CS112 Data Structures - Midterm 1 - Spring 2022

Name: ____ NetID: ____________________________

- **WRITE your name and NetID on EVERY page.**
- **DO NOT REMOVE** THE STAPLE IN YOUR EXAM.
- **DO NOT BEGIN** UNTIL INSTRUCTED TO DO SO.
- WRITE NEATLY AND CLEARLY. If we cannot read your handwriting, you will not receive credit. Please plan your space usage. No additional paper will be given.
- This exam is worth 150 points.

Problem 1 – Arrays (30 points)

Given the code segment below.

```
for ( int i = 0; i < n - 1; i++ ) {

    int min = i;

    for ( int j = i + 1; j < n; j++ ) {

        if ( a[j] < a[min] ) {   // 2 array accesses
            min = j;
        }
    }

    int temp = a[i]; // 1 array access
    a[i] = a[min];   // 2 array access
    a[min] = temp;   // 1 array access
}
```

(a) **(10 points)** How many times is the `if` statement in the inner `for` loop executed? Give your answer as a function of n with a succinct explanation.

counter i	counter j	# of times if statement is performed
0	1 to n - 1	n - 1
1	2 to n - 1	n - 2
	…	…
	…	…
n - 2	n -1 to n - 1	1
sum		(n - 1) * n / 2

// 5 points for explanation: a table like the above OR an explanation on the number of times the inner loop is executed for each outer loop execution.

// 5 points: considered correct number of times the if statement is executed:

Name: ______________________________ NetID: ______________________

- (n-1) * n/2 OR
- n^2 – n / 2 OR
- (n -1) * (n-1+1)/2 OR
- (n-1) + (n-2) + … + 1

Name: ____ NetID: ____________________________

(b) **(10 points)** What is the maximum number of array accesses (reads and writes) for the entire code segment as a function of *n*? Justify your answer.

(2 * (n - 1) * n / 2) + // if statement in the inner for loop has 2 array accesses
4 * (n - 1) + // 4 array accesses in the outer for loop, it executes n-1 times
➔ $n^2 + 3n - 4$

// the solution does not have to be simplified.

(c) **(5 points)** Write the tilde notation for the function in (b).

$\sim n^2$

(d) **(5 points)** Write the Big-O notation for the function in (b).

$O(n^2)$

Name: ______________________________ NetID: ______________________

Problem 2 – Union-Find (30 points)

A client program is using the union-find API to solve a dynamic connectivity problem in networking. Assume that there are 10 sites identified as: 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 in the network. Answer the following questions.

(a) **(10 points)** If the union-find API is implemented with the quick-find discussed in class, show the content of the `id[]` array after adding the following edges in sequence: 0-7, 1-5, 2-8, 7-1, 8-3

0	1	2	3	4	5	6	7	8	9
5	5	3	3	4	5	6	5	3	9

- Indices 0,1,5,7 have the SAME value (either 0, 1, 5, or 7) // 3 points, no partial credit
- Indices 2,3,8 have the SAME value (either 2, 3, or 8) // 3 points, no partial credit
- Indices 4,6,9 have the value 4, 6, 9 respectively // 4 points, no partial credit

(b) **(10 points)** If the API is implemented with quick-union discussed in class, show the content of the `parent[]` array after adding the following edges in sequence: 0-7, 1-5, 2-8, 7-1, 5-2

0	1	2	3	4	5	6	7	8	9
7	5	8	3	4	8	6	5	8	9

- Indices 1,7 have the same value 5 // 2 points, no partial credit
- Index 0 have the value 7 // 2 point
- Indices 2,5,8 have the value 8 // 3 points , no partial credit
- Indices 3,4,6,9 separately have the value 3,4,6,9 // 3 points, no partial credit

Name: ____ NetID: ____________________________

(c) **(10 points)** If the API is implemented with weighted-quick-union discussed (union by size) in class, show the content of the `parent[]` and `size[]` array after adding the same edges from part (b).

parent[]

0	1	2	3	4	5	6	7	8	9
0	0	0	3	4	1	6	0	2	9

- Indices 0,1,2,7 have the value 0 // 3 points , no partial credit
- Index 5 has the value 1, Index 8 has the value 2 // 1 point, no partial credit
- Indices 3,4,6,9 separately have the value 3, 4, 6, 9 // 1 point, no partial credit

size[]

0	1	2	3	4	5	6	7	8	9
6	2	2	1	1	1	1	1	1	1

- Index 0 has 6 // 3 points
- Indices 1 and 2 have 2 // 1 point
- All others have 1. // 1 point

Name: ______________________________ NetID: ______________________

Problem 3 – Stacks and Queues (30 points)

(a) **(8 points)** What is the output of the following code segment?

```
Stack<Integer> stack = new Stack<Integer>();
int item1 = 1;
int item2 = 0;
int item3 = 4;

stack.push( item2 );
stack.push( item1 );
stack.push( item1 + item3 );

item2 = stack.pop();

stack.pop();
stack.push( item3 * item3 );
stack.push( item2 );
stack.push( 3 );

item1 = stack.pop();

stack.pop();
System.out.println( item1 + " " + item2 + " " + item3 );

while ( !stack.empty() ) {
    System.out.println( stack.pop() );
}
```

Answer:

```
3 5 4
16
0
```

Name: ____ NetID: ____________________________

(b) **(8 points)** What is the output of the following code segment?

```
 Queue<Character> queue = new Queue<Character>();
Stack<Character> stack = new Stack<Character>();

String s = "stack and queue";

char [] charArray = s.toCharArray();

for ( int i = 0; i < charArray.length; i++ ) {
    stack.push( charArray[i] );
}

while ( !stack.isEmpty() ) {
    queue.enqueue(stack.pop());
}

while ( !queue.isEmpty() ) {
    System.out.print( queue.dequeue() );
}
```

Answer:

eueuq dna kcats

Name: ______________________________________ NetID: __________________________

(c) Answer questions based in the code segment below. Note that `randomBoolean()` randomly returns true or false.

```
Stack<Integer> stack = new Stack<Integer>();

for ( int count = 0; count <= 5; count++ ) {

    if ( randomBoolean() ) {
        System.out.print( count );
    } else {
        stack.push( count );
    }
}

while ( !stack.isEmpty() ) {
    System.out.print( stack.pop() );
}
```

An announcement was made during the exam that count starts at 1.

Consider correct answers for count starting either at 0 or 1.

1. **(7 points)** Is the output 13524 possible? Explain.

// count starts at 1
No. The for-loop push (add) the count values in ascending order, and the while loop pop (remove) the values in reverse order; the output 135 is possible, however, the output 24 is not possible.

// count starts at 0
No. Zero is not present in the output.

2. **(7 points)** Is the output 13542 possible? Explain.

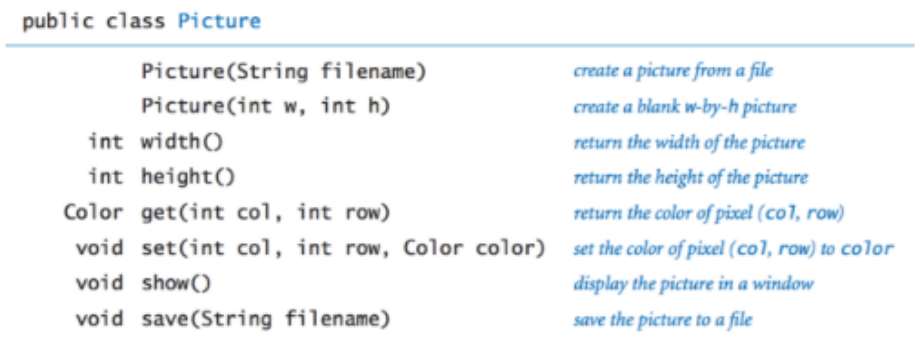
//count starts at 1
Yes; the output is generated when the odd values are printed directly and the even values are pushed to the stack. The even values will be displayed in a reverse order.

// count starts at 0
No. Zero is not present in the output.

Name: ______________________________________ NetID: ___________________________

Problem 4 – Linked Lists and Arrays (30 points)

On ArtCollage you worked with the Picture class which follow the digital image abstraction. The Picture is a 2D array of Color values, see operations below.



`public class Picture`

	`Picture(String filename)`	*create a picture from a file*
	`Picture(int w, int h)`	*create a blank w-by-h picture*
`int`	`width()`	*return the width of the picture*
`int`	`height()`	*return the height of the picture*
`Color`	`get(int col, int row)`	*return the color of pixel (col, row)*
`void`	`set(int col, int row, Color color)`	*set the color of pixel (col, row) to color*
`void`	`show()`	*display the picture in a window*
`void`	`save(String filename)`	*save the picture to a file*

The ImageProcessing class below also uses the Picture class, it contains two instance variables:

- ***image***: a reference to a Picture.
- ***uniqueColorList***: a linked list that stores all the unique colors from the image.

The class also contains the methods:

- ***populateUniqueColorList***: that inserts in the *uniqueColorList* all pixel colors that are unique. The list will NOT contain duplicate colors.
- ***isPresent(Color color)***: returns true if the parameter *color* is present in the *uniqueColorList*.
- ***insertFront(Color color)***: creates a new linked list node where *pixel* refers to the color of the pixel, and inserts the newly created node at the front of the *uniqueColorList*.

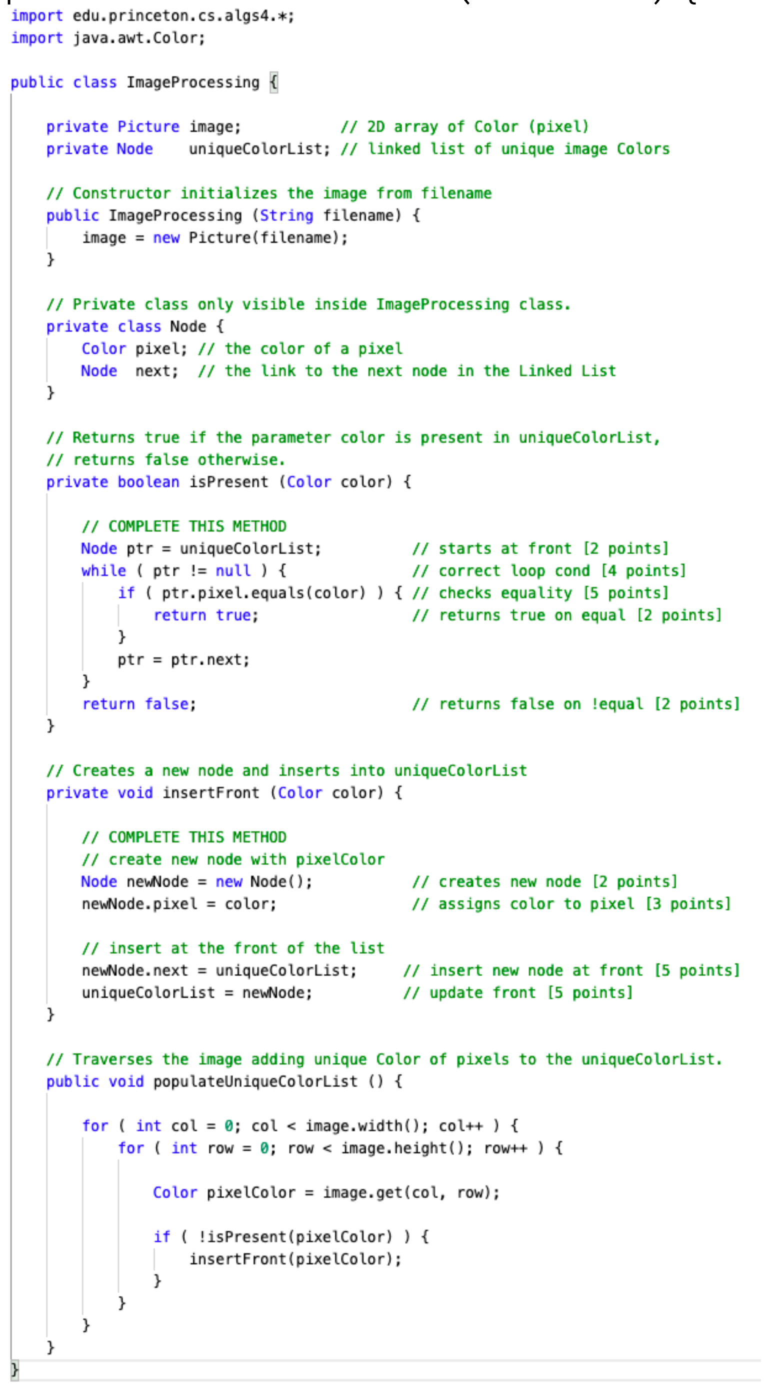
a) **(15 points)** Implement the ***isPresent*** method.

```
private boolean isPresent (Color color) {
```

Name: ______________________________________ NetID: ___________________________

b) **(15 points)** Implement the ***insertFront*** method.

```
private boolean insertFront (Color color) {
```



```
import edu.princeton.cs.algs4.*;
import java.awt.Color;

public class ImageProcessing {

    private Picture image;          // 2D array of Color (pixel)
    private Node    uniqueColorList; // linked list of unique image Colors

    // Constructor initializes the image from filename
    public ImageProcessing (String filename) {
        image = new Picture(filename);
    }

    // Private class only visible inside ImageProcessing class.
    private class Node {
        Color pixel; // the color of a pixel
        Node  next;  // the link to the next node in the Linked List
    }

    // Returns true if the parameter color is present in uniqueColorList,
    // returns false otherwise.
    private boolean isPresent (Color color) {

        // COMPLETE THIS METHOD
        Node ptr = uniqueColorList;          // starts at front [2 points]
        while ( ptr != null ) {              // correct loop cond [4 points]
            if ( ptr.pixel.equals(color) ) { // checks equality [5 points]
                return true;                 // returns true on equal [2 points]
            }
            ptr = ptr.next;
        }
        return false;                        // returns false on !equal [2 points]
    }

    // Creates a new node and inserts into uniqueColorList
    private void insertFront (Color color) {

        // COMPLETE THIS METHOD
        // create new node with pixelColor
        Node newNode = new Node();           // creates new node [2 points]
        newNode.pixel = color;               // assigns color to pixel [3 points]

        // insert at the front of the list
        newNode.next = uniqueColorList;     // insert new node at front [5 points]
        uniqueColorList = newNode;          // update front [5 points]
    }

    // Traverses the image adding unique Color of pixels to the uniqueColorList.
    public void populateUniqueColorList () {

        for ( int col = 0; col < image.width(); col++ ) {
            for ( int row = 0; row < image.height(); row++ ) {

                Color pixelColor = image.get(col, row);

                if ( !isPresent(pixelColor) ) {
                    insertFront(pixelColor);
                }
            }
        }
    }
}
```

Name: ______________________________ NetID: ______________________

Problem 5 – Binary Search Tree (BST) (30 points)

(a) **(7 points)** After inserting 18, 51, 37, 11, 46, 25, and 20 into an empty BST in that order, what would be the worst case number of comparisons (`compareTo` calls) for a successful search?
5 comparisons

```
    18          There is no need to draw the tree
11        51
        37
     25    46
   20
```

(b) **(8 points)** If we perform searches for 1, 2, 6, 7, 9, 11, 20 in the following BST, what would be the average number of comparisons (`compareTo` calls), regardless of whether the search ends in success or failure? Give the reasoning.

```
  7
 / \
4   9
 \
  6
```

- 2 comparisons // 3 pts
- Average #comparisons = (2+2+3+1+2+2+2)/7=2 // 5 pts

Search fails for 1, 2, 11, 20: 2 comparisons each
Search succeeds for 6, 7, 9: 3, 1, 2 comparisons respectively

(c) **(8 points)** In the worst case scenario, what would be the time complexity to delete the node containing the largest value in a BST? Give the reasoning.

- 3 points: O(n)
- 5 points: In the worst case, the tree would be skewed to the right with the largest element being the only leaf node on the right side. Therefore, you would have to iterate through all the elements to get to the largest element.

(d) **(7 points)** Given the following numbers to insert into an empty BST: 2, 7, 8, 10, 15, 20. What insertion order would yield the tree with the least height? D

A. 15, 2, 20, 8, 7, 10
B. 8, 20, 7, 2, 15, 10
C. 7, 2, 10, 8, 15, 20
D. 10, 7, 15, 20, 2, 8