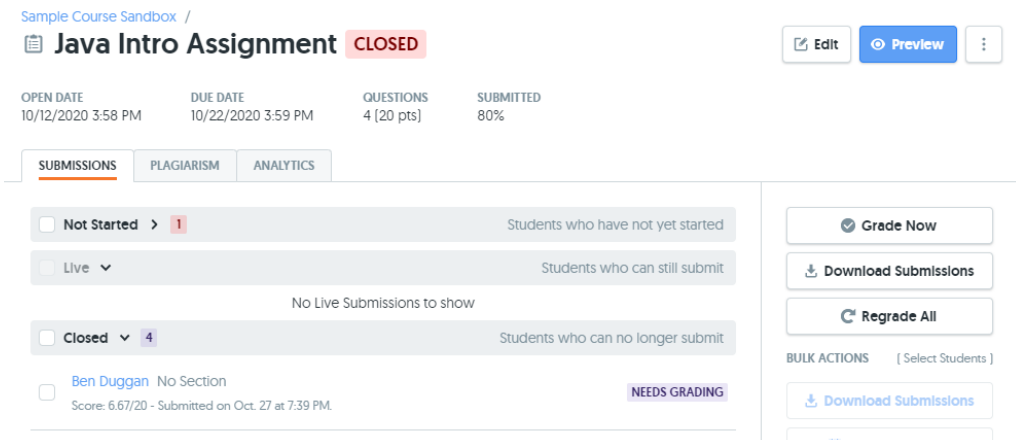
Marking Assignments

Mimir

- Navigate to the course Mimir section. Select the student checkbox. Then, select Grade Now.



Sample Course Sandbox /

Java Intro Assignment CLOSED

OPEN DATE	DUE DATE	QUESTIONS	SUBMITTED
10/12/2020 3:58 PM	10/22/2020 3:59 PM	4 (20 pts)	80%

SUBMISSIONS PLAGIARISM ANALYTICS

Not Started 1 — Students who have not yet started

Live — Students who can still submit

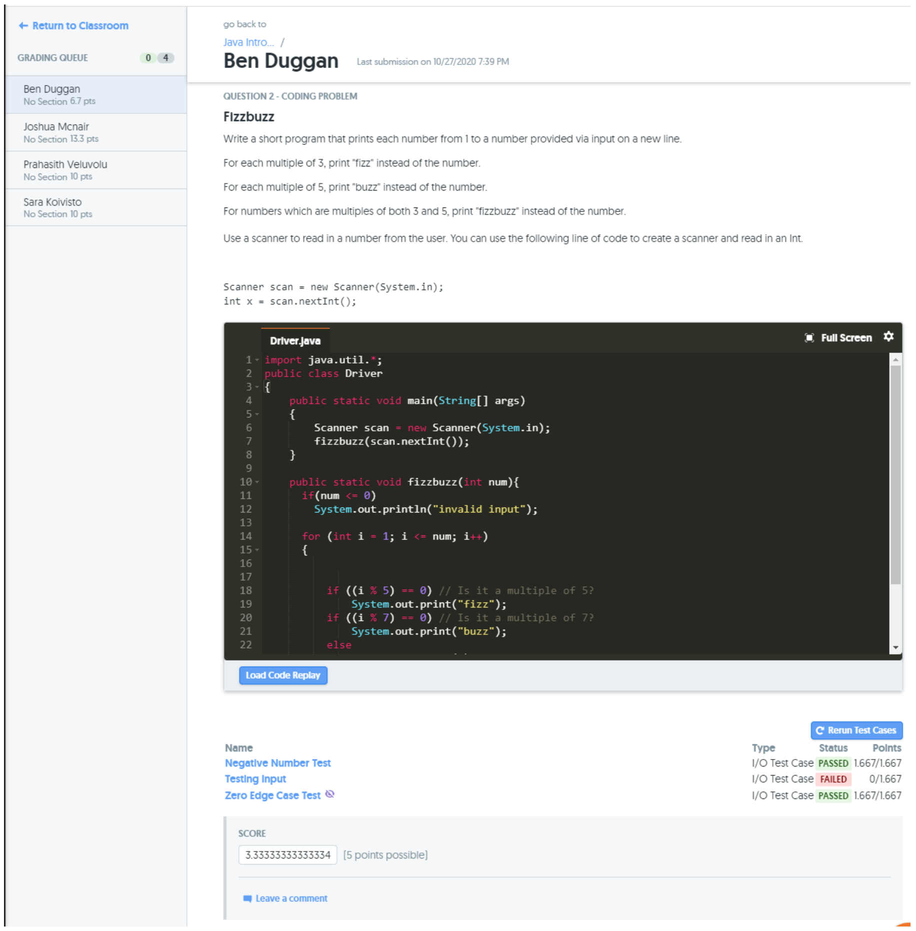
No Live Submissions to show

Closed 4 — Students who can no longer submit

Ben Duggan No Section — NEEDS GRADING

Score: 6.67/20 - Submitted on Oct. 27 at 7:39 PM.

- Mimir has a range of features to help you grade assignments, including an overview of the problem description, solution submitted by students, as well as test cases that were either passed or failed.



GRADING QUEUE 0 4

Ben Duggan
No Section 6.7 pts

Joshua Mcnair
No Section 13.3 pts

Prahasith Veluvolu
No Section 10 pts

Sara Koivisto
No Section 10 pts

go back to
Java Intro... /

Ben Duggan Last submission on 10/27/2020 7:39 PM

QUESTION 2 - CODING PROBLEM

Fizzbuzz

Write a short program that prints each number from 1 to a number provided via input on a new line.

For each multiple of 3, print "fizz" instead of the number.

For each multiple of 5, print "buzz" instead of the number.

For numbers which are multiples of both 3 and 5, print "fizzbuzz" instead of the number.

Use a scanner to read in a number from the user. You can use the following line of code to create a scanner and read in an Int.

```
Scanner scan = new Scanner(System.in);
int x = scan.nextInt();
```

Driver.java

```java
import java.util.*;
public class Driver
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);
        fizzbuzz(scan.nextInt());
    }

    public static void fizzbuzz(int num){
      if(num <= 0)
        System.out.println("invalid input");

      for (int i = 1; i <= num; i++)
      {


          if ((i % 5) == 0) // Is it a multiple of 5?
            System.out.print("fizz");
          if ((i % 7) == 0) // Is it a multiple of 7?
            System.out.print("buzz");
          else
```

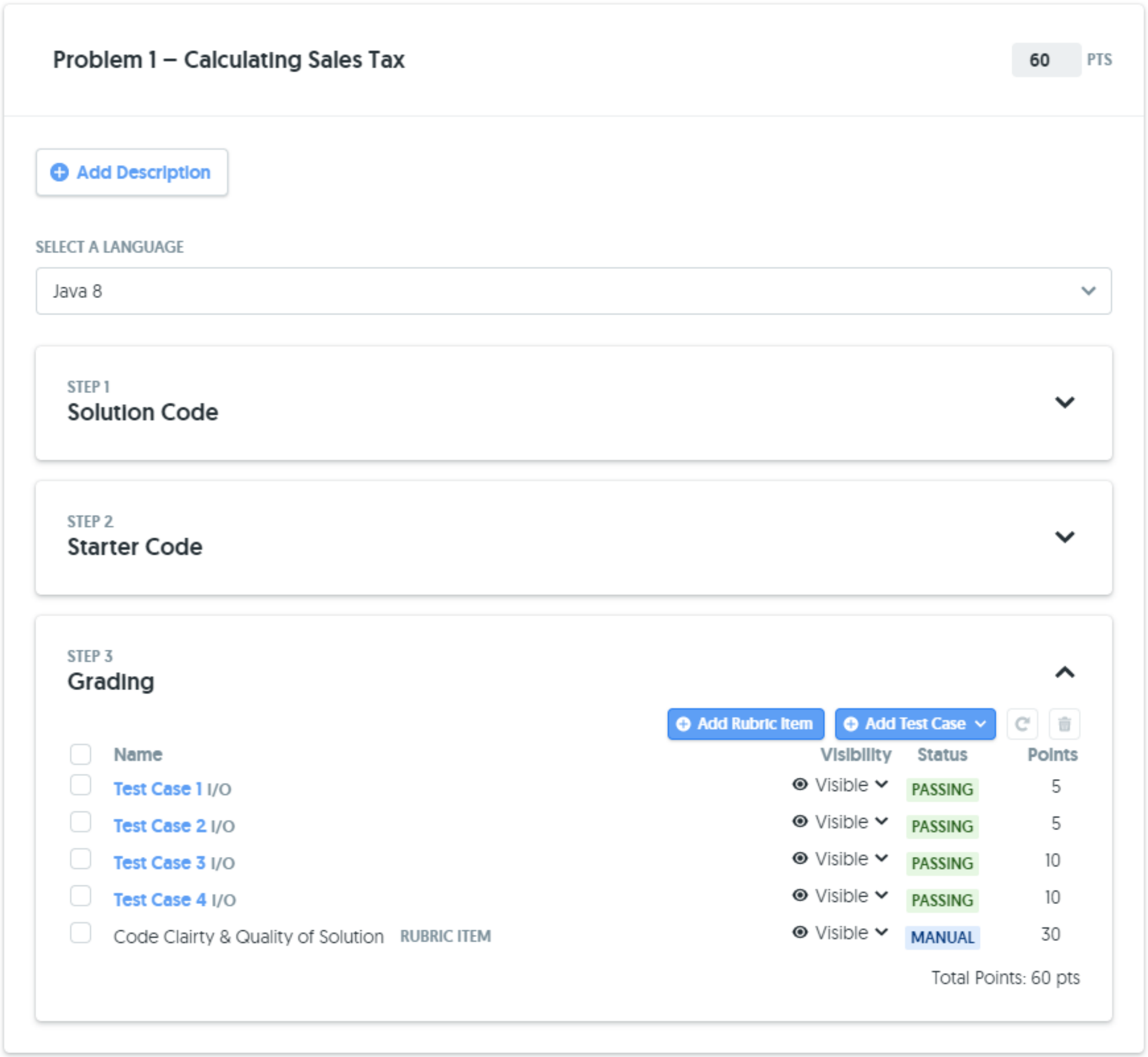
Name	Type	Status	Points
Negative Number Test	I/O Test Case	PASSED	1.667/1.667
Testing Input	I/O Test Case	FAILED	0/1.667
Zero Edge Case Test	I/O Test Case	PASSED	1.667/1.667

SCORE

3.33333333333334 (5 points possible)

Notice the Full Screen preview option. You can choose whether to Load Code Reply or Rerun Test Cases buttons.

The score can be entered in the textbox below. It will automatically be calculated for assignments that are evaluated through test cases; however, assignments are also manually graded using a scoring rubric. This is illustrated below in the code clarity and quality of solution category.



Problem 1 – Calculating Sales Tax 60 PTS

SELECT A LANGUAGE: Java 8

STEP 1 Solution Code

STEP 2 Starter Code

STEP 3 Grading

Name		Visibility	Status	Points
Test Case 1 I/O		Visible	PASSING	5
Test Case 2 I/O		Visible	PASSING	5
Test Case 3 I/O		Visible	PASSING	10
Test Case 4 I/O		Visible	PASSING	10
Code Clairty & Quality of Solution	RUBRIC ITEM	Visible	MANUAL	30

Total Points: 60 pts

- For manual coding of these rubric items, review the detailed description of the rubric available in the Brightspace course section in Home > Content > Assignments.

Grading Scheme

Each problem on the assignment will be graded based on three criteria:

Functionality

"Does it work according to specifications?" This is determined in an automated fashion by running your program on a number of inputs and ensuring that the outputs match the expected outputs. The score is determined based on the number of tests that your program passes.

Quality of Solution

"Is it a good solution?" This considers whether the solution is correct, efficient, covers boundary conditions, does not have any obvious bugs, etc. This is determined by visual inspection of the code. Initially full marks are given to each solution and marks are deducted based on faults found in the solution.

Code Clarity

"Is it well written?" This considers whether the solution is properly formatted, well-documented, and follows coding style guidelines. A single code clarity score is assigned for all solutions. If your program does not compile, it is considered non-functional and of extremely poor quality, meaning you will receive 0 for the solution.

PROBLEM		PARTIAL POINTS	POINTS
PROBLEM 1			40
	Functionality	30	
	Quality of Solution & Code Clarity	10	
PROBLEM 2			30
	Functionality	25	
	Quality of Solution & Code Clarity	5	
PROBLEM 3			30
	Functionality	25	
	Quality of Solution & Code Clarity	5	
TOTAL		**100**	**100**

- To provide feedback to students on the quality of the solution and clarity of the code, consider making a reference to the following website: https://ericpoitras.github.io/CSCI1105_StyleGuidelines/.
- This will be embedded along in Teams in addition to activity worksheets that include examples and counter example of each rubric category (available in Class Notebook):

- Code Style – Documentation
    - Code Style – Presentation
    - Code Style – Flow
    - Code Style – Idiom
    - Code Style – Expression
    - Code Style - Structure
- Providing proper feedback is critical for the grade to be meaningful and for students to improve over time. Please review the example comments highlighted in the gray boxes. You are encouraged to copy/paste with slight modifications as necessary to ensure consistency across markers.
- Share any common difficulties or misconceptions that students may have towards their assignments. We can revise the examples and counter examples to mention these more explicitly or address them during the lecture.

How did the student perform on this problem, based on a set of criteria? The focus is on comprehensibility, or how easy or difficult is it to understand a piece of code.

Choose **any** of the following options, as applicable. The category level mentioned in the instructions above indicates that problematic features are present, and grades should be deducted. Copy and paste the category that is problematic, as in the following:

Example Comment

Address the following issue(s) to improve the comprehensibility of your code:

Names appear unreadable, meaningless, or misleading (-2)

Header comment is missing (-1)

Documentation – Is the code well-annotated to ensure rapid understanding?

Category	Level			
	1	2	3	4
Names	Names appear unreadable, meaningless, or misleading	Names accurately describe the intent of the code, but can be incomplete, lengthy, misspelled or inconsistent use of casing	Names accurately describe the intent of the code, and are complete, distinctive, concise, correctly spelled and consistent use of casing	All names in the program use a consistent vocabulary

Headers	Headers are generally missing, or descriptions are redundant or obsolete; use mixed languages or are misspelled	Header comments are generally present; summarize the goal of parts of the program and how to use those; but may be somewhat inaccurate or incomplete	Header comments are generally present; accurately summarize the role of parts of the program and how to use those; but may still be wordy	header comments are generally present; contain only essential explanations, information and references
Comments	Comments are generally missing, redundant or obsolete; use mixed languages or are misspelled	Comments explain code and potential problems, but may be wordy	Comments explain code and potential problems, are concise	Comments are only present where strictly needed

Presentation – Is the code visually organized for a quick read?

Category	Level			
	1	2	3	4
Layout	Old commented out code is present, or lines are generally too long to read	Positioning of elements within source files is not optimized for readability	Positioning of elements within source files is optimized for readability	Positioning of elements is consistent between files and in line with Java conventions
Formatting	Formatting is missing or misleading	Indentation, line breaks, spacing and brackets highlight the intended structure but erratically	Indentation, line breaks, spacing and brackets consistently highlight the intended structure	Formatting makes similar parts of code clearly identifiable

Algorithms – Is each part of the code as simple as possible?

Category	Level			
	1	2	3	4

Flow	There is deep nesting; Code performs more than one task per line; Unreachable code is present	Flow is complex or contains too many exceptions or jumps; Parts of code are duplicate	Flow is simple and contains few exceptions or jumps; Duplication is very limited	In the case of exceptions or jumps, the most common path through the code is clearly visible
Idiom	Control structures are customized in a misleading way	Choice of control structures is inappropriate	Choice of control structures is appropriate; Reuse of library functionality may be limited	Reuse of library functionality and generic data structures where possible
Expressions	Expressions are repeated or contain unnamed constants	Expressions are complex or long; data types are inappropriate	Expressions are simple; data types are appropriate	Expressions are all essential for control flow

Structure – Is the code organized for quick understanding of parts and the whole?

Category	Level			
	1	2	3	4
Decomposition	Most code is in one or a few big methods; variables are reused for different purposes	Most methods are limited in length but mix tasks; Methods share many variables; Parts of code are repeated	Methods perform a limited set of tasks divided into parts; Shared variables are limited; Code is unique	Methods perform a very limited set of tasks and the number of parameters and shared variables is limited
Modularization	Classes are artificially separated	Classes have vague subjects, contain many variables or contain many methods	Classes have clearly defined subjects, contain few variables and a limited amount of methods	Classes are defined such that communication between them is limited

Is the student sufficiently competent at programming to implement the statement covered in this assignment (i.e., arithmetic/string operations, selection statements)?

Choose **one** of the following options:

- If the answer is NO, but with a specific **MINOR** difficulty. Insert a hint:

Action	Description	Line Number
Choose an action: Add Remove Move Change	Statement identifier, keep it brief and to the point (e.g., equals true, nested ifs, type of loop, add 0 to sum, duplicated sum + = …, and so on)	Line identifier (e.g., line 5)

Example

Add an arithmetic operation to assign value of sum (line 10)

- **MAJOR** difficulties - You may also recommend that they drop into TA office hours to ask for help if they do not understand what the question was asking or with help on how to solve them. Copy and paste the following:

Example (Copy/Paste)

You may want to discuss your assignment with a TA and review the correct code solution. Remember to drop into office hours to ask help from TAs to understand what the question in the problem is asking you to do as well as how to solve it. Pay careful attention to the test cases to understand what output should be produced given the test case input values, and to keep doing your best to solve these problems!

References

Fitzgerald, S., Hanks, B., Lister, R., McCauley, R., Murphy, L. (2013). What are we thinking when we grade programs? In Proceeding of the 44th ACM technical symposium on Computer Science Education (pp. 471-476).

Keuning, H., Heeren, B., & Jeuring, J. (2019, July). How teachers would help students to improve their code. In Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education (pp. 119-125).

Stegeman, M., Barendsen, E., & Smetsers, S. (2014, November). Towards an empirically validated model for assessment of code quality. In Proceedings of the 14th Koli Calling international conference on computing education research (pp. 99-108).

Stegeman, M., Barendsen, E., & Smetsers, S. (2016, November). Designing a rubric for feedback on code quality in programming courses. In Proceedings of the 16th Koli Calling International Conference on Computing Education Research (pp. 160-164).