Technologies in
Computer Science Education

Jade Cheng • April 20, 2012

Introduction & Overview

Computer Science (CS) is a diverse and rapidly developing discipline.

Computer Science Education (CSE) is the subject specific to educational research for CS.

The main motivation of CSE:

> To improve the quality and efficiency of teaching and learning for the subject matter.
>
> To measure the success of this work.

This presentation will consist of two parts.

> First, we will examine some software tools and techniques developed in my CSE practices.
>
> Second, I will discuss my research interests in CSE and my potential and ongoing projects.

CSE Technologies in Practice

Grading Tools for Programming Assignments

Learning to program is one of the most important goals in CS.

Grading programming assignments and delivering feedback can be difficult:

- tedious
- subjective
- error-prone

To address this, I have designed several software tools, which have been used with success in several programming courses, including CSCI 2912 at HPU.

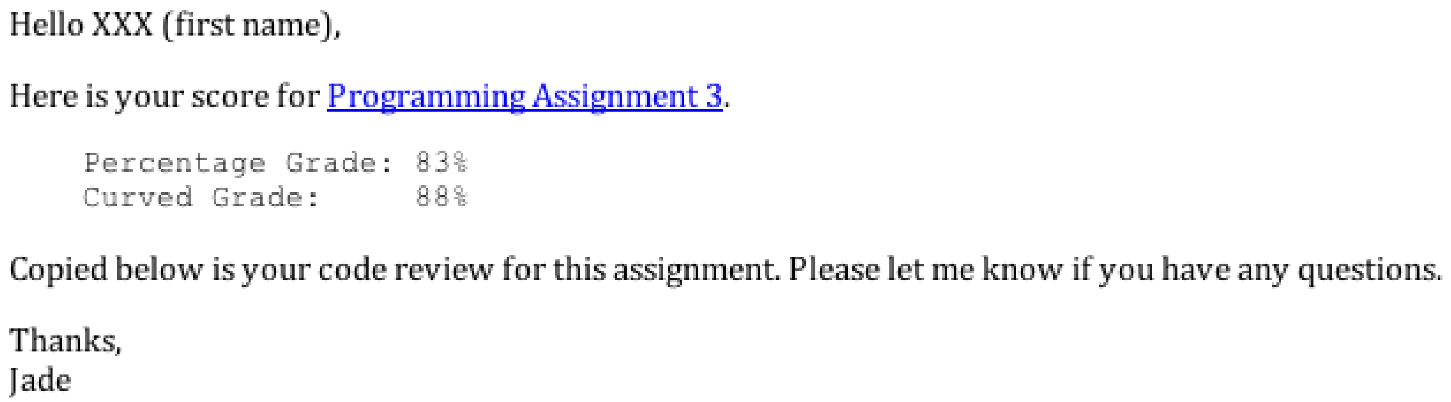
Technologies:

- Java
- XML
- HTML/CSS

Grading Tools for Programming Assignments

The Deliverables

First, deliver the scores, before and after curving.



Hello XXX (first name),

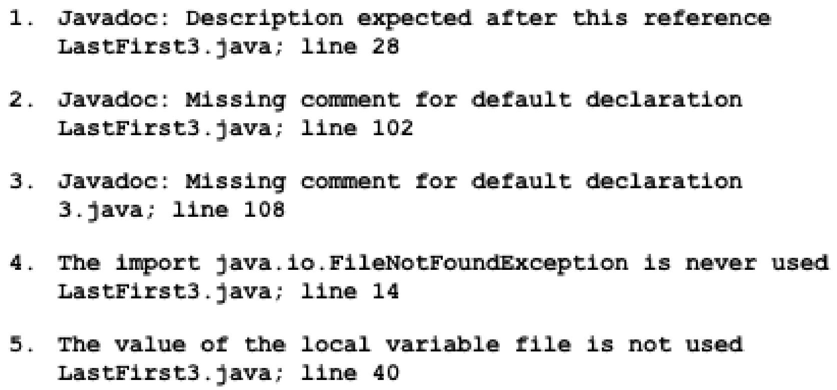
Here is your score for Programming Assignment 3.

```
Percentage Grade: 83%
Curved Grade:     88%
```

Copied below is your code review for this assignment. Please let me know if you have any questions.

Thanks,
Jade

Second, report errors and warnings.

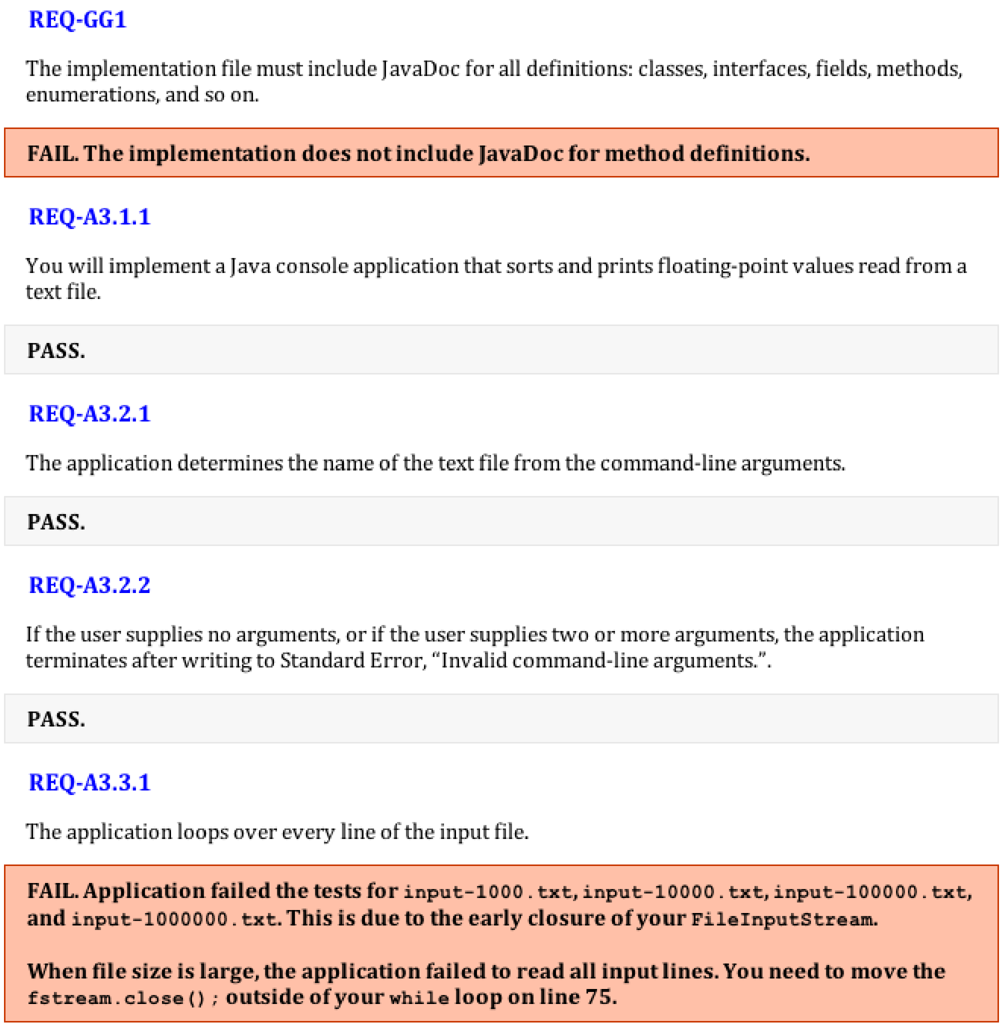


```
1.  Javadoc: Description expected after this reference
    LastFirst3.java; line 28

2.  Javadoc: Missing comment for default declaration
    LastFirst3.java; line 102

3.  Javadoc: Missing comment for default declaration
    3.java; line 108

4.  The import java.io.FileNotFoundException is never used
    LastFirst3.java; line 14

5.  The value of the local variable file is not used
    LastFirst3.java; line 40
```

Grading Tools for Programming Assignments

The Deliverables

Third, report the performance on each requirement and how to correct the mistakes.



REQ-GG1

The implementation file must include JavaDoc for all definitions: classes, interfaces, fields, methods, enumerations, and so on.

FAIL. The implementation does not include JavaDoc for method definitions.

REQ-A3.1.1

You will implement a Java console application that sorts and prints floating-point values read from a text file.

PASS.

REQ-A3.2.1

The application determines the name of the text file from the command-line arguments.

PASS.

REQ-A3.2.2

If the user supplies no arguments, or if the user supplies two or more arguments, the application terminates after writing to Standard Error, "Invalid command-line arguments.".

PASS.

REQ-A3.3.1

The application loops over every line of the input file.

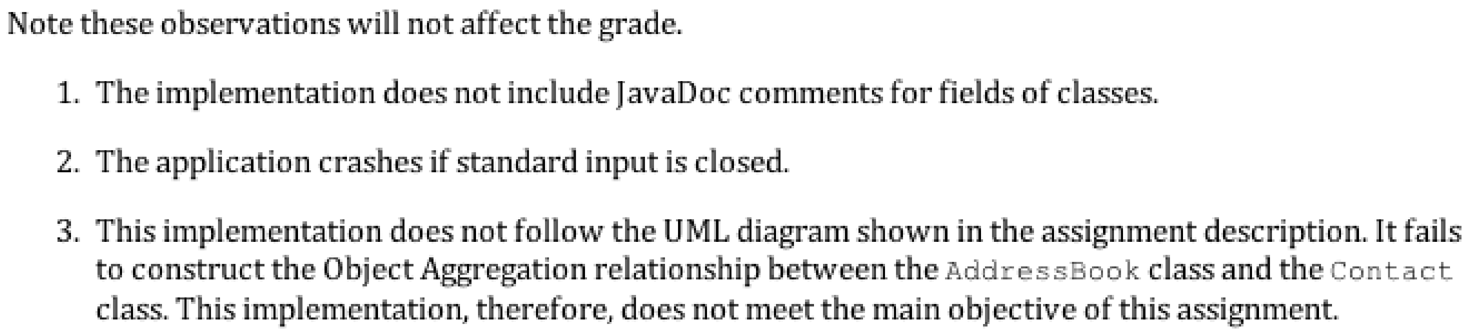
FAIL. Application failed the tests for `input-1000.txt`, `input-10000.txt`, `input-100000.txt`, and `input-1000000.txt`. This is due to the early closure of your `FileInputStream`.

When file size is large, the application failed to read all input lines. You need to move the `fstream.close();` outside of your `while` loop on line 75.

Grading Tools for Programming Assignments

The Deliverables

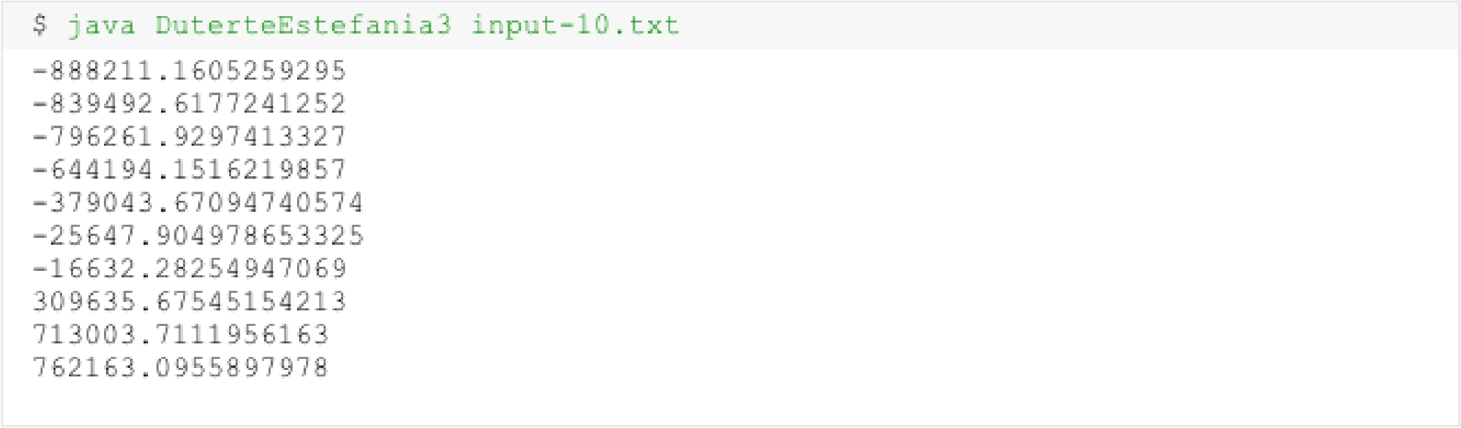
Fourth, report other observations.



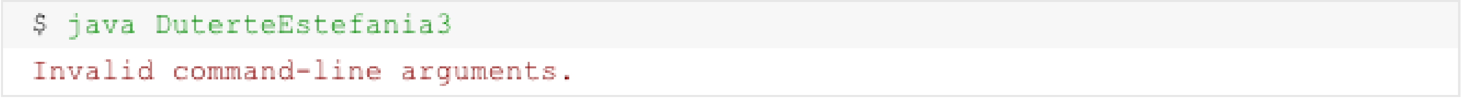
Note these observations will not affect the grade.

1. The implementation does not include JavaDoc comments for fields of classes.
2. The application crashes if standard input is closed.
3. This implementation does not follow the UML diagram shown in the assignment description. It fails to construct the Object Aggregation relationship between the `AddressBook` class and the `Contact` class. This implementation, therefore, does not meet the main objective of this assignment.

Finally, demonstrate program I/O for various cases.



```
$ java DuterteEstefania3 input-10.txt
-888211.1605259295
-839492.6177241252
-796261.9297413327
-644194.1516219857
-379043.67094740574
-25647.904978653325
-16632.28254947069
309635.67545154213
713003.7111956163
762163.0955897978
```



```
$ java DuterteEstefania3
Invalid command-line arguments.
```

Grading Tools for Programming Assignments

The Grading Process

Start by defining each requirement – e.g., Assignment3.xml

```
<assignment>

<requirement id="REQ-A3.3.1"><![CDATA[
<p>The application loops over every line of the input file.</p>
]]></requirement>

<requirement id="REQ-A3.3.2"><![CDATA[
<p>For every iteration, the application parses the line as a
<code>Double</code>.</p>
]]></requirement>

<requirement id="REQ-A3.3.3"><![CDATA[
<p>If a line from the file is empty, the application terminates after
writing to Standard Error, “Empty line encountered.”.</p>
]]></requirement>

:

</assignment>
```

http://www.jade-cheng.com/hpu/research-presentation/Assignment.xml

Grading Tools for Programming Assignments

The Grading Process

Record each student's performance in XML format – e.g., DoeJohn3.xml

```
<review first="First Name" last="Last Name">

<requirement id="REQ-SP4" result="pass"><![CDATA[
]]></requirement>

<requirement id="REQ-SP4" result="pass"><![CDATA[
]]></requirement>

<requirement id="REQ-SP5" result="pass"><![CDATA[
]]></requirement>

<requirement id="REQ-GG1" result="fail"><![CDATA[
The implementation does not include JavaDoc for method definitions.
]]></requirement>

<requirement id="REQ-A3.1.1" result="pass"><![CDATA[
]]></requirement>

:

</review>
```

http://www.jade-cheng.com/hpu/research-presentation/student-code-review.xml

Grading Tools for Programming Assignments

Curve Calculations

After recording all student performance, a summary text file is generated.

```
STUDENT                   PASS      FAIL     TOTAL   PERCENT
-------------------- --------- --------- --------- ---------
Adam Amick                  22         2        24       92%
Joshua Cayco                24         0        24      100%
Carlos Chavarria            22         2        24       92%
Kevin Drake                 20         4        24       83%

:

Jose Reque                   8        16        24       33%
Kayla Schlaich              17         7        24       71%
Lauren Smoot                20         4        24       83%
Kyle Tobara                 12        12        24       50%
-------------------- --------- --------- --------- ---------
AVERAGES                    17         7        24       72%


Warning: Curve is not optimal.

CURVE:   0% to 100%            CURVE:   36% to 100%
QUALITY: 85.00                 QUALITY: 93.25
-------------------            --------------------
A    7                         A    8
B    5                         B    5
C    1                         C    6
D    5                         D    2
F    6                         F    3
```

http://www.jade-cheng.com/hpu/research-presentation/Summary.txt

Grading Tools for Programming Assignments

Curve Calculations

With the curving recommendation, an optimal curving schema is selected.

```
STUDENT                   PASS      FAIL     TOTAL   PERCENT     CURVE
-------------------- --------- --------- --------- --------- ---------
Adam Amick                  22         2        24       92%       94%
Joshua Cayco                24         0        24      100%      100%
Carlos Chavarria            22         2        24       92%       94%
Kevin Drake                 20         4        24       83%       88%

:

Jose Reque                   8        16        24       33%       51%
Kayla Schlaich              17         7        24       71%       79%
Lauren Smoot                20         4        24       83%       88%
Kyle Tobara                 12        12        24       50%       63%
-------------------- --------- --------- --------- --------- ---------
AVERAGES                    17         7        24       72%       80%

CURVE:   36% to 100%
QUALITY: 93.25
--------------------
A    8
B    5
C    6
D    2
F    3
```

http://www.jade-cheng.com/hpu/research-presentation/Summary.txt

http://www.jade-cheng.com/hpu/research-presentation/student-code-review.html

CSE Technologies in Practice

Performance Visualization

Measuring and presenting the success of teaching and learning is a significant aspect in CSE.

Without a systematic and informative evaluation system, students and instructors lose track of their statuses gradually.

To address this, I have created a set of tools to evaluate and present performance.

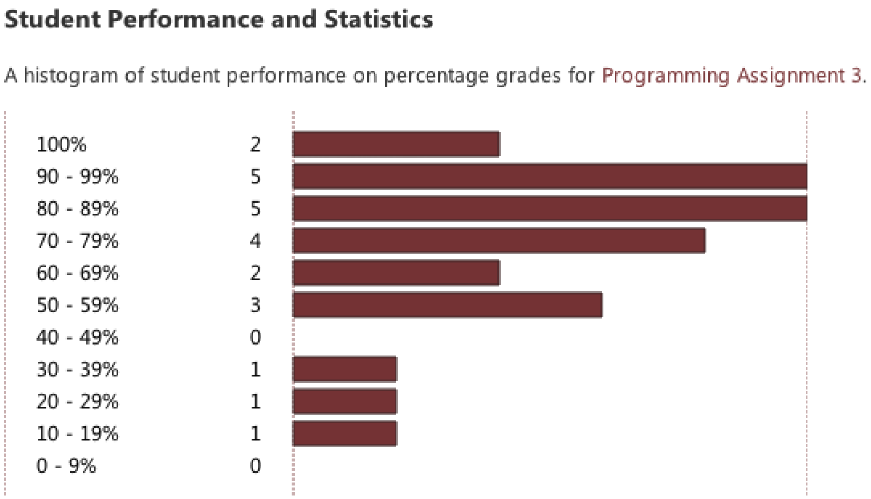
These presentations have not only served their intended purpose, they have also generated a more friendly classroom atmosphere.

Technologies:

- PHP
- SVG
- Google Charting API
- HTML/CSS
- Java

Performance Visualization

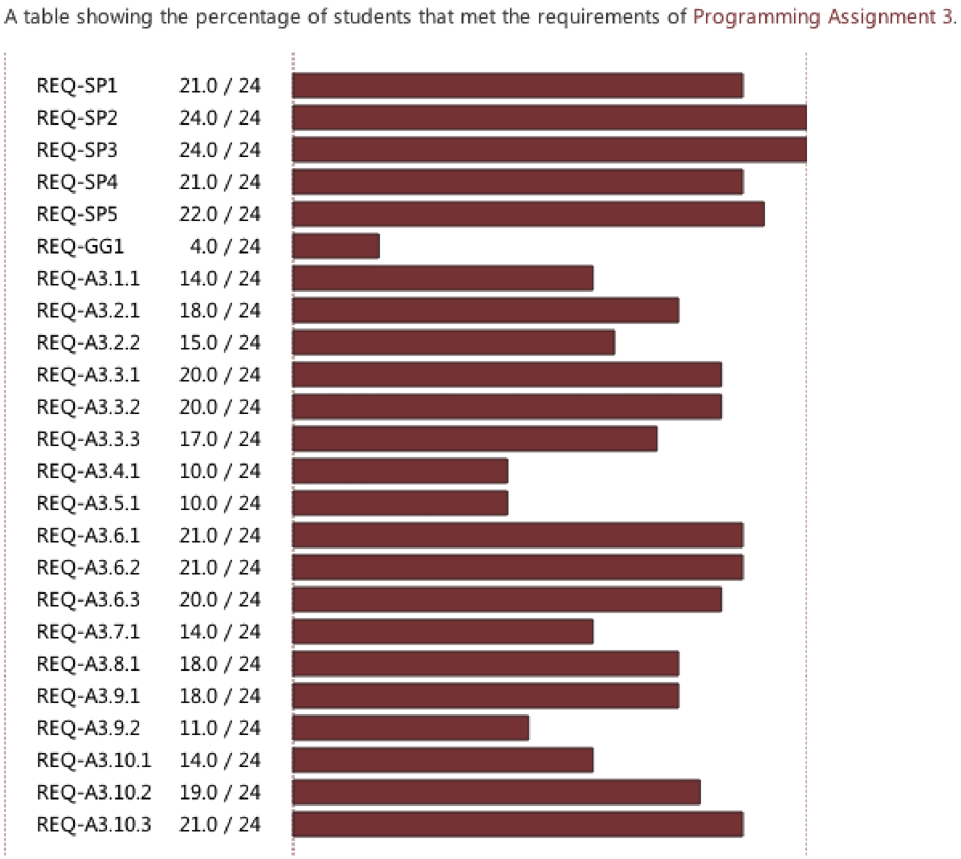
Student Performance Statistics for Programming Assignments



Student Performance and Statistics

A histogram of student performance on percentage grades for Programming Assignment 3.

100%	2	
90 - 99%	5	
80 - 89%	5	
70 - 79%	4	
60 - 69%	2	
50 - 59%	3	
40 - 49%	0	
30 - 39%	1	
20 - 29%	1	
10 - 19%	1	
0 - 9%	0	



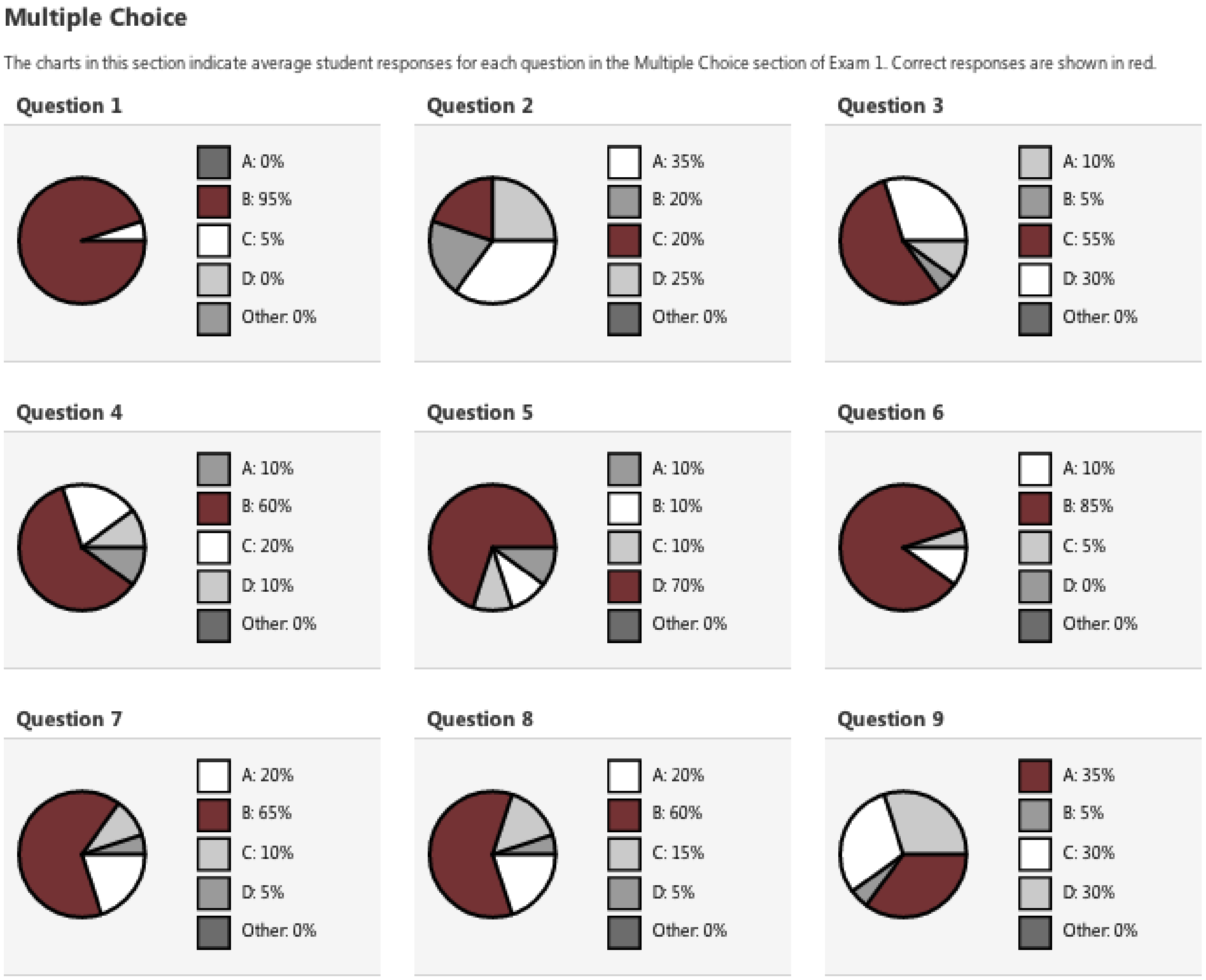
A table showing the percentage of students that met the requirements of Programming Assignment 3.

REQ-SP1	21.0 / 24	
REQ-SP2	24.0 / 24	
REQ-SP3	24.0 / 24	
REQ-SP4	21.0 / 24	
REQ-SP5	22.0 / 24	
REQ-GG1	4.0 / 24	
REQ-A3.1.1	14.0 / 24	
REQ-A3.2.1	18.0 / 24	
REQ-A3.2.2	15.0 / 24	
REQ-A3.3.1	20.0 / 24	
REQ-A3.3.2	20.0 / 24	
REQ-A3.3.3	17.0 / 24	
REQ-A3.4.1	10.0 / 24	
REQ-A3.5.1	10.0 / 24	
REQ-A3.6.1	21.0 / 24	
REQ-A3.6.2	21.0 / 24	
REQ-A3.6.3	20.0 / 24	
REQ-A3.7.1	14.0 / 24	
REQ-A3.8.1	18.0 / 24	
REQ-A3.9.1	18.0 / 24	
REQ-A3.9.2	11.0 / 24	
REQ-A3.10.1	14.0 / 24	
REQ-A3.10.2	19.0 / 24	
REQ-A3.10.3	21.0 / 24	

http://www.jade-cheng.com/hpu/2012-spring/csci-2912/assignments/

Performance Visualization

Student Performance Statistics for Exams



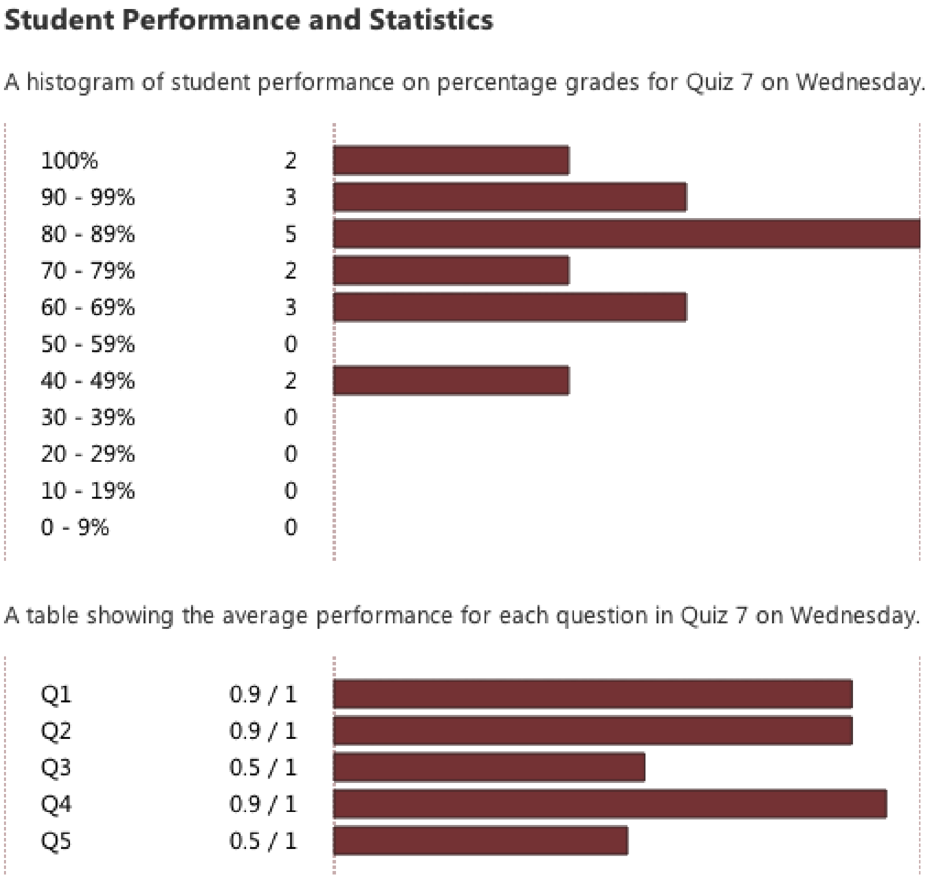
Multiple Choice

The charts in this section indicate average student responses for each question in the Multiple Choice section of Exam 1. Correct responses are shown in red.

http://www.jade-cheng.com/hpu/2012-spring/csci-2912/exams/

Performance Visualization

Student Performance Statistics for Quizzes



Student Performance and Statistics

A histogram of student performance on percentage grades for Quiz 7 on Wednesday.

100%	2
90 - 99%	3
80 - 89%	5
70 - 79%	2
60 - 69%	3
50 - 59%	0
40 - 49%	2
30 - 39%	0
20 - 29%	0
10 - 19%	0
0 - 9%	0

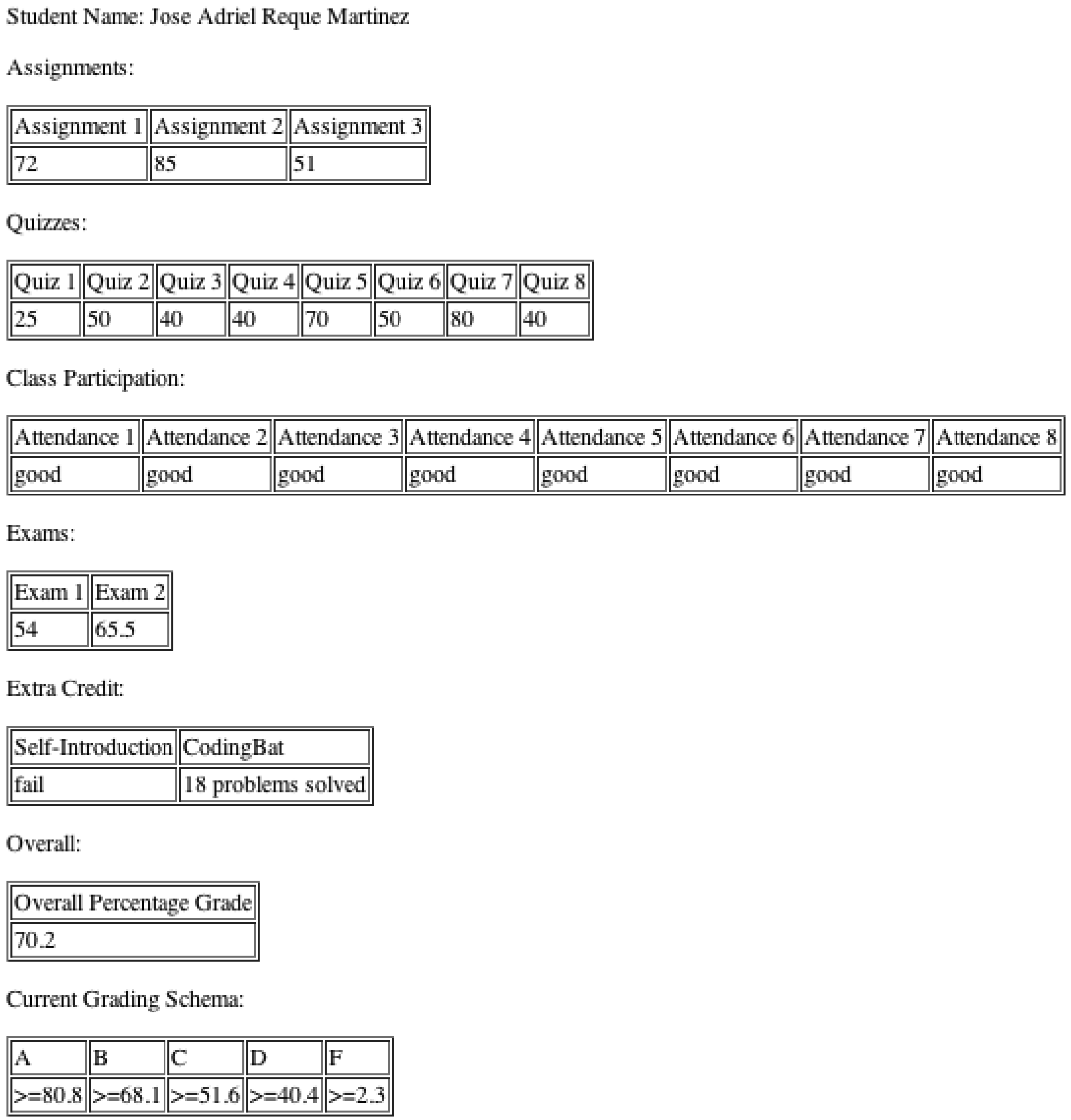
A table showing the average performance for each question in Quiz 7 on Wednesday.

Q1	0.9 / 1
Q2	0.9 / 1
Q3	0.5 / 1
Q4	0.9 / 1
Q5	0.5 / 1

http://www.jade-cheng.com/hpu/2012-spring/csci-2912/quizzes/

Performance Visualization

Semester Performance Estimation



Student Name: Jose Adriel Reque Martinez

Assignments:

Assignment 1	Assignment 2	Assignment 3
72	85	51

Quizzes:

Quiz 1	Quiz 2	Quiz 3	Quiz 4	Quiz 5	Quiz 6	Quiz 7	Quiz 8
25	50	40	40	70	50	80	40

Class Participation:

Attendance 1	Attendance 2	Attendance 3	Attendance 4	Attendance 5	Attendance 6	Attendance 7	Attendance 8
good	good	good	good	good	good	good	good

Exams:

Exam 1	Exam 2
54	65.5

Extra Credit:

Self-Introduction	CodingBat
fail	18 problems solved

Overall:

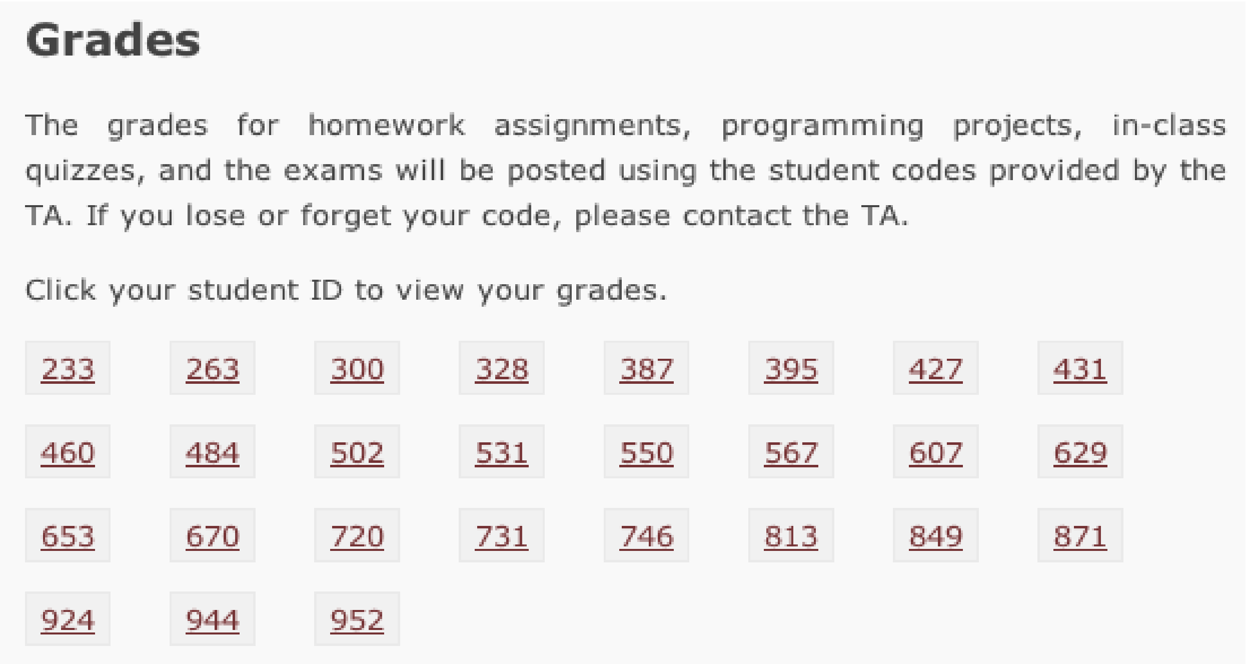
Overall Percentage Grade
70.2

Current Grading Schema:

A	B	C	D	F
>=80.8	>=68.1	>=51.6	>=40.4	>=2.3

Performance Visualization

Student grades made available online with ID codes.



Grades

The grades for homework assignments, programming projects, in-class quizzes, and the exams will be posted using the student codes provided by the TA. If you lose or forget your code, please contact the TA.

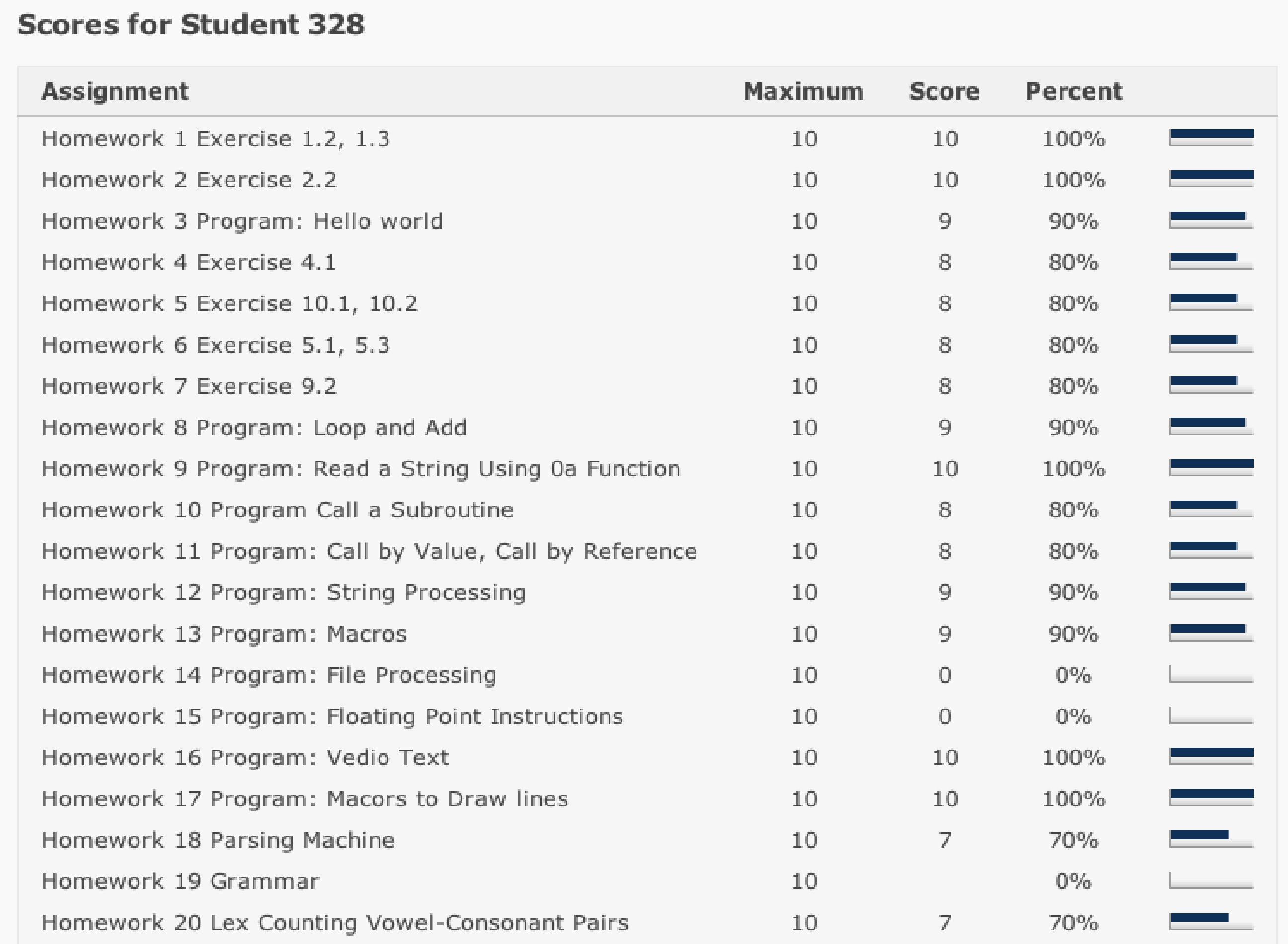
Click your student ID to view your grades.

233	263	300	328	387	395	427	431
460	484	502	531	550	567	607	629
653	670	720	731	746	813	849	871
924	944	952					

http://www2.hawaii.edu/~yucheng/ta/ics-312-fall-2009/#grades

Performance Visualization

Student grades made available online with ID codes.



Scores for Student 328

Assignment	Maximum	Score	Percent
Homework 1 Exercise 1.2, 1.3	10	10	100%
Homework 2 Exercise 2.2	10	10	100%
Homework 3 Program: Hello world	10	9	90%
Homework 4 Exercise 4.1	10	8	80%
Homework 5 Exercise 10.1, 10.2	10	8	80%
Homework 6 Exercise 5.1, 5.3	10	8	80%
Homework 7 Exercise 9.2	10	8	80%
Homework 8 Program: Loop and Add	10	9	90%
Homework 9 Program: Read a String Using 0a Function	10	10	100%
Homework 10 Program Call a Subroutine	10	8	80%
Homework 11 Program: Call by Value, Call by Reference	10	8	80%
Homework 12 Program: String Processing	10	9	90%
Homework 13 Program: Macros	10	9	90%
Homework 14 Program: File Processing	10	0	0%
Homework 15 Program: Floating Point Instructions	10	0	0%
Homework 16 Program: Vedio Text	10	10	100%
Homework 17 Program: Macors to Draw lines	10	10	100%
Homework 18 Parsing Machine	10	7	70%
Homework 19 Grammar	10		0%
Homework 20 Lex Counting Vowel-Consonant Pairs	10	7	70%

http://www2.hawaii.edu/~yucheng/ta/ics-312-fall-2009/?student=328#grades

CSE Research Interests

OOP Introductory Programming Environment

I will discuss an introductory programming environment developed by Stanford.

I will commenting on some of its shortcomings.

I will then propose a new system inspired by the work at Stanford to assist in teaching object oriented programming.

Programming with Robots

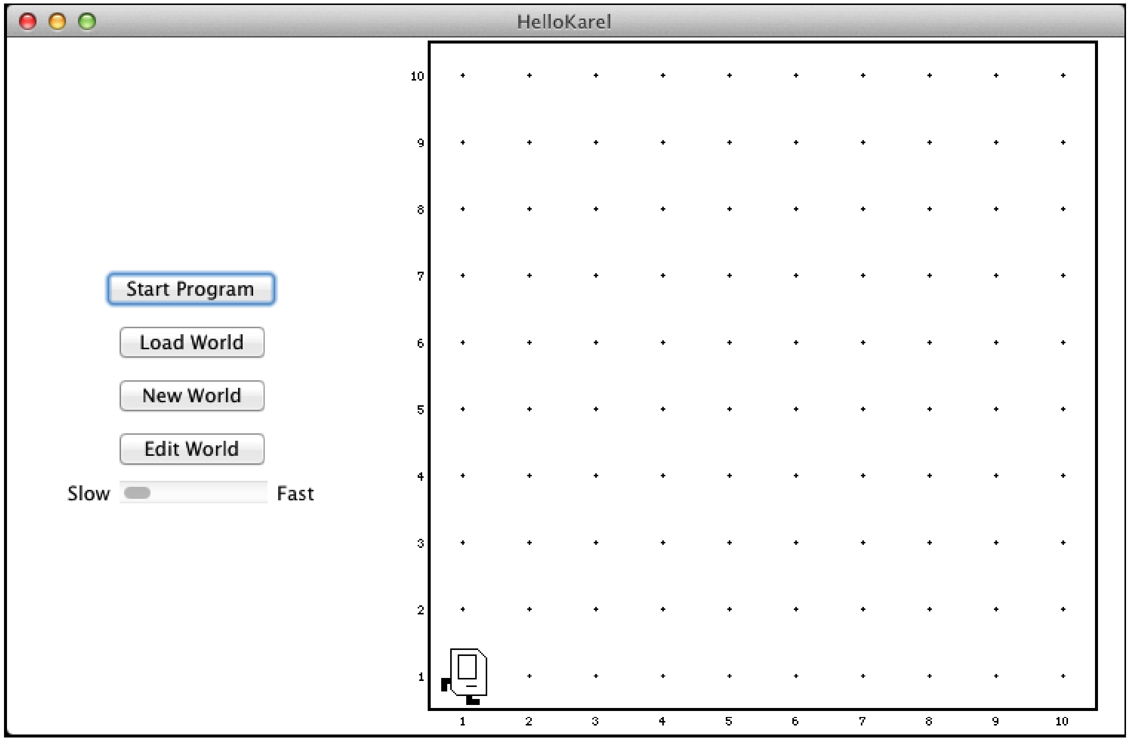
Finally, I will introduce a robotic development environment.

I will discuss its potential use for both entry-level and advanced programming courses.

OOP Introductory Programming Environment

Introduction

Karel the Robot developed in Stanford



How Karel is Used

Karel is used in introductory computer science courses all across the country.

Let's look at an example from Eclipse.

OOP Introductory Programming Environment

Shortcomings

Overwhelming complexity for a newcomers.

Inefficient environment for small-scale problems.

Difficult for students to relate the problems being solved

Improvements

Improve metaphor to help students quickly understand goals and obstacles.

Design online environment to avoid platform-specific installations.

Design Domain Specific Language to eliminate language distractions.

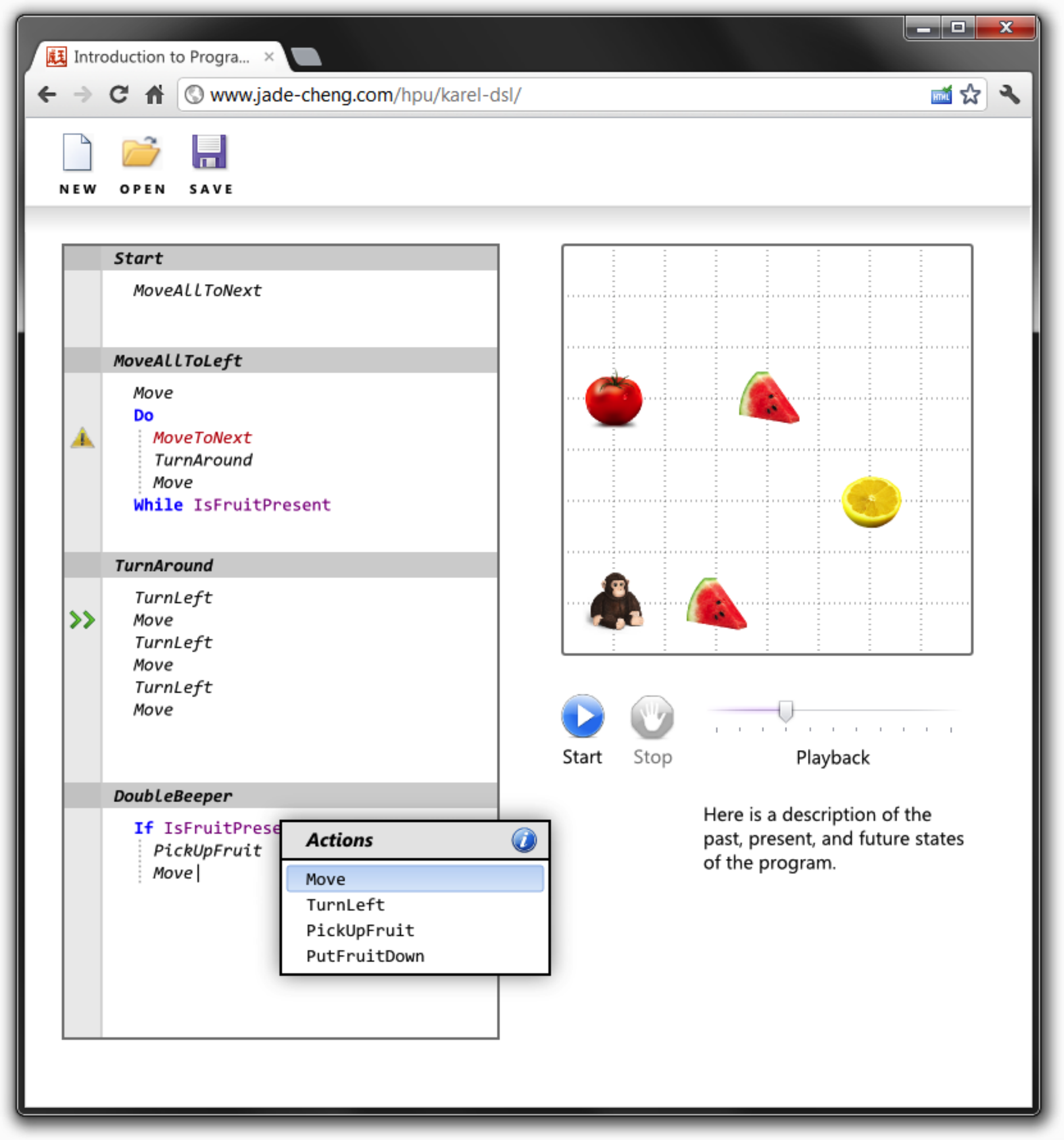
Provide immediate feedback in every possible way.

Implement ability to step forward and backward through the execution.

Do not try to be everything to everyone; solve a smaller problem.

OOP Introductory Programming Environment

Design of a OOP Introductory Programming Environment



```
Start
  MoveAllToNext

MoveAllToLeft
  Move
  Do
    MoveToNext
    TurnAround
    Move
  While IsFruitPresent

TurnAround
  TurnLeft
  Move
  TurnLeft
  Move
  TurnLeft
  Move

DoubleBeeper
  If IsFruitPrese
    PickUpFruit
    Move|
```

Here is a description of the past, present, and future states of the program.

Programming with Robots

Introduction

Lego Mindstorms is a line of programmable robotics and construction toys.

Lego Mindstorms contains many components including cables, sensors, and the NXT.

Lego NXT with Sensors and Motors

Programming with Robots

Lego NXT

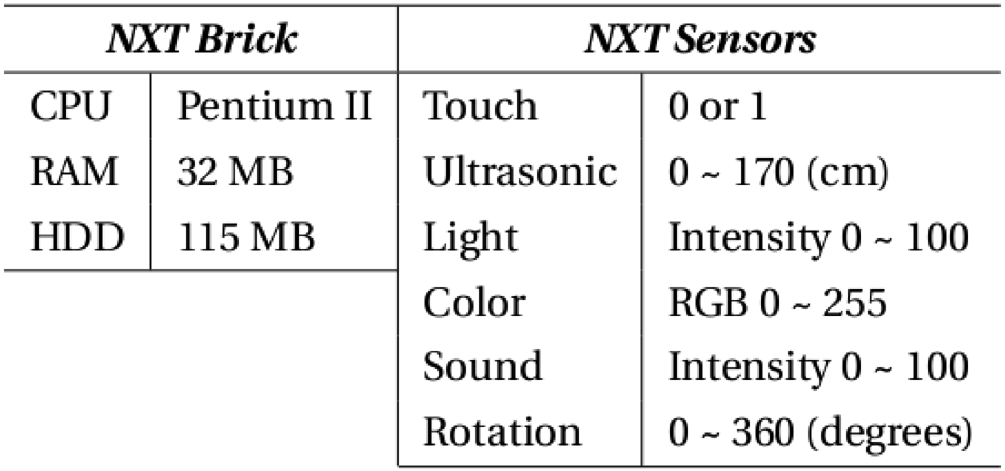
The NXT is the main component of Lego Mindstorms.

The NXT controls as many as four sensors and three motors via RJ12 cables.

The NXT has a 100×64-pixel monochrome LCD display and four buttons.

The NXT's stock firmware provides a user interface of hierarchical menus.

The NXT has a speaker and can play sound files at sampling rates up to 8 kHz.



NXT Brick		***NXT Sensors***	
CPU	Pentium II	Touch	0 or 1
RAM	32 MB	Ultrasonic	0 ~ 170 (cm)
HDD	115 MB	Light	Intensity 0 ~ 100
		Color	RGB 0 ~ 255
		Sound	Intensity 0 ~ 100
		Rotation	0 ~ 360 (degrees)

Lego NXT specs

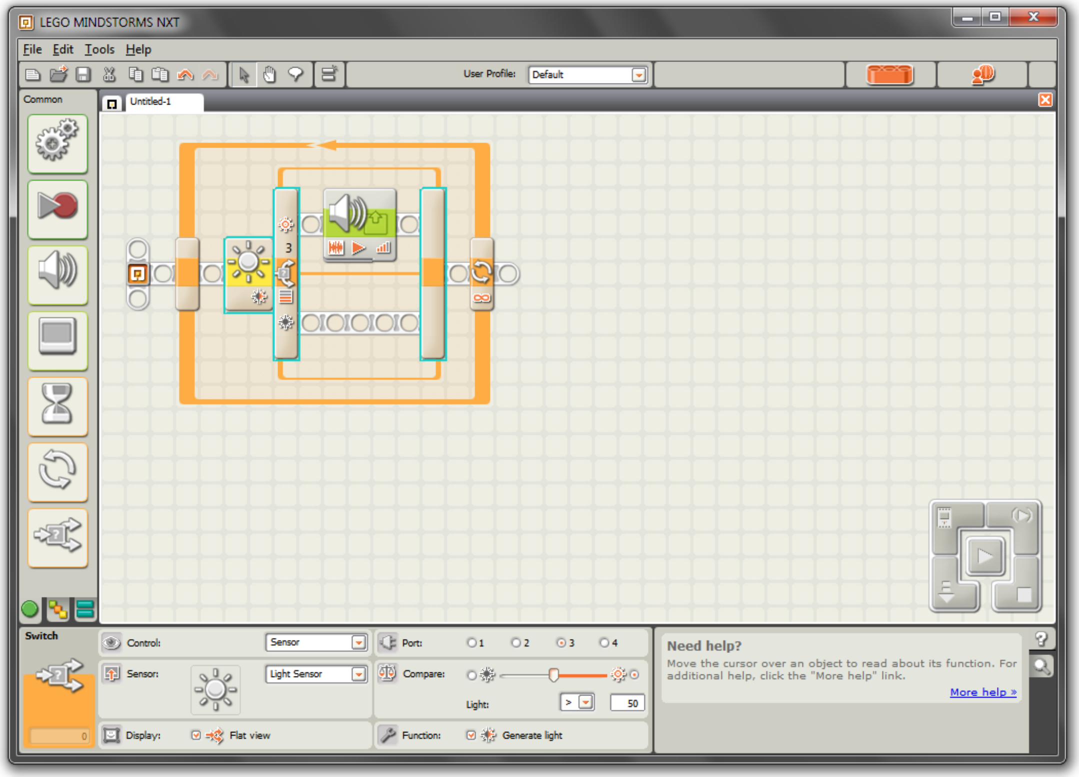
Programming with Robots

National Instruments

NXT-G is a programming package distributed with Lego Mindstorms.

LabVIEW for Lego NXT is a version of LabVIEW designed for work with NXT.

They both features an interactive drag-and-drop environment.



Switch

Control: Sensor

Sensor: Light Sensor

Display: ☑ Flat view

Port: ○ 1 ○ 2 ● 3 ○ 4

Compare:

Light: > 50

Function: ☑ Generate light

Need help?

Move the cursor over an object to read about its function. For additional help, click the "More help" link.

More help »

NXT-G Example Program

Programming with Robots

Java Programming Environment for NXT

leJOS is a high-level, open-source language based on Java.

leJOS applications execute on the NXT using a custom firmware.

leJOS's firmware provides a subset of the typical JRE.

Embedded Programming Environment

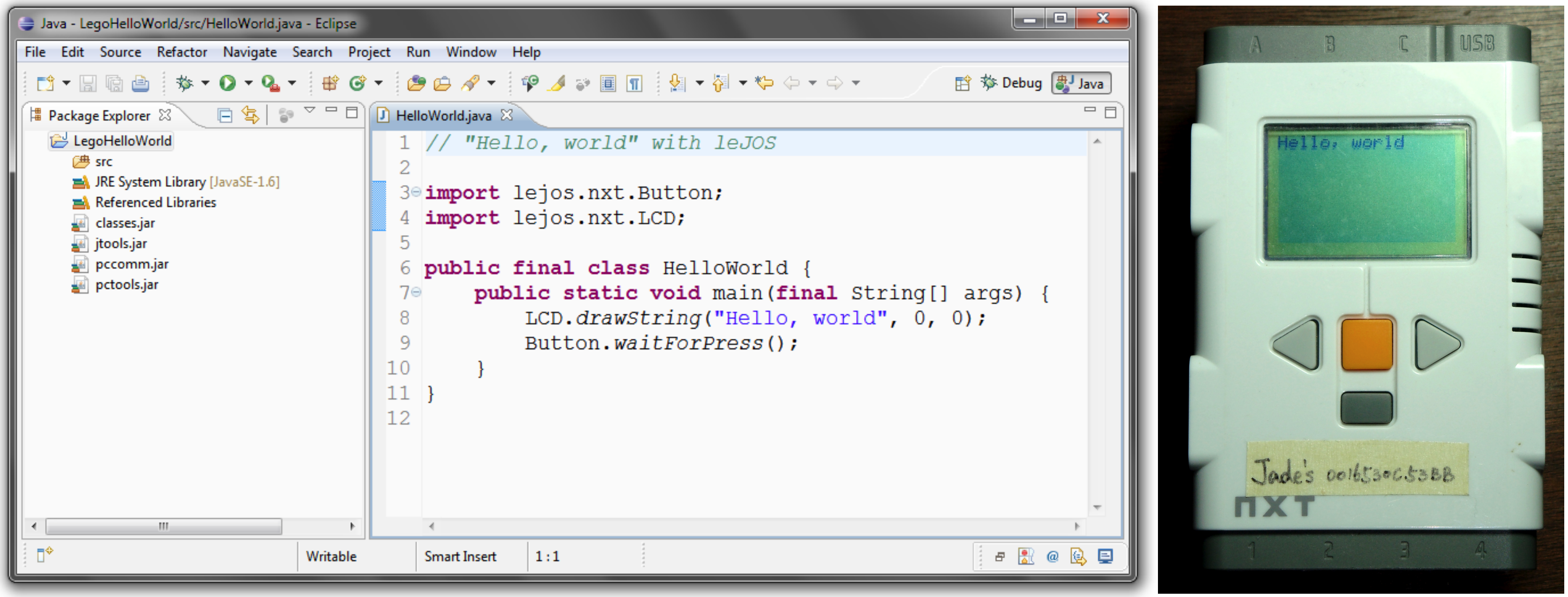
The Eclipse IDE can be used as the programming environment.

An external compiling tool is used to compile the code into a leJOS application.

An external transferring tool is used to upload the program to the NXT brick.

Programming with Robots

leJOS at work from Eclipse



```
// "Hello, world" with leJOS

import lejos.nxt.Button;
import lejos.nxt.LCD;

public final class HelloWorld {
    public static void main(final String[] args) {
        LCD.drawString("Hello, world", 0, 0);
        Button.waitForPress();
    }
}
```

leJOS Hello World Program

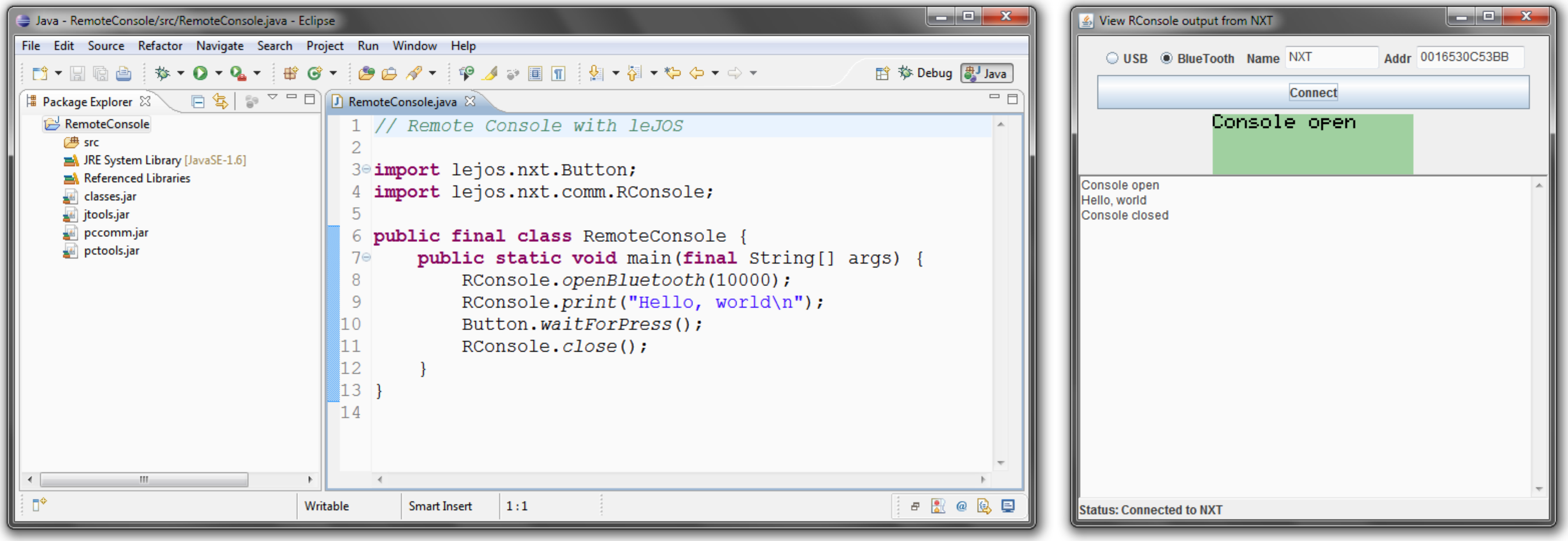
Programming with Robots

Runtime Diagnostics

The NXT LCD is inadequate for diagnosing and debugging high-speed operations.

leJOS provides a remote console module.

The remote console requires a connection, either USB or Bluetooth.


Java - RemoteConsole/src/RemoteConsole.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Java
Package Explorer
RemoteConsole
src
JRE System Library [JavaSE-1.6]
Referenced Libraries
classes.jar
jtools.jar
pccomm.jar
pctools.jar
RemoteConsole.java
Writable
Smart Insert
1 : 1
View RConsole output from NXT
Connect

```java
// Remote Console with leJOS

import lejos.nxt.Button;
import lejos.nxt.comm.RConsole;

public final class RemoteConsole {
    public static void main(final String[] args) {
        RConsole.openBluetooth(10000);
        RConsole.print("Hello, world\n");
        Button.waitForPress();
        RConsole.close();
    }
}

```

○ USB ● BlueTooth Name NXT Addr 0016530C53BB

```
Console open
```

```
Console open
Hello, world
Console closed
```

Status: Connected to NXT

leJOS Remote Console

Programming with Robots

Sample Programming Project with NXT

Goal:

Drive away from obstacles. In other words, look for center of a room.

Algorithm:

Continuously read from sensors, evaluate data, and decide new directions.

Repeat this procedure until center is found within margin.

Implementation:

Simple state machine with command-response PC-robot protocol.

See the robot in action from the URL below.

http://www.jade-cheng.com/hpu/research-presentation/nxt-sample-project.html