    this.node = node;
    this.entrant = entrant;
    this.time = time;
  }

  public abstract char getType();

  public Node getNode() {
    return node;
  }

  public Entrant getEntrant() {
    return entrant;
  }

  public Time getTime() {
    return time;
  }

  public abstract void execute();

}
```

2.1.14 event/gui/CheckpointPanel.java

```
package gui;

import event.node.CheckpointNode;
import event.node.MedicalCheckpointNode;
import event.node.Node;
import event.Entrant;
import event.Event;
import event.update.*;
import util.FileIO;
import util.Time;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.BorderLayout;
import java.util.Calendar;

import javax.swing.JButton;
import javax.swing.JCheckBox;
import javax.swing.JComboBox;
import javax.swing.JPanel;
import javax.swing.JTextField;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class CheckpointPanel extends JPanel {

  private static final long serialVersionUID = 7212846449217761749L;

  private Event event;

  private JComboBox<String> entrantBox;
  private JComboBox<String> nodeBox;

  private JTextField hrsField;
  private JTextField minsField;
  private JCheckBox currTimeBox;

  private JButton arriveButton;
  private JButton departButton;
  private JButton submitButton;
  private JButton excludeButton;

  public CheckpointPanel(Event event, String timesFile, String
     logFile) {
    this.event = event;
    this.setLayout(new BorderLayout());

    // make components
    ActionListener listener = new Listener(timesFile, logFile);

    // north panel
    JPanel northPanel = new JPanel();

    String[] entrants = new String[event.getEntrants().size()];
```

```
for (int i = 0; i < event.getEntrants().size(); i++) {
  entrants[i] = event.getEntrants().get(i).getName();
}
entrantBox = new JComboBox<String>(entrants);
entrantBox.addActionListener(listener);
northPanel.add(entrantBox);

String[] nodes = new String[event.getNodes().size()];
for (int i = 0; i < event.getNodes().size(); i++) {
  nodes[i] = Integer.toString(event.getNodes().get(i).getId());
}
nodeBox = new JComboBox<String>(nodes);
nodeBox.addActionListener(listener);
northPanel.add(nodeBox);

add(northPanel, BorderLayout.NORTH);

// centre panel
JPanel centrePanel = new JPanel();

hrsField = new JTextField();
hrsField.setColumns(2);
hrsField.setText("00");
centrePanel.add(hrsField);

minsField = new JTextField();
minsField.setColumns(2);
minsField.setText("00");
centrePanel.add(minsField);

currTimeBox = new JCheckBox("Use current time");
currTimeBox.addActionListener(listener);
centrePanel.add(currTimeBox);

add(centrePanel, BorderLayout.CENTER);

// south panel
JPanel southPanel = new JPanel();

arriveButton = new JButton("Arrive");
arriveButton.addActionListener(listener);
southPanel.add(arriveButton);

departButton = new JButton("Depart");
departButton.addActionListener(listener);
southPanel.add(departButton);

submitButton = new JButton("Submit");
submitButton.addActionListener(listener);
southPanel.add(submitButton);

excludeButton = new JButton("Exclude");
excludeButton.addActionListener(listener);
southPanel.add(excludeButton);

add(southPanel, BorderLayout.SOUTH);

updateTime();
updateButtons();
```

```
}

private Entrant getSelectedEntrant() {
  String selected = (String)entrantBox.getSelectedItem();
  for (Entrant e : event.getEntrants()) {
    if (e.getName().equals(selected)) return e;
  }
  return null;
}

private Node getSelectedNode() {
  String selected = (String)nodeBox.getSelectedItem();
  for (Node n : event.getNodes()) {
    if (Integer.toString(n.getId()).equals(selected)) return n;
  }
  return null;
}

private Time getSelectedTime() {
  int hours = Integer.parseInt(hrsField.getText());
  int minutes = Integer.parseInt(minsField.getText());
  return new Time(hours, minutes);
}

private boolean correctNode() {
  return getSelectedEntrant().getNextCheckpoint() ==
      getSelectedNode();
}

private void updateButtons() {
  Node n = getSelectedNode();

  if (n instanceof CheckpointNode) {
    // enable submit, disable the rest
    submitButton.setEnabled(true);
    arriveButton.setEnabled(false);
    departButton.setEnabled(false);
    excludeButton.setEnabled(false);
  } else if (n instanceof MedicalCheckpointNode) {
    // disable submit, enable the rest
    submitButton.setEnabled(false);
    if (getSelectedEntrant().getStatus() == Entrant.STOPPED) {
      arriveButton.setEnabled(false);
      departButton.setEnabled(true);
      excludeButton.setEnabled(true);
    } else {
      arriveButton.setEnabled(true);
      departButton.setEnabled(false);
      excludeButton.setEnabled(false);
    }
  } else {
    // disable them all
    submitButton.setEnabled(false);
    arriveButton.setEnabled(false);
    departButton.setEnabled(false);
    excludeButton.setEnabled(false);
  }
}
```

```
private void updateTime() {
  Calendar cal = Calendar.getInstance();
  hrsField.setText(Integer.toString(
        cal.get(Calendar.HOUR_OF_DAY)));
  minsField.setText(Integer.toString(
        cal.get(Calendar.MINUTE)));
}

private class Listener implements ActionListener {

  private String timesFile;
  private String logFile;

  public Listener(String timesFile, String logFile) {
    this.timesFile = timesFile;
    this.logFile = logFile;
  }

  @Override
  public void actionPerformed(ActionEvent evt) {
    if (evt.getSource() == entrantBox || evt.getSource() == nodeBox)
       {
      updateButtons();
    } else if (evt.getSource() == currTimeBox) {
      if (currTimeBox.isSelected()) {
        // set time boxes
        updateTime();
        // disable them
        hrsField.setEnabled(false);
        minsField.setEnabled(false);
      } else {
        // enable boxes
        hrsField.setEnabled(true);
        minsField.setEnabled(true);
      }
    } else if (evt.getSource() == arriveButton) {
      Update update = new ArrivalUpdate(
        getSelectedNode(), getSelectedEntrant(), getSelectedTime());
      update.execute();
      writeUpdate(update);
      FileIO.appendToFile(logFile, "CM: A type event recorded");
    } else if (evt.getSource() == departButton) {
      Update update = new DepartureUpdate(
        getSelectedNode(), getSelectedEntrant(), getSelectedTime());
      update.execute();
      writeUpdate(update);
      FileIO.appendToFile(logFile, "CM: D type event recorded");
    } else if (evt.getSource() == submitButton) {
      if (correctNode()) {
        Update update = new TimeUpdate(
            getSelectedNode(), getSelectedEntrant(),
               getSelectedTime());
        update.execute();
        writeUpdate(update);
        FileIO.appendToFile(logFile, "CM: T type event recorded");
      } else {
        Update update = new InvalidUpdate(
            getSelectedNode(), getSelectedEntrant(),
               getSelectedTime());
```

```
          update.execute();
          writeUpdate(update);
          FileIO.appendToFile(logFile, "CM: I type event recorded");
        }
      } else if (evt.getSource() == excludeButton) {
        Update update = new ExcludedUpdate(
            getSelectedNode(), getSelectedEntrant(),
                getSelectedTime());
        update.execute();
        writeUpdate(update);
        FileIO.appendToFile(logFile, "CM: E type event recorded");
      }

      // we'll want to keep current time updated if necessary
      // (but without changing what the user thinks they're using for
      // the time) so update it here
      if (currTimeBox.isSelected())
        updateTime();
    }

    private void writeUpdate(Update update) {
      FileIO.appendToFile(timesFile, update.getType() + " " +
          update.getNode().getId() + " " +
          update.getEntrant().getId() + " " +
          update.getTime());
    }

  }

}
```

2.1.15 event/gui/Driver.java

```
package gui;

import event.Course;
import event.Entrant;
import event.Event;
import event.node.Node;
import event.update.Update;
import util.FileIO;
import util.Parser;

import java.util.List;

import javax.swing.JFrame;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class Driver {

  public static void main(String[] args) {
    if (args.length < 7) {
      System.out.println("Usage:");
      System.out.println("java Driver <event_file> <node_file>
          <track_file> " +
          "<course_file> <entrant_file> <time_file> <log_file>");
      System.exit(1);
    }

    //String eventFile = args[0];
    String nodeFile = args[1];
    //String trackFile = args[2];
    String courseFile = args[3];
    String entrantFile = args[4];
    String timeFile = args[5];
    String logFile = args[6];

    // Now read everything in
    List<Node> nodes = Parser.parseNodes(nodeFile);
    List<Course> courses = Parser.parseCourses(courseFile, nodes);
    List<Entrant> entrants = Parser.parseEntrants(entrantFile,
        courses);
    Event event = new Event(nodes, entrants);

    // Process any times already in the file
    List<Update> updates = Parser.parseUpdates(timeFile, event);
    for (Update u : updates) {
      u.execute();
    }
    FileIO.appendToFile(logFile, "CM: processed " + updates.size() + "
        updates from " + timeFile);

    JFrame top = new JFrame("Checkpoint Manager");
    top.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    // this breaks stuff somehow?
    //Dimension dim = new Dimension(800, 600);
    //top.setSize(dim);
    //top.setPreferredSize(dim);
    //top.setMinimumSize(dim);
    //top.setMaximumSize(dim);

    CheckpointPanel panel = new CheckpointPanel(event, timeFile,
       logFile);
    top.setContentPane(panel);
    panel.setVisible(true);

    top.pack();
    top.setVisible(true);
  }

}
```

2.1.16 event/util/FileIO.java

```
package util;

import java.io.*;
import java.nio.channels.FileLock;
import java.util.ArrayList;
import java.util.List;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class FileIO {

  public static List<String> readLines(String filename)
      throws FileNotFoundException {
    List<String> lines = new ArrayList<String>();

    File f = new File(filename);
    try {
      BufferedReader in = new BufferedReader(new FileReader(f));
      String line = in.readLine();
      while (line != null) {
        lines.add(line);
        line = in.readLine();
      }
      in.close();
    } catch (FileNotFoundException e) {
      throw e;
    } catch (IOException e) {
      e.printStackTrace();
    }

    return lines;
  }

  public static void appendToFile(String filename, String text) {
    File f = new File(filename);
    try {
      if (!f.exists())
        f.createNewFile();
      FileOutputStream fos = new FileOutputStream(f, true);
      FileLock fl = fos.getChannel().tryLock();
      while (fl == null) {
        fl = fos.getChannel().tryLock();
      }
      FileWriter out = new FileWriter(fos.getFD());
      out.write(text + "\n");

      fl.release();
      out.close();
    } catch (IOException e) {
      e.printStackTrace();
    }
  }

}
```

2.1.17 event/util/Parser.java

```
package util;

import event.Course;
import event.Entrant;
import event.Event;
import event.node.CheckpointNode;
import event.node.JunctionNode;
import event.node.MedicalCheckpointNode;
import event.node.Node;
import event.update.ArrivalUpdate;
import event.update.DepartureUpdate;
import event.update.ExcludedUpdate;
import event.update.InvalidUpdate;
import event.update.TimeUpdate;
import event.update.Update;

import java.io.FileNotFoundException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class Parser {

  public static List<Node> parseNodes(String filename) {
    List<Node> nodes = new ArrayList<Node>();
    List<String> lines = new ArrayList<String>();

    try {
      lines = FileIO.readLines(filename);
    } catch (FileNotFoundException e) {
      System.err.println(filename + " not found");
      System.exit(1);
    }

    for (String line : lines) {
      String[] tokens = line.split(" ");
      int id = Integer.parseInt(tokens[0]);
      switch (tokens[1]) {
        case "JN":
          nodes.add(new JunctionNode(id));
          break;
        case "CP":
          nodes.add(new CheckpointNode(id));
          break;
        case "MC":
          nodes.add(new MedicalCheckpointNode(id));
          break;
        default:
          System.err.println("Failed to parse node type " + tokens[1]);
          System.exit(1);
          break;
      }
    }
    return nodes;
  }

  public static List<Course> parseCourses(String filename, List<Node>
      nodes) {
    List<Course> courses = new ArrayList<Course>();
    List<String> lines = new ArrayList<String>();

    try {
      lines = FileIO.readLines(filename);
    } catch (FileNotFoundException e) {
      System.err.println(filename + " not found");
      System.exit(1);
    }

    for (String line : lines) {
      String[] tokens = line.split(" ");
      char id = tokens[0].charAt(0); // should be 1 char
      // ignore the next token, I don't care
      List<Node> courseNodes = new ArrayList<Node>();
      for (int i = 2; i < tokens.length; i++) {
        courseNodes.add(findNode(Integer.parseInt(tokens[i]), nodes));
      }

      courses.add(new Course(id, courseNodes));
    }
    return courses;
  }

  private static Node findNode(int id, List<Node> nodes) {
    for (Node n : nodes) {
      if (n.getId() == id) return n;
    }
    return null;
  }

  public static List<Entrant> parseEntrants(String filename,
      List<Course> courses) {
    List<Entrant> entrants = new ArrayList<Entrant>();
    List<String> lines = new ArrayList<String>();

    try {
      lines = FileIO.readLines(filename);
    } catch (FileNotFoundException e) {
      System.err.println(filename + " not found");
      System.exit(1);
    }

    for (String line : lines) {
      String[] tokens = line.split(" ");
      int id = Integer.parseInt(tokens[0]);
      Course course = findCourse(tokens[1].charAt(0), courses);
      String name = new String();
      for (int i = 2; i < tokens.length; i++) {
        name = name + tokens[i] + " ";
      }
      name = name.substring(0, name.length() - 1);

      entrants.add(new Entrant(id, course, name));
    }
    return entrants;
  }

  private static Course findCourse(char id, List<Course> courses) {
    for (Course c : courses) {
      if (c.getId() == id) return c;
    }
    return null;
  }

  public static List<Update> parseUpdates(String filename, Event
     event) {
    List<Update> updates = new ArrayList<Update>();
    List<String> lines = new ArrayList<String>();

    try {
      lines = FileIO.readLines(filename);
    } catch (FileNotFoundException e) {
      // Maybe we've just started, return an empty list
      return updates;
    }

    for (String line : lines) {
      String[] tokens = line.split(" ");
      char type = tokens[0].charAt(0); // should be 1 char
      Node node = event.getNode(Integer.parseInt(tokens[1]));
      Entrant entrant = event.getEntrant(Integer.parseInt(tokens[2]));
      Time time = new Time(Integer.parseInt(tokens[3].split(":")[0]),
            Integer.parseInt(tokens[3].split(":")[1]));

      switch (type) {
        case 'T':
          updates.add(new TimeUpdate(node, entrant, time));
          break;
        case 'A':
          updates.add(new ArrivalUpdate(node, entrant, time));
          break;
        case 'D':
          updates.add(new DepartureUpdate(node, entrant, time));
          break;
        case 'I':
          updates.add(new InvalidUpdate(node, entrant, time));
          break;
        case 'E':
          updates.add(new ExcludedUpdate(node, entrant, time));
          break;
        default:
          System.err.println("Failed to parse update type " + type);
          System.exit(1);
          break;
      }
    }
    return updates;
  }

}
```

2.1.18 event/util/Time.java

```java
package util;

/**
 *
 * @author Tom Leaman (thl5@aber.ac.uk)
 *
 */
public class Time {

  private int hours;
  private int minutes;

  public Time(int hours, int minutes) {
    this.hours = hours;
    this.minutes = minutes;
  }

  public int getHours() {
    return hours;
  }

  public int getMinutes() {
    return minutes;
  }

  @Override
  public String toString() {
    return hours + ":" + minutes;
  }

}
```

2.2 Compiler output

```
tom@twoflower:~/cs22510-assignment/checkpoint_manager $ javac -verbose
   -sourcepath src -classpath bin -d bin src/gui/Driver.java
[parsing started RegularFileObject[src/gui/Driver.java]]
[parsing completed 16ms]
[search path for source files: src]
[search path for class files: /usr/lib/jvm/java-7-openjdk/jre/lib/
   resources.jar,/usr/lib/jvm/java-7-openjdk/jre/lib/rt.jar,/usr/lib/
   jvm/java-7-openjdk/jre/lib/sunrsasign.jar,/usr/lib/jvm/java-7-
   openjdk/jre/lib/jsse.jar,/usr/lib/jvm/java-7-openjdk/jre/lib/jce.
   jar,/usr/lib/jvm/java-7-openjdk/jre/lib/charsets.jar,/usr/lib/jvm/
   java-7-openjdk/jre/lib/netx.jar,/usr/lib/jvm/java-7-openjdk/jre/lib
   /plugin.jar,/usr/lib/jvm/java-7-openjdk/jre/lib/rhino.jar,/usr/lib/
   jvm/java-7-openjdk/jre/lib/jfr.jar,/usr/lib/jvm/java-7-openjdk/jre/
   classes,/usr/lib/jvm/java-7-openjdk/jre/lib/ext/sunpkcs11.jar,/usr/
   lib/jvm/java-7-openjdk/jre/lib/ext/dnsns.jar,/usr/lib/jvm/java-7-
   openjdk/jre/lib/ext/pulse-java.jar,/usr/lib/jvm/java-7-openjdk/jre/
   lib/ext/zipfs.jar,/usr/lib/jvm/java-7-openjdk/jre/lib/ext/
   localedata.jar,/usr/lib/jvm/java-7-openjdk/jre/lib/ext/
   sunjce_provider.jar,bin]
[loading RegularFileObject[src/event/Course.java]]
[parsing started RegularFileObject[src/event/Course.java]]
[parsing completed 1ms]
[loading RegularFileObject[src/event/Entrant.java]]
[parsing started RegularFileObject[src/event/Entrant.java]]
[parsing completed 2ms]
[loading RegularFileObject[src/event/Event.java]]
[parsing started RegularFileObject[src/event/Event.java]]
[parsing completed 1ms]
[loading RegularFileObject[src/event/node/Node.java]]
[parsing started RegularFileObject[src/event/node/Node.java]]
[parsing completed 0ms]
[loading RegularFileObject[src/event/update/Update.java]]
[parsing started RegularFileObject[src/event/update/Update.java]]
[parsing completed 1ms]
[loading RegularFileObject[src/util/Parser.java]]
[parsing started RegularFileObject[src/util/Parser.java]]
[parsing completed 4ms]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/util/List.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/JFrame.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Object.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/String.class)]]
[loading RegularFileObject[src/event/node/JunctionNode.java]]
[parsing started RegularFileObject[src/event/node/JunctionNode.java]]
[parsing completed 1ms]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Enum.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Comparable.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/Serializable.class)]]
[loading RegularFileObject[src/util/Time.java]]
[parsing started RegularFileObject[src/util/Time.java]]
[parsing completed 1ms]
```

```
[loading RegularFileObject[src/event/node/CheckpointNode.java]]
[parsing started RegularFileObject[src/event/node/CheckpointNode.java]]
[parsing completed 0ms]
[loading RegularFileObject[src/event/node/MedicalCheckpointNode.java]]
[parsing started RegularFileObject[src/event/node/MedicalCheckpointNode
   .java]]
[parsing completed 1ms]
[loading RegularFileObject[src/event/update/ArrivalUpdate.java]]
[parsing started RegularFileObject[src/event/update/ArrivalUpdate.java
   ]]
[parsing completed 0ms]
[loading RegularFileObject[src/event/update/DepartureUpdate.java]]
[parsing started RegularFileObject[src/event/update/DepartureUpdate.
   java]]
[parsing completed 0ms]
[loading RegularFileObject[src/event/update/ExcludedUpdate.java]]
[parsing started RegularFileObject[src/event/update/ExcludedUpdate.java
   ]]
[parsing completed 0ms]
[loading RegularFileObject[src/event/update/InvalidUpdate.java]]
[parsing started RegularFileObject[src/event/update/InvalidUpdate.java
   ]]
[parsing completed 0ms]
[loading RegularFileObject[src/event/update/TimeUpdate.java]]
[parsing started RegularFileObject[src/event/update/TimeUpdate.java]]
[parsing completed 1ms]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/FileNotFoundException.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/util/ArrayList.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Override.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/annotation/Annotation.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/annotation/Target.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/annotation/ElementType.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/annotation/Retention.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/annotation/RetentionPolicy.class)]]
[checking gui.Driver]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/AutoCloseable.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/System.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/PrintStream.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/FilterOutputStream.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/OutputStream.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Iterable.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/util/Collection.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/GraphicsConfiguration.class)]]
```

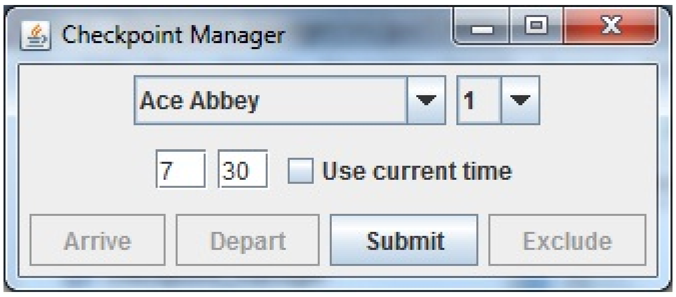
```
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/Frame.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/Window.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/Container.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/Component.class)]]
[loading RegularFileObject[src/gui/CheckpointPanel.java]]
[parsing started RegularFileObject[src/gui/CheckpointPanel.java]]
[parsing completed 3ms]
[loading RegularFileObject[src/util/FileIO.java]]
[parsing started RegularFileObject[src/util/FileIO.java]]
[parsing completed 1ms]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/event/ActionEvent.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/event/ActionListener.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/BorderLayout.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/util/Calendar.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/JButton.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/JCheckBox.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/JComboBox.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/JPanel.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/JTextField.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/accessibility/Accessible.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/JComponent.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/TransferHandler.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/
   TransferHandler$HasGetTransferHandler.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/image/ImageObserver.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/MenuContainer.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/util/EventListener.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/BufferedReader.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/File.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/FileOutputStream.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/FileReader.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/FileWriter.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/IOException.class)]]
```

```
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/nio/channels/FileLock.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/beans/ConstructorProperties.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Error.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/HeadlessException.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/UnsupportedOperationException.class)
   ]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/RuntimeException.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Exception.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Throwable.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/util/Iterator.class)]]
[wrote RegularFileObject[bin/gui/Driver.class]]
[checking event.Course]
[wrote RegularFileObject[bin/event/Course.class]]
[checking event.node.Node]
[wrote RegularFileObject[bin/event/node/Node.class]]
[checking event.Entrant]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/CloneNotSupportedException.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Class.class)]]
[wrote RegularFileObject[bin/event/Entrant$Status.class]]
[wrote RegularFileObject[bin/event/Entrant.class]]
[checking event.Event]
[wrote RegularFileObject[bin/event/Event.class]]
[checking event.update.Update]
[wrote RegularFileObject[bin/event/update/Update.class]]
[checking util.Time]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/StringBuilder.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/AbstractStringBuilder.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/CharSequence.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/StringBuffer.class)]]
[wrote RegularFileObject[bin/util/Time.class]]
[checking util.Parser]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/util/AbstractList.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/util/AbstractCollection.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Integer.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Number.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/NumberFormatException.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/IllegalArgumentException.class)]]
[wrote RegularFileObject[bin/util/Parser.class]]
```

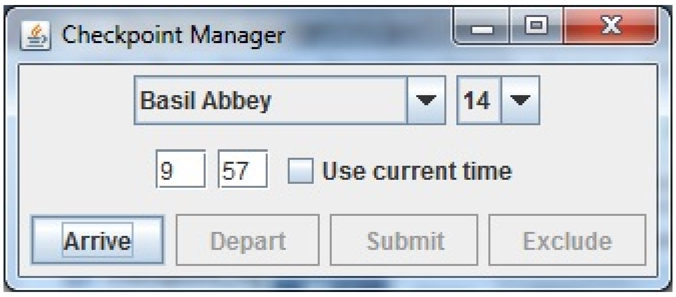
```
[checking event.node.JunctionNode]
[wrote RegularFileObject[bin/event/node/JunctionNode.class]]
[checking event.node.CheckpointNode]
[wrote RegularFileObject[bin/event/node/CheckpointNode.class]]
[checking event.node.MedicalCheckpointNode]
[wrote RegularFileObject[bin/event/node/MedicalCheckpointNode.class]]
[checking event.update.ArrivalUpdate]
[wrote RegularFileObject[bin/event/update/ArrivalUpdate.class]]
[checking event.update.DepartureUpdate]
[wrote RegularFileObject[bin/event/update/DepartureUpdate.class]]
[checking event.update.ExcludedUpdate]
[wrote RegularFileObject[bin/event/update/ExcludedUpdate.class]]
[checking event.update.InvalidUpdate]
[wrote RegularFileObject[bin/event/update/InvalidUpdate.class]]
[checking event.update.TimeUpdate]
[wrote RegularFileObject[bin/event/update/TimeUpdate.class]]
[checking gui.CheckpointPanel]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/LayoutManager.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/LayoutManager2.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/util/Vector.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/ComboBoxModel.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/PopupMenu.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/text/JTextComponent.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/Action.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/Icon.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/JToggleButton.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/javax/swing/AbstractButton.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/lang/Cloneable.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/awt/AWTEvent.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/util/EventObject.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/util/Set.class)]]
[wrote RegularFileObject[bin/gui/CheckpointPanel$Listener.class]]
[wrote RegularFileObject[bin/gui/CheckpointPanel.class]]
[checking util.FileIO]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/net/URI.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/FileDescriptor.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/InputStreamReader.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/io/Reader.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
   META-INF/sym/rt.jar/java/nio/channels/FileChannel.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/nio/channels/spi/
    AbstractInterruptibleChannel.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/nio/channels/SeekableByteChannel.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/nio/channels/ByteChannel.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/nio/channels/ReadableByteChannel.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/nio/channels/Channel.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/io/Closeable.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/nio/channels/WritableByteChannel.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/nio/channels/GatheringByteChannel.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/nio/channels/ScatteringByteChannel.class)
    ]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/nio/channels/InterruptibleChannel.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/io/OutputStreamWriter.class)]]
[loading ZipFileIndexFileObject[/usr/lib/jvm/java-7-openjdk/lib/ct.sym(
    META-INF/sym/rt.jar/java/io/Writer.class)]]
[wrote RegularFileObject[bin/util/FileIO.class]]
[total 411ms]
```

2.3 Example usage

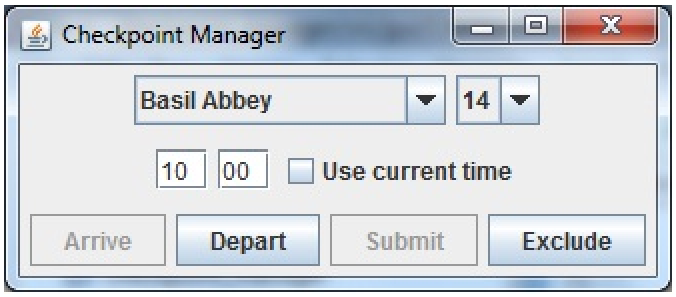
2.3.1 Checkpoint logging



2.3.2 Arriving at a medical checkpoint



2.3.3 Departing a medical checkpoint



2.3.4 File locking

The output from `lsof data/log.txt` while Event Manager is holding a read/write lock:

```
COMMAND   PID USER   FD   TYPE DEVICE SIZE/OFF   NODE NAME
main    31069  tom   3uW  REG    8,2       82 526665 data/log.txt
main    31069  tom   4uW  REG    8,2       82 526665 data/log.txt
```

3 Event Manager

3.1 Compiler output

```
tom@twoflower:~/cs22510-assignment/event_manager $ make main
cc -Wall -g -std=c89   -c -o node.o node.c
cc -Wall -g -std=c89   -c -o track.o track.c
cc -Wall -g -std=c89   -c -o course.o course.c
cc -Wall -g -std=c89   -c -o entrant.o entrant.c
cc -Wall -g -std=c89   -c -o event.o event.c
cc -Wall -g -std=c89   -c -o vector.o vector.c
cc -Wall -g -std=c89   -c -o util.o util.c
cc -Wall -g -std=c89    main.c node.o track.o course.o entrant.o
   event.o vector.o util.o   -o main
```

3.2 Example usage & results listing

```
tom@twoflower:~/cs22510-assignment/event_manager $ ./main
Please enter name file: ../data/name.txt
Please enter nodes file: ../data/nodes.txt
Please enter tracks file: ../data/tracks.txt
Please enter courses file: ../data/courses.txt
Please enter entrants file: ../data/entrants.txt
Please enter times file: ../data/times.txt
Please enter log file: ../data/log.txt


  Endurance Horse Race - The Main Event
  27th June 2012
  7:30


Please select from the following options:

   1. Locate a entrant
   2. Show how many entrants have not yet started
   3. Show how many entrants are currently on the course
   4. Show how many entrants have finished
   5. List entrants excluded for safety
   6. List entrants excluded for incorrect route
   7. Display results list
   8. Exit the program

12:55  >>  1
Enter entrant id: 1

    1: Ace Abbey
    Running course:        E
    Started at:            07:30
    Finished at:           09:34
    Total time:            124 mins

Please select from the following options:

   1. Locate a entrant
   2. Show how many entrants have not yet started
   3. Show how many entrants are currently on the course
   4. Show how many entrants have finished
   5. List entrants excluded for safety
   6. List entrants excluded for incorrect route
   7. Display results list
   8. Exit the program

12:55  >>  2
  0

Please select from the following options:

   1. Locate a entrant
   2. Show how many entrants have not yet started
   3. Show how many entrants are currently on the course
   4. Show how many entrants have finished
   5. List entrants excluded for safety
   6. List entrants excluded for incorrect route
   7. Display results list
   8. Exit the program

12:55  >>  3
  47

Please select from the following options:

   1. Locate a entrant
   2. Show how many entrants have not yet started
   3. Show how many entrants are currently on the course
   4. Show how many entrants have finished
   5. List entrants excluded for safety
   6. List entrants excluded for incorrect route
   7. Display results list
   8. Exit the program

12:55  >>  4
  46

Please select from the following options:

   1. Locate a entrant
   2. Show how many entrants have not yet started
   3. Show how many entrants are currently on the course
   4. Show how many entrants have finished
   5. List entrants excluded for safety
   6. List entrants excluded for incorrect route
   7. Display results list
   8. Exit the program

12:55  >>  5
  No entrants disqualified for safety reasons

Please select from the following options:

   1. Locate a entrant
   2. Show how many entrants have not yet started
   3. Show how many entrants are currently on the course
   4. Show how many entrants have finished
   5. List entrants excluded for safety
   6. List entrants excluded for incorrect route
   7. Display results list
   8. Exit the program
```

```
12:55  >>  6
  23: Beau Fudge
    Course: C Last node:  9
  28: Black Jack Fudge
    Course: A Last node: 13
  36: Bubbles Fudge
    Course: D Last node: 17
  41: Chalkie Fudge
    Course: F Last node:  7
  44: Copper Fudge
    Course: B Last node:  5
  46: Diamond Fudge
    Course: B Last node: 13
  59: Izzy Fudge
    Course: A Last node:  7
  68: Lemon Fudge
    Course: E Last node: 14
  78: Maddy Abbey
    Course: F Last node: 17

Please select from the following options:

   1. Locate a entrant
   2. Show how many entrants have not yet started
   3. Show how many entrants are currently on the course
   4. Show how many entrants have finished
   5. List entrants excluded for safety
   6. List entrants excluded for incorrect route
   7. Display results list
   8. Exit the program

12:55  >>  7

  Finished:
     26: Bella Fudge
      Course: F Total time: 109 mins
     27: Black Jack Abbey
      Course: F Total time: 109 mins
     48: Dinky Fudge
      Course: F Total time: 114 mins
     56: Honey Abbey
      Course: F Total time: 114 mins
     69: Lord Abbey
      Course: F Total time: 114 mins
     16: Barfields Marco Fudge
      Course: F Total time: 115 mins
     61: Jasmine Fudge
      Course: F Total time: 115 mins
      8: Ash Abbey
      Course: F Total time: 116 mins
      9: Ash Fudge
      Course: D Total time: 118 mins
     52: Ginger Fudge
      Course: F Total time: 118 mins
     22: Beau Abbey
      Course: D Total time: 122 mins
      6: April Abbey
      Course: D Total time: 123 mins
```

```
34: Bobby Fudge
 Course: E Total time: 123 mins
40: Chalkie Abbey
 Course: D Total time: 123 mins
76: Lord Abbey
 Course: D Total time: 123 mins
 1: Ace Abbey
 Course: E Total time: 124 mins
42: Copper Abbey
 Course: E Total time: 125 mins
47: Dinky Abbey
 Course: E Total time: 130 mins
 5: Amber Fudge
 Course: E Total time: 131 mins
55: Goldie Fudge
 Course: E Total time: 132 mins
74: Lucky Fudge
 Course: E Total time: 132 mins
70: Lord Fudge
 Course: E Total time: 134 mins
19: Beatrice Abbey
 Course: C Total time: 147 mins
45: Diamond Abbey
 Course: C Total time: 149 mins
65: Lady Tara Abbey
 Course: C Total time: 149 mins
12: Autumn Abbey
 Course: C Total time: 150 mins
35: Bubbles Abbey
 Course: C Total time: 152 mins
51: Ginger Abbey
 Course: C Total time: 152 mins
50: Ebony Fudge
 Course: C Total time: 155 mins
57: Honey Fudge
 Course: C Total time: 156 mins
 4: Amber Abbey
 Course: C Total time: 157 mins
30: Blue Abbey
 Course: B Total time: 163 mins
31: Blue Fudge
 Course: B Total time: 164 mins
 7: April Fudge
 Course: B Total time: 166 mins
17: Basil Abbey
 Course: B Total time: 169 mins
39: Captain Fudge
 Course: B Total time: 171 mins
13: Autumn Fudge
 Course: B Total time: 173 mins
24: Bella Abbey
 Course: B Total time: 174 mins
49: Ebony Abbey
 Course: B Total time: 184 mins
10: Asti Abbey
 Course: A Total time: 229 mins
14: Barfields Marco Abbey
 Course: A Total time: 229 mins
18: Basil Fudge
```

```
     Course: A Total time: 230 mins
    20: Beatrice Fudge
     Course: A Total time: 230 mins
    11: Asti Fudge
     Course: A Total time: 231 mins
     3: Ace Fudge
     Course: A Total time: 232 mins
    32: Bobby Abbey
     Course: A Total time: 232 mins

Running:
    38: Captain Abbey
     Course: A Track:  1 Run time: 225 mins
    53: Goldie Abbey
     Course: A Track: 18 Run time: 187 mins
    58: Izzy Abbey
     Course: A Track: 17 Run time: 169 mins
    62: Lady Abbey
     Course: D Track:  1 Run time: 163 mins
    60: Jasmine Abbey
     Course: A Track: 17 Run time: 162 mins
    64: Lady Fudge
     Course: B Track: 13 Run time: 157 mins
    66: Lady Tara Fudge
     Course: B Track: 13 Run time: 149 mins
    67: Lemon Abbey
     Course: B Track: 13 Run time: 145 mins
    71: Lucky Abbey
     Course: A Track: 21 Run time: 133 mins
    77: Lord Fudge
     Course: B Track: 17 Run time: 123 mins
    79: Maddy Fudge
     Course: A Track: 18 Run time: 116 mins
    80: Magic Abbey
     Course: D Track:  1 Run time: 115 mins
    81: Magic Fudge
     Course: D Track:  1 Run time: 111 mins
    83: Major Abbey
     Course: A Track: 17 Run time: 108 mins
    85: Major Fudge
     Course: A Track: 17 Run time: 104 mins
    86: Mattie Abbey
     Course: B Track: 17 Run time: 101 mins
    87: Mattie Fudge
     Course: A Track: 15 Run time:  98 mins
    89: Prince Abbey
     Course: B Track: 15 Run time:  94 mins
    90: Prince Fudge
     Course: A Track: 15 Run time:  91 mins
    91: Princess Abbey
     Course: B Track:  8 Run time:  87 mins
    92: Princess Fudge
     Course: B Track:  8 Run time:  84 mins
    93: Rosie Abbey
     Course: D Track: 11 Run time:  81 mins
    94: Rosie Fudge
     Course: B Track:  8 Run time:  78 mins
    95: Ruby Abbey
     Course: F Track: 13 Run time:  75 mins
    97: Ruby Fudge
     Course: C Track:  7 Run time:  72 mins
    98: Sapphire Abbey
     Course: C Track:  8 Run time:  69 mins
   100: Sapphire Fudge
     Course: F Track: 12 Run time:  66 mins
   101: Scarlet Abbey
     Course: C Track:  5 Run time:  63 mins
   102: Scarlet Fudge
     Course: F Track: 12 Run time:  60 mins
   103: sienna Abbey
     Course: D Track:  5 Run time:  56 mins
   106: sienna Fudge
     Course: B Track:  5 Run time:  53 mins
   107: Silver Abbey
     Course: F Track: 12 Run time:  50 mins
   108: Silver Fudge
     Course: A Track:  4 Run time:  47 mins
   109: Smokey Abbey
     Course: A Track:  4 Run time:  44 mins
   110: Smokey Fudge
     Course: D Track:  4 Run time:  41 mins
   111: Snowy Abbey
     Course: E Track: 11 Run time:  38 mins
   113: Snowy Fudge
     Course: C Track:  3 Run time:  35 mins
   114: sonic Abbey
     Course: A Track:  3 Run time:  32 mins
   115: sonic Fudge
     Course: D Track:  2 Run time:  29 mins
   117: Summer Abbey
     Course: A Track:  2 Run time:  25 mins
   118: Summer Fudge
     Course: E Track:  2 Run time:  22 mins
   121: Tango Abbey
     Course: B Track:  1 Run time:  19 mins
   122: Tango Fudge
     Course: A Track:  1 Run time:  16 mins
   123: Topaz Abbey
     Course: B Track:  1 Run time:  13 mins
   124: Topaz Fudge
     Course: F Track:  1 Run time:  10 mins
   126: Zizou Abbey
     Course: D Track:  1 Run time:   6 mins
   127: Zizou Fudge
     Course: F Track:  1 Run time:   3 mins

Disqualified:
    28: Black Jack Fudge
     Course: A Disqualified for incorrect route
    44: Copper Fudge
     Course: B Disqualified for incorrect route
    68: Lemon Fudge
     Course: E Disqualified for incorrect route
    78: Maddy Abbey
     Course: F Disqualified for incorrect route
    41: Chalkie Fudge
     Course: F Disqualified for incorrect route
    46: Diamond Fudge
```

```
      Course: B Disqualified for incorrect route
     23: Beau Fudge
      Course: C Disqualified for incorrect route
     59: Izzy Fudge
      Course: A Disqualified for incorrect route
     36: Bubbles Fudge
      Course: D Disqualified for incorrect route

Please select from the following options:

   1. Locate a entrant
   2. Show how many entrants have not yet started
   3. Show how many entrants are currently on the course
   4. Show how many entrants have finished
   5. List entrants excluded for safety
   6. List entrants excluded for incorrect route
   7. Display results list
   8. Exit the program

12:55  >>  8
```

3.3 Generated log file

```
CM: processed 198 updates from ../data/times.txt
EM: entrant query
EM: entrants listed - disqualified
CM: D type event recorded
EM: entrant query
EM: quitting
```

4 Program descriptions

4.1 Event Creator

The Event Creator program has been implemented in C++. It allows the user to create a new event, add competitors to an event and create courses for an event. It does virtually no error checking (it will crash if it is given the wrong format for data e.g. a string instead of an int). I feel this is its greatest short-coming.

4.2 Checkpoint Manager

The Checkpoint Manager has been implemented in Java and makes use of the Swing framework. It allows the user to update the location of an entrant and performs very simple error checking to ensure that an entrant cannot be logged as arriving at a medical checkpoint twice, for example. It includes an option to use the current time for each update. It locks both the times file and log file when writing.

4.3 Event Manager

The Event Manager has been implemented in C (based on the previous CS237 assignment). It allows the user to query the status of an individual entrant, list entrants in various states of competition and list the results (in sorted format). It also locks the log file when writing.