UIL Computer Science Concepts

Hands On Element - The First Steps

Written by
Kirby Rankin

Edited by Linda Tarrant and Nancy Barnard

Author Kirby Rankin brings over 25 years of teaching experience and has coached Computer Science for most of those years. His successes include three individual champions and six 2A team champions, and these were in a row from 2008 through 2013. He had many, many more competitors qualify for region and state during his years teaching.

We are a small company that listens! If you have any questions or if there is an area that you would like fully explored, let us hear from you. We hope you enjoy this product and stay in contact with us throughout your academic journey. Linda Tarrant, President Hexco Inc.

HEXCO ACADEMIC
www.hexco.com
P.O. Box 199 • Hunt, Texas 78024
Phone: 800.391.2891 • Fax: 830.367.3824
Email: hexco@hexco.com

Introduction

There are two parts to the UIL Computer Science contest. There is a written exam that contains forty multiple choice and free response questions and a set of twelve programming problems for each team to attempt to code. This guide is intended to get contestants prepared to participate in the programming portion of the contest. While it is possible to read this guide and get a partial understanding of Java programming in general, it will greatly enhance your understanding to be enrolled in a high school computer science course of some kind. Also, reading and thoroughly studying the Hexco guide titled ***"UIL Computer Science – The First Fifteen"*** will improve your chances of being successful during the programming portion of the contest.

There are two goals for this guide. ***First,*** to get teams prepared to participate. That is, how to show up at a contest with the proper equipment, software and a basic knowledge of how the contest will be run as well as what to expect and how to deal with it. The ***second*** goal is to provide teams with enough programming examples to successfully code the easier two or three problems in the problem set.

While it may not seem to be very competitive to solve only two or three problems in a twelve-problem set, consider the statistics presented in this table from 2015 UIL Computer Science district contests.

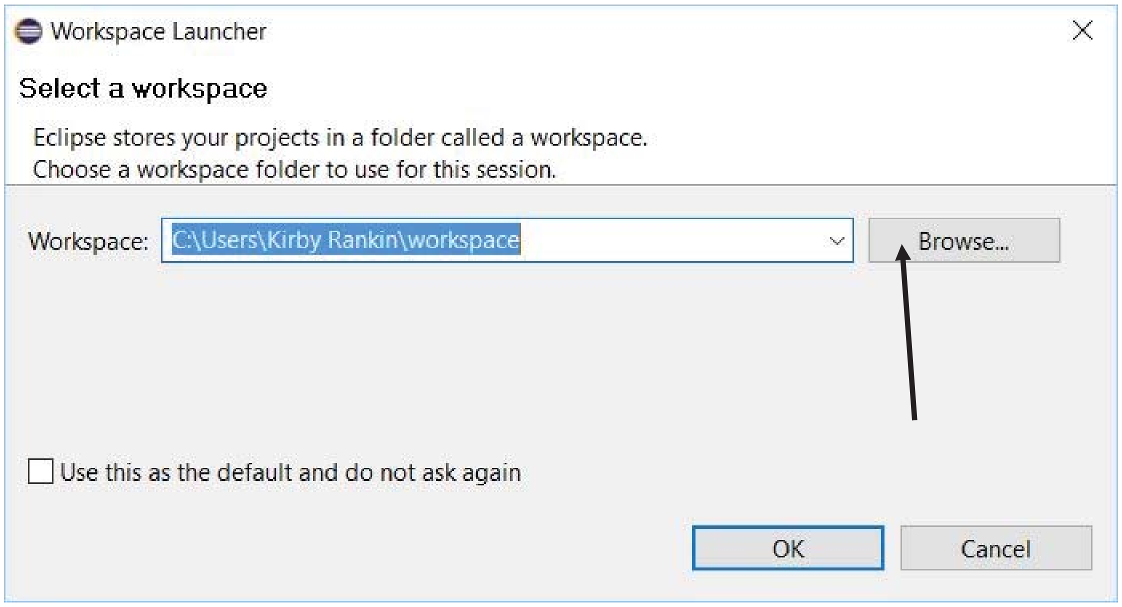
Percent of district Computer Science contests where a team score of 270 or above would have earned a team medal (3rd place or better).	
Classification	Percent
1A	100%
2A	100%
3A	100%
4A	88%
5A	72%
6A	34%

Given that one half of a team's score comes from the top three individual scores on the written portion of the test and that each of those scores could be as high as 240 points, it becomes very clear that successfully completing just a few of the programs can lead to great success at the district level.

Of course, to be competitive at the region and state level teams will have to score many more than 270 points. We will leave that level of expertise for another guide. This guide is all about getting off to a good start.

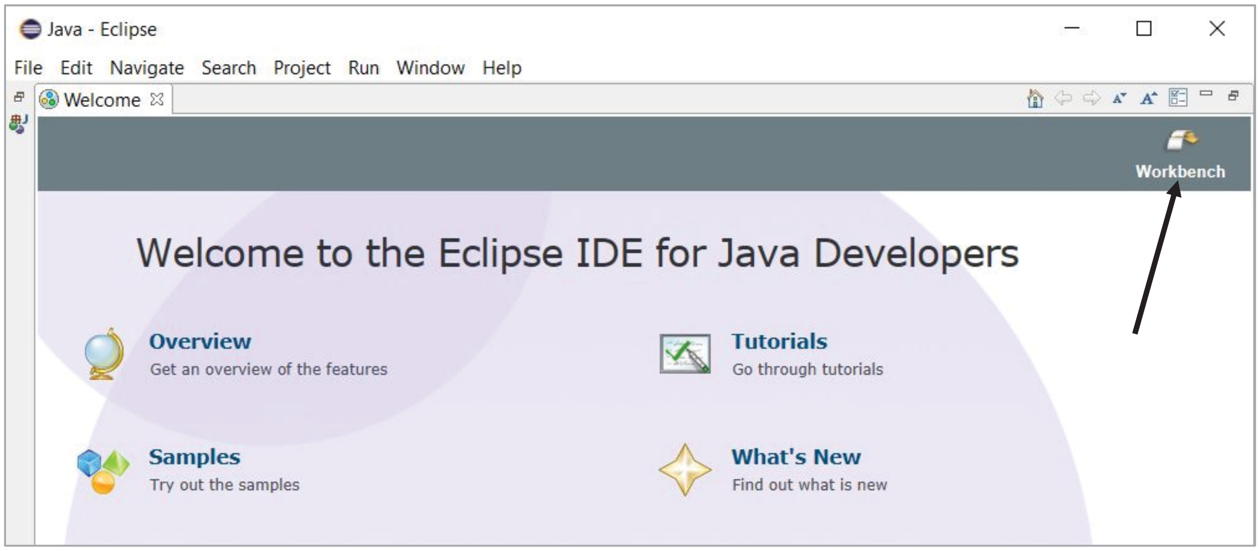
Here we go!

Table of Contents



Workspace Launcher
Select a workspace
Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.
Workspace: C:\Users\Kirby Rankin\workspace
Browse...
Use this as the default and do not ask again
OK
Cancel

After you have created a *Workspace*, Eclipse will take you to the Welcome Screen. You may explore the samples or tutorials, if you like, but when you are ready to write some Java code, click on *Workbench*.



Java - Eclipse
File Edit Navigate Search Project Run Window Help
Welcome
Workbench
Welcome to the Eclipse IDE for Java Developers
Overview
Get an overview of the features
Tutorials
Go through tutorials
Samples
Try out the samples
What's New
Find out what is new

After you click on Workbench, you will get to the Eclipse IDE workspace. This includes a title bar, menu, a toolbar and several windows. Not all of the windows are crucial for UIL competition.

A shot is considered a bullseye if the bullet strikes any part of the circular bullseye.

The center of the bullseye and thus the center of the target is located at 0.0, 0.0.

Example Input File

```
5
1.5 0.3
0.1 0.3
0.0 0.0
2.5 3.8
0.0 0.5
```

Example Output to Screen

```
Miss
Bullseye
Bullseye
Miss
Bullseye
```

Using An Array (and some REALLY big numbers)

11. Planetary Travel

Program Name: Planets.java **Input File: planets.dat**

It is the year 3016 and space travel has become a reality. The largest passenger space travel company is SpaceTrekker. At SpaceTrekker you can purchase a ticket to travel between any one of the eight planets and Pluto. To calculate the fare between each planet SpaceTrekker needs to know the distance between each planet. Write a program that will do that calculation for them.

Input

The file will begin with nine (9) lines each containing the name of a planet (and Pluto) and its distance from the ***sun*** in kilometers. These nine lines will be followed by a number *t* representing the number of trips taken between planets. Each of the *t* lines will contain the names of the departure planet and the destination planet separated by one space.

Output

Your program should print the planet of departure and the destination planet separated by a space and followed by a space then the distance in kilometers travelled on each trip between planets followed by a space, the units of measure (kilometers) and a period.

Constraints

1 <= t <= 20

Example Input File

```
Mercury  57910000
Venus 108200000
Earth 149600000
Mars  227940000
Jupiter  778330000
Saturn   1424600000
Uranus   2873550000
Neptune  4501000000
Pluto 5945900000
5
Mars Earth
Earth Uranus
Neptune Mercury
Jupiter Saturn
Mercury Pluto
```

Example Output to Screen

```
Mars to Earth is 78340000 kilometers.
Earth to Uranus is 2723950000 kilometers.
Neptune to Mercury is 4443090000 kilometers.
Jupiter to Saturn is 646270000 kilometers.
Mercury to Pluto is 5887990000 kilometers.
```

Working With Text And Characters

---

12. Clean Up Those Numbers

Program Name: Phone.java **Input File: phone.dat**

PhoneBooksRUs Inc. needs to clean up a list of phone numbers that they have stored in a file. The numbers are listed in a variety of formats and some of the numbers in the file are not valid phone numbers. Your job is to write a program to read the phone numbers and then print a list of the valid numbers in the proper format.

Input

The file will contain an unknown number of possible phone numbers, each on a separate line. A valid phone number entry will have one of the following formats: