CS 501: Introduction to Java Programming
Fall 2022 Final Exam

This final exam contains four problems. Students are allowed to use the textbook and given codes provided by the instructor and TAs but nothing else. Students must not work together; any tendency to cheat or copy will mark the semester grade as 0.

The exam must be submitted by Friday 12/16th, at 11:59 PM. Late submission is not accepted and will not be negotiated for any delays.

All files, including the test paper with your signature, must be compressed and submitted in a single file as following structure, **LastName FirstName FinalExam.zip**.

Statement of Academic Honesty:

1. I will not use any non-instructor approved electronic devices and sources to assist me on an exam.
2. I have not received, I have not given, nor will I give or receive, any assistance to another student taking this exam, including discussing the exam with students in this course.
3. I understand that acts of academic dishonesty may be penalized to the full extent allowed by the University Code, including receiving a failing grade for the course. I recognize that I am responsible for understanding the provisions of University Conduct Code as they relate to this academic exercise.

Name:
Zimeng Zhao

Date:

12.16

Initial:

ZmZ

1 Arrays [20 pts]

Create a class called "**ArraysCheck**". You will be given two integer arrays A and B of equal length. You are allowed to select any n number of elements from A and add 1 to those elements. If all the elements of A now match elements of B or if all elements of A are a permutation (some jumbled order) of B you will return True. If not, you will return False. You are allowed to using arrays library. This algorithm must be implemented in a public static **boolean** check(int[] a, int[] b) method.

Example:

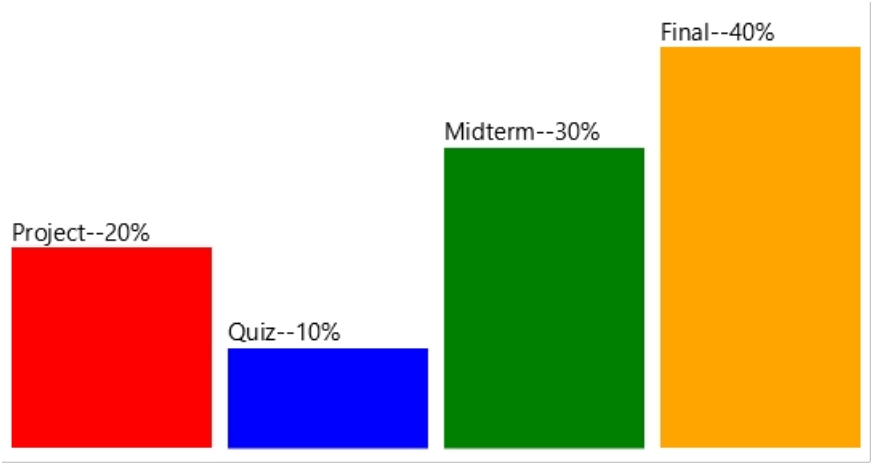
$$\text{if} A = [4, 3, 2, 4, 0] \text{ and } B = [1, 2, 3, 4, 5], \quad \text{Output} = \text{True} \tag{1}$$

Explanation: If you select $A[0] = 4$ and $A[5] = 0$ and increase it by 1, you get $A = [5, 3, 2, 4, 1]$ which is a permutation (jumbled order) of $B = [1, 2, 3, 4, 5]$. You can also select $A[3] = 4$ and $A[5] = 0$ and get the same result.

Note: This problem may seem a little intimidating, but it has a very simple solution to it.

2 Bar Chart [20 pts]

Write a program that uses a bar chart to display the percentages of the overall grade represented by projects, quizzes, midterm exams, and the final exam, as shown in Figure. The program should prompt the user to enter the percentages of Project, Quiz, Midterm and Final. If the percentages don't add up to 100, the program should throw an error message saying "Total percentage not equal to 100". If the total percentage is equal to 100, then display the bar chart.



Suppose projects take 20% and are displayed in red, quizzes take 10% and are displayed in blue, midterm exams take 30% and are displayed in green, and the final exam takes 40% and is displayed in orange. Use the Rectangle class to display the bars. You are free to name the class as you wish. Please also submit screenshot of the output along with the code in the background.

3 Linked List [30 pts]

Implement a singly linked list. Create two classes – Node and LL. Node class will act as the inner class of the LL class. You will be using generic data types.

1. (5 pts) Node⟨E⟩ will contain two attributes:

   1-1. E val

   1-2. Node⟨E⟩ next

   You need to create two constructors for Node⟨E⟩:

   1-a. public Node(E val)

   1-b. public Node(E val, Node⟨E⟩ nextNode)

2. (15 pts) LL⟨E⟩ will contain the following attributes:

   2-1. Node⟨E⟩ head

   2-2. ⟨E⟩ tail

   2-3. int len

   You need to create a constructors that create an empty linked list.

   LL⟨E⟩ will have the following methods:

   2-a. **public void append(E val)** which will create a new node and add it to the end of the linked list

   2-b. **public void prepend(E val)** which will create a new node and add it to the beginning of the linked list

   2-c. **public void deleteEnd()** which will delete the last node in the Linked List

   2-d. **public void deleteStart()** which will delete the first node in the Linked List

   2-e. **public void delete(E val)** which will delete the first node which contains the E data of value equal to val

   2-f. **public void deleteAll(E val)** which will delete all the nodes which contains the E data of value equal to val

   2-g. **public int getLen()** which will return the length of the linked list

   2-h. **public void sort()** which will sort the linked list

   2-i. **public void reverse()** which will reverse the linked list

3. (10 pts) Then perform the following:

   3-a. (5 pts) Using a provided array A = [" s00" , " 4s1" , " 41s" , " s31" , " 12s" , " s14" , " s23" , " s42"], create a linked list B by adding an element in the order of appearance and then sort. For each insertion, print out the index of A, B, and the length of B.

   3-b. (5 pts) Reverse B and print. Then remove from the end of the list. For each insertion, print out the index of B, B, and the length of B.

4 Matrix Multiplication [30 pts]

Create a class called "Matrix Multiplication". Consider two n×n matrices A and B where n is a power of 2. We can calculate the product of A and B recursively as following.

1. Partition A and B $n/2 \times n/2$ matrices.
2. Compute $C_{ij} = A_{ij} \times B_{ij}$

$$\begin{matrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{matrix} = \begin{matrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{matrix} \times \begin{matrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{matrix}. \tag{2}$$

a. (5 pts) Consider a matrix

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ -8 & -7 & -6 & -5 \\ -4 & -3 & -2 & -1 \end{bmatrix}. \tag{3}$$

Implement a function that partitions into four 2 × 2 matrices

$$A_{11} = \begin{matrix} 1 & 2 \\ 5 & 6 \end{matrix}, A_{12} = \begin{matrix} 3 & 4 \\ 7 & 8 \end{matrix}, A_{21} = \begin{matrix} -8 & -7 \\ -4 & -3 \end{matrix}, A_{22} = \begin{matrix} -6 & -5 \\ -2 & -1 \end{matrix} \tag{4}$$

Find the determinant of each partitioned matrix $\det A = a_{11}a_{22} - a_{12}a_{21}$, and find the sum of them.

b. (10 pts) Implement an algorithm that computes the product of two matrices, **recursiveMatrixMultiplication(int[] A, int[] B)**.

Algorithm 1 recursiveMatrixMultiplication(A,B)

$n = A.length()$
Let C be a new $n \times n$ matrix.
if $n = 1$ **then**
  $c_{11} = a_{11} \cdot b_{11}$
else {partition A, B, and C}
  C_{11} = recursiveMatrixMultiplication(A_{11},B_{11}) + recursiveMatrixMultiplication(A_{12},B_{21})
  C_{12} = recursiveMatrixMultiplication(A_{11},B_{12}) + recursiveMatrixMultiplication(A_{12},B_{22})
  C_{21} = recursiveMatrixMultiplication(A_{21},B_{11}) + recursiveMatrixMultiplication(A_{22},B_{21})
  C_{22} = recursiveMatrixMultiplication(A_{21},B_{12}) + recursiveMatrixMultiplication(A_{22},B_{22})
end if
return C

c. (5 pts) Compute the product of A and B

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ -8 & -7 & -6 & -5 \\ -4 & -3 & -2 & -1 \end{bmatrix}, B = \begin{bmatrix} 10 & 11 & 14 & 15 \\ 12 & 13 & 16 & 17 \\ -18 & -17 & -16 & -15 \\ -14 & -13 & -12 & -11 \end{bmatrix} \tag{5}$$

and return the result C.

d. (10 pts) Implement a matrix comparison algorithm, **equalMatrix(*A*,*B*)**, that contains the following methods

   d-1. (5 pts) **iterativeMatrixMultiplication(*A*,*B*)** computes the matrix multiplication iterative fashion.

   d-2. (5 pts) **matrixEqual(*A*,*B*)** returns boolean **True** if result from [c.] and matrices are the same and **False** otherwise. Print the result.