Capstone Documentation

Release

Paul Vincent Craven

Oct 06, 2017

Contents

Contents:

CHAPTER 1

Assignment 1 - Zukei Puzzle

Description

In a Zukei puzzle, the player is given a set of seemingly random coordinates in the Cartesian plane. The player must find a geometric shape whose vertices are in the set. In this problem, the player must find only rectangles:

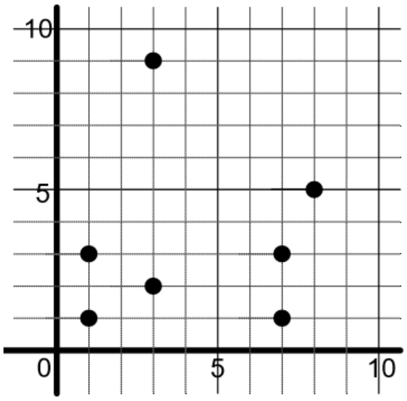

Fig. 1.1: Random coordinates?

Identify which vertices form rectangles in several Zukei puzzles given as input. Assume that only one rectangle is hidden within each puzzle and that rectangle sides are aligned with the X and Y axes - they are not rotated.

Input

The first line contains the number of puzzles. Each of the following lines contain the information about a puzzle. A puzzle line starts with the number of 2-D vertices followed by pairs of integers for the vertex coordinates, listed in XY order. For example, the first puzzle in the example input below has 7 vertices. The first vertex is (1, 1), and the final vertex is (8, 5).

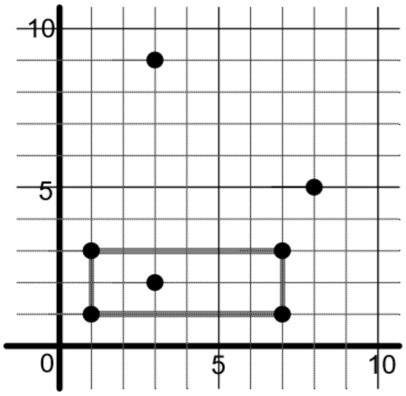

Fig. 1.2: Nope, a rectangle!

```
2
7 1 1 7 1 7 3 1 3 3 9 3 2 8 5
8 4 7 8 4 7 4 4 1 5 9 2 8 5 4 8 9
```

Output

For each puzzle, print to standard output a case label and the vertices that form the rectangle in the puzzle. Lower vertices (i.e., smaller Y) appear before higher ones. In the case of a tie, the vertex farther to the left (i.e., smaller X) is printed first. For the example input given above, the output is:

```
Case 1: 1 1 7 1 1 3 7 3
Case 2: 5 4 8 4 5 9 8 9
```

Turn In

Turn in a Java or Python file that will accept input via a file or `stdio`, and output the result via a file or `stdout`.

CHAPTER 2

Assignment 2 - Node.js Report to Management

Goal

Ok, time to take what you learned from the last three assignments and practice it with one more technology.

This assignment is designed to test your ability to go out and do research new technologies. By itself, this isn't useful unless you can communicate those findings to management. With Assignment 1 you need to do both.

Details

Your bosses at Fine Iowa Car Loans (FICL) have heard a lot about this new "node.js". But your company doesn't yet have anything to do with it. Should they? It is your job to find out.

Write a report to your technology managers on Node.js. The report should provide a factual, balanced view of the technology.

Include the following in your report:

- Report title, date, version, author - Make sure the managers know who wrote the report, when, and what version of the report they are looking at. (As things often get updated.)
- Executive summary paragraph - Write this last. Give a quick one-paragraph summary of what you came up with.
- What is Node.js?
- What does Node.js do?
- History of Node.js. Who wrote it, how long has it been around, etc.
- Opportunities for using Node.js. What could the company use Node.js for?
- Risks in using Node.js. Is it too new? Outdated? Subject to change? No vendor to stand behind it?
- Performance considerations with Node.js? Does it run fast? Does it take a long time to download?
- Scalability of using Node.js. If we have 1,000 people on our web server, will this help or hurt us?

- What support is out their for Node.js? If something goes wrong, who will we get support from?
- Popularity: Can we find Node.js programmers? Is it a fad? Is it getting old and out of date?
- Your conclusions on Node.js.
- References (Where did you get your info? Back it your information with citations from reputable publications.)

References

When listing references, don't just list URLs. You can hyper-link the title to the URL. Here is an example of what I'm looking for:

- GManNickG. Why is it faster to process a sorted array than an unsorted array? Stack Overflow, Jun 27 2012. Retrieved Jan 16, 2017.

Double-Check

When you are done, check the following:

- Make sure that you use paragraphs. No one will read long walls of text. Follow the PIE methodology for paragraphs.
- Bullet lists, numbered lists, these are ok to use. Write for a technical manager that wants to be able to skim the report.

Note that in two weeks you'll write a different report. The next report will not be to your managers, but to other programmers.

CHAPTER 3

Assignment 3 - Node.js Proof-of-Concept

Goal

See if you can download, install, and follow tutorials on how to get a new technology working at a proof-of-concept level.

Details

Create a proof-of-concept on how to use NodeJS. Theme it using our Fine Iowa Car Loans company. Show how to get something useful working for our car loan company. Like a simple website for ICL, or a calculator for loan payments, or something similar.

There are tutorials out there you can find. TutorialsPoint Node.js tutorial is good.

For this tutorial, you may copy/paste material from the web. **However** to not count as plagiarism you must cite where you got information from. Include the citation in comments before the code. Like:

```
// The following code was adapted from the TutorialsPoint example at:
// https://www.tutorialspoint.com/nodejs/
```

Zip up the files and upload, or link to the proof of concept on GitHub.

Next week you will create your own tutorial stepping through how to create a Node.js application.

CHAPTER 4

Assignment 4 - Node.js Tutorial/Technical Report

Goal

In assignment 4, we had you writing at a high-level to your technical managers. Now the goal is to write a lower level explanation to your fellow programmers.

Details

Take the proof-of-concept that you created, and make a tutorial based off it.

- Use the “Snip” tool, the Alt-Print-screen combination, or some other tool to make screen shots.
- During the tutorial, explain concepts like REPL, callbacks, event loop, etc.

CHAPTER 5

Assignment 5 - Database Diagramming

Goal

Demonstrate your proficiency in diagramming a relational schema for a database.

Details

In this assignment, you will create an relational schema diagram for a database that will store academic information for a college.

- Have a common table to track people
- Track which people are students
- Track which people are instructors
- Track which people are alumni
- Track classes (Like CMSC 150, not ties to a particular time)
- Track class sections (A particular section of a class, like CMSC 150 A for Fall 2018)
- Track class section registrations
- Track who is in what section, with what room, and who teaches
- Track grades
- Track rooms (what classes are taught in what rooms)
- Track semesters (just fall and spring)

Make sure you identify:

- Tables
- Fields in the table

- Data types
- Primary keys
- Foreign keys
- Relationships (One-to-one, zero-too-many-to-one, etc.)

Turn in the diagram. If you are drawing by hand, upload a photo of the drawing. Make sure it is neat enough I can read it.

CHAPTER 6

Assignment 6 - Cloud Services Part 1

Goal

Learn and experiment with a new-to-you cloud service. Both this week and next week I'm looking for you to:

- Select a new cloud service **you haven't used before**.
- Get it up and running.
- Write about what it is, and how you got it working.

Some suggestions:

- Create a web server (Use EC2 or Elastic Beanstalk)
- Create a database (RDS)
- Set up your own name server (Route 53)
- Learn how to set up storage, and serve files via the cloud (S3)
- Monitor services (CloudWatch)
- Learn how to set up load balancers

Details

If you haven't already, sign up for [Amazon Web Servives](https://aws.amazon.com/). You can also use [Microsoft Azure](https://azure.microsoft.com/en-us/) if you'd rather. I'm familiar with AWS, but not Azure. So if you want any help, use AWS.

Select one of the cloud services to learn more about.

If you've used cloud services before, there's probably a service you haven't had a chance to explore before.

If you haven't used cloud services, I'd suggest trying EC2 first. EC2 allows you to create your own virtual server. You can work with Linux or Windows. RDS is another good choice as it allows you to set up your own database. These two are likely the "bread and butter" services.

Elastic Beanstalk helps you set up a web service on EC2, and is another good choice.

Grading

- Introduction (30% of the grade)
  - Describe what the service is.
  - Describe how it could be used.
  - Reflect on how you could use it personally.
- Demonstration (40% of the grade)
  - Show what you were able to create. Screen shots are encouraged. (Use the Windows "Snipping Tool", Snag-It, or similar.)
- Tutorial (30% of the grade)
  - Talk about the basics of getting the service up and running.

Cost of Services

While these services require a credit card, if you are careful you can select "free tier" services and not have to pay for it.

It is *very important* to shut down the services after you are done using them. You don't want any surprise charges.

CHAPTER 7

Assignment 7 - Cloud Services Part 2

Goal

Learn and experiment with a new-to-you cloud service. Both this week and next week I'm looking for you to:

- Select a new cloud service **you haven't used before**.
- Get it up and running.
- Write about what it is, and how you got it working.

Some suggestions:

- Create a web server (Use EC2 or Elastic Beanstalk)
- Create a database (RDS)
- Set up your own name server (Route 53)
- Learn how to set up storage, and serve files via the cloud (S3)
- Monitor services (CloudWatch)
- Learn how to set up load balancers

Details

If you haven't already, sign up for [Amazon Web Servives](https://aws.amazon.com/). You can also use [Microsoft Azure](https://azure.microsoft.com/en-us/) if you'd rather. I'm familiar with AWS, but not Azure. So if you want any help, use AWS.

Select one of the cloud services to learn more about.

If you've used cloud services before, there's probably a service you haven't had a chance to explore before.

If you haven't used cloud services, I'd suggest trying EC2 first. EC2 allows you to create your own virtual server. You can work with Linux or Windows. RDS is another good choice as it allows you to set up your own database. These two are likely the "bread and butter" services.

Elastic Beanstalk helps you set up a web service on EC2, and is another good choice.

Grading

- Introduction (30% of the grade)
  - Describe what the service is.
  - Describe how it could be used.
  - Reflect on how you could use it personally.
- Demonstration (40% of the grade)
  - Show what you were able to create. Screen shots are encouraged. (Use the Windows "Snipping Tool", Snag-It, or similar.)
- Tutorial (30% of the grade)
  - Talk about the basics of getting the service up and running.

Cost of Services

While these services require a credit card, if you are careful you can select "free tier" services and not have to pay for it.

It is *very important* to shut down the services after you are done using them. You don't want any surprise charges.

CHAPTER 8

Assignment 8 - Data Structures 1

Assignment

For each program below:

- Tell me what the program does. Show what it prints.
- Explain the results of the program. What is it doing and printing?
- Try to explain WHY the results are what they are. If you don't know for certain, create a hypothesis.
- This has a lot to do with data structures. Make sure your explanation does dive into the particulars of the data structures and what is happening. Don't give a "surface" explanation, make sure you understand. Ask me questions if needed or check to make sure your are in the right ballpark. This assignment is the easiest to get a bad grade on if you don't check to make sure.

Part 1

Listing 8.1: Test01.java

```java
import java.util.LinkedList;

public class Test01 {
    private static int n = 10000000;

    private static void link_list_add_test() {
        LinkedList<Integer> list = new LinkedList<Integer>();

        for(int i=0; i < n; i++) {
            // Look up the documentation for the following command
            list.add(0, 0);
        }
    }

    public static void main(String [] args) {
        System.out.println("Starting test...");

        double total_duration = 0.0;
        for(int test=0; test < 20; test++) {
            long startTime = System.nanoTime();
            link_list_add_test();
            long endTime = System.nanoTime();
            double duration = (endTime - startTime) / 1000000000.;
            total_duration += duration;
            System.out.println(String.format("Test took %f seconds.", duration));
        }
        System.out.println(String.format("Total test took %f seconds.", total_
↪duration));

    }
}
```

Part 2

Listing 8.2: Test02.java

```
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.Stack;

public class Test02 {
    private static int n = 10000000;

    private static void link_list_add_test() {
        LinkedList<Integer> list = new LinkedList<Integer>();

        for(int i=0; i < n; i++) {
            // Note, this command is different than the one in
            // the prior example!
            list.add(0);
        }
    }

    private static void array_add_test() {
        ArrayList<Integer> list = new ArrayList<Integer>();

        for(int i=0; i < n; i++) {
            list.add(0);
        }
    }

    private static void stack_add_test() {
        Stack<Integer> stack = new Stack<Integer>();

        for(int i=0; i < n; i++) {
            stack.add(0);
        }
    }
    public static void main(String [] args) {
        System.out.println("Warming up...");
        for(int test=0; test < 20; test++) {
            array_add_test();
        }

        double total_duration = 0.0;
        System.out.println("Starting array_add_test test...");
        for(int test=0; test < 20; test++) {
            long startTime = System.nanoTime();
            array_add_test();
            long endTime = System.nanoTime();
            double duration = (endTime - startTime) / 1000000000.;
            total_duration += duration;
            System.out.println(String.format("array_add_test test took %f seconds.", duration));
        }
        System.out.println(String.format("Total array_add_test took %f seconds.", total_duration));

        total_duration = 0.0;
        System.out.println("Starting link_list_add_test test...");
        for(int test=0; test < 20; test++) {
            long startTime = System.nanoTime();
            link_list_add_test();
            long endTime = System.nanoTime();
            double duration = (endTime - startTime) / 1000000000.;
            total_duration += duration;
            System.out.println(String.format("link_list_add_test test took %f seconds.", duration));
        }
        System.out.println(String.format("Total link_list_add_test took %f seconds.", total_duration));

        total_duration = 0.0;
        System.out.println("Starting stack_add_test test...");
        for(int test=0; test < 20; test++) {
            long startTime = System.nanoTime();
            stack_add_test();
            long endTime = System.nanoTime();
            double duration = (endTime - startTime) / 1000000000.;
            total_duration += duration;
            System.out.println(String.format("stack_add_test test took %f seconds.", duration));
        }
        System.out.println(String.format("Total stack_add_test took %f seconds.", total_duration));

    }
}
```

Part 3

Listing 8.3: Test03.java

```java
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.Stack;

public class Test04 {
    public static int n = 100000;

    public static void link_list_access_test() {
        LinkedList<Integer> list = new LinkedList<Integer>();

        for(int i=0; i < n; i++) {
            list.add(0, 0);
        }

        long startTime = System.nanoTime();
        for(int i = 0; i < n; i++) {
            int x = list.get(i);
        }
        long endTime = System.nanoTime();
        double duration = (endTime - startTime) / 1000000000.;
        System.out.println(String.format("link_list_access_test test took %f seconds.", duration));
    }

    public static void array_access_test() {
        ArrayList<Integer> list = new ArrayList<Integer>();

        for(int i=0; i < n; i++) {
            list.add(0,0);
        }
        long startTime = System.nanoTime();
        for(int i = 0; i < n; i++) {
            int x = list.get(i);
        }
        long endTime = System.nanoTime();
        double duration = (endTime - startTime) / 1000000000.;
        System.out.println(String.format("array_add_test test took %f seconds.", duration));
    }

    public static void stack_access_test() {
        Stack<Integer> stack = new Stack<Integer>();

        for(int i=0; i < n; i++) {
            stack.add(0, 0);
        }
        long startTime = System.nanoTime();
        for(int i = 0; i < n; i++) {
            int x = stack.get(i);
        }
        long endTime = System.nanoTime();
        double duration = (endTime - startTime) / 1000000000.;
        System.out.println(String.format("stack_access_test test took %f seconds.", duration));
    }

    public static void main(String [] args) {
        link_list_access_test();
        link_list_access_test();
        array_access_test();
        stack_access_test();
    }
}
```

CHAPTER 9

Assignment 9 - Data Structures 2

Setup

Download the zip file `Data Structures.zip` and expand out Test05.java, dictionary.txt, and dracula.txt.

Experiments

Run the Java program with the following data types:

- HashSet
- LinkedList
- TreeSet
- ArrayList
- Stack

Results

Present and explain the following:

- What does the program do?
- Why does the program read the whole story into memory before spell-checking?
- What are the timings of the different data structures?
- Why are those data structures timed the way they are? Make sure you go back to your data structures days, and explain how the data structure is impacting the performance. Show you understand how the data structures work.
- What is the difference between a Set and a Map?

CHAPTER 10

Assignment 10 - Final Project Draft

Goal

This is your final project for the capstone. This assignment is worth twice the points as the other assignments.

This assignment 10 is an ungraded draft of your final project.

Details

For the final project, pick your own subject to research. The research must involve learning something new-to-you, and include a component where you create some sample code.

Some possible things to learn about:

- [D3](https://d3js.org/) Graph Library for JavaScript
- [React](https://facebook.github.io/react/) UI library for JavaScript
- [Phaser.io](http://phaser.io/) Game library for JavaScript
- [libgdx](https://libgdx.badlogicgames.com/) Game library for Java
- [Haskell](https://www.haskell.org/)Programming language
- [Arduino](https://www.arduino.cc/) Using the Arduino
- UNIX command line tools, including grep, sed, and awk
- [Powershell](https://msdn.microsoft.com/en-us/powershell/mt173057.aspx) Microsoft scripting language

This is a pretty open research paper. Please feel free to drop me a note to double-check that a subject would be good.

Grading

- Introduction (30% of the grade)
    - Describe what the topic is.
    - Describe how it could be used.
    - Reflect on how you could use it personally.
- Demonstration (60% of the grade)
    - Show what you were able to create. Screen shots are encouraged. (Use the Windows “Snipping Tool”, Snag-It, or similar.)
    - Show actual code that you’ve written.
- Conclusion (10% of the grade)
    - Where do you see this technology going? Is it useful?

CHAPTER 11

Assignment 11 - Final Project

Goal

This is your final project for the capstone. This assignment is worth twice the points as the other assignments. There is also a rough-draft turn-in part way through.

Details

For the final project, pick your own subject to research. The research must involve learning something new-to-you, and include a component where you create some sample code.

Some possible things to learn about:

- [D3](https://d3js.org/) Graph Library for JavaScript
- [React](https://facebook.github.io/react/) UI library for JavaScript
- [Phaser.io](http://phaser.io/) Game library for JavaScript
- [libgdx](https://libgdx.badlogicgames.com/) Game library for Java
- [Haskell](https://www.haskell.org/)Programming language
- [Arduino](https://www.arduino.cc/) Using the Arduino
- UNIX command line tools, including grep, sed, and awk
- [Powershell](https://msdn.microsoft.com/en-us/powershell/mt173057.aspx) Microsoft scripting language

This is a pretty open research paper. Please feel free to drop me a note to double-check that a subject would be good.

Grading

- Introduction (30% of the grade)

  - Describe what the topic is.
  - Describe how it could be used.
  - Reflect on how you could use it personally.

- Demonstration (60% of the grade)
  - Show what you were able to create. Screen shots are encouraged. (Use the Windows “Snipping Tool”, Snag-It, or similar.)
  - Show actual code that you’ve written.
- Conclusion (10% of the grade)
  - Where do you see this technology going? Is it useful?