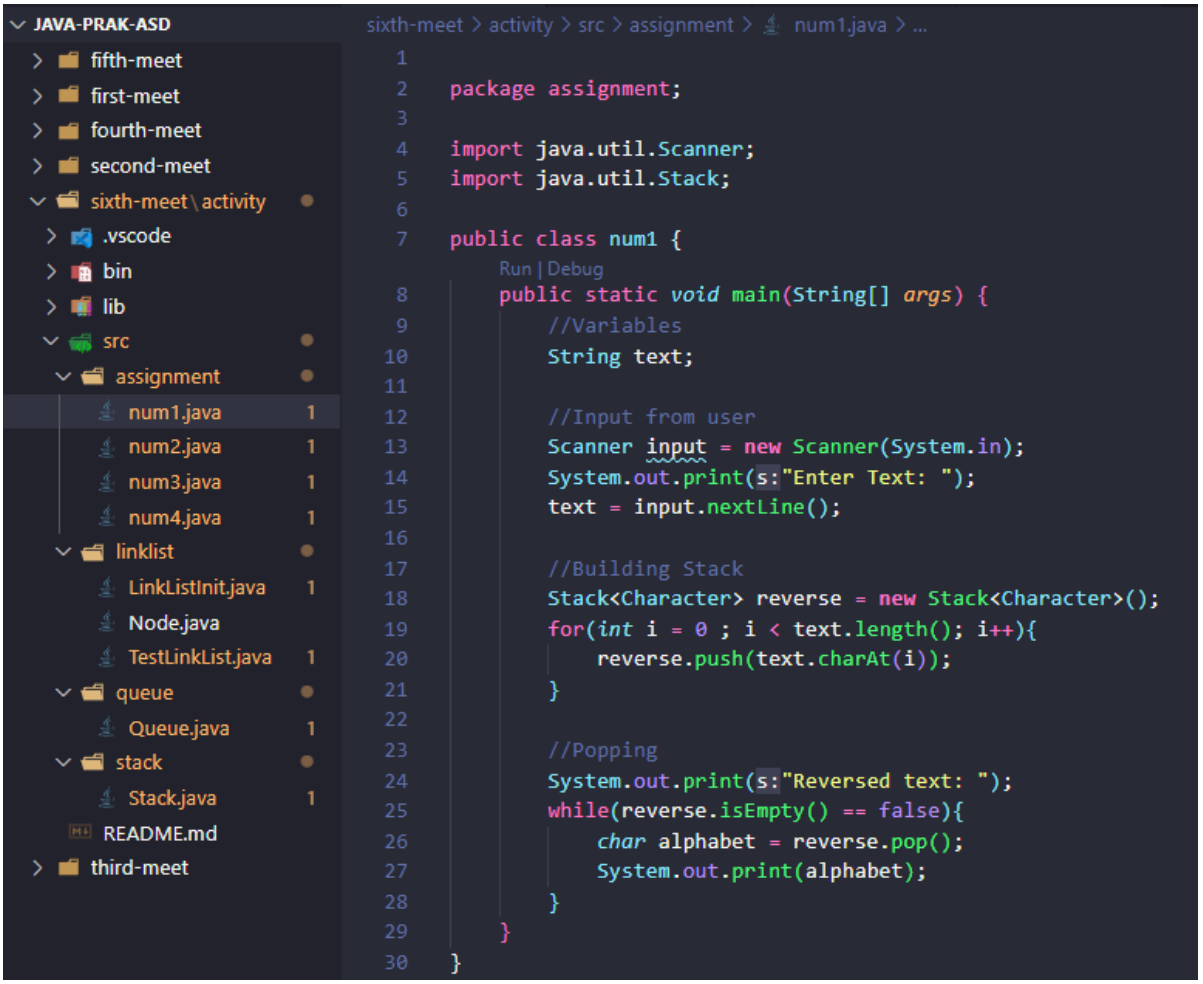
Homework 7

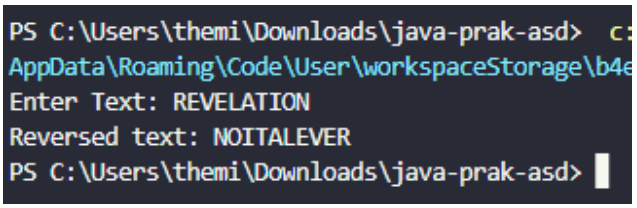
Ramzy Izza Wardhana - 21/472698/PA/20322

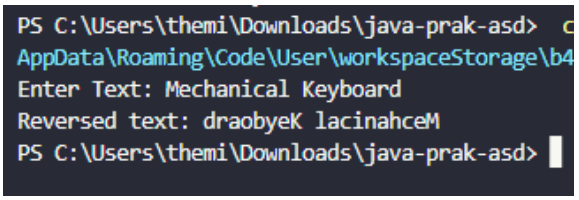
1) Do Task number 1 & 2 on the module. May use methods from the built-in Stack Class, but may not use methods from other classes (including String).

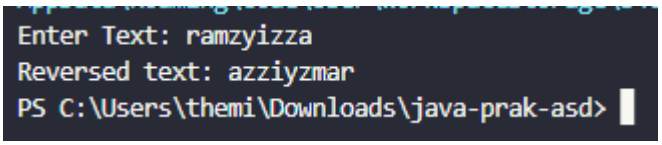
a. Create a program to reverse the order of a string of characters inserted by the user (eg REVELATION -> NOITALEVER).

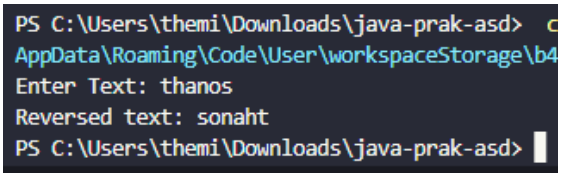


Output

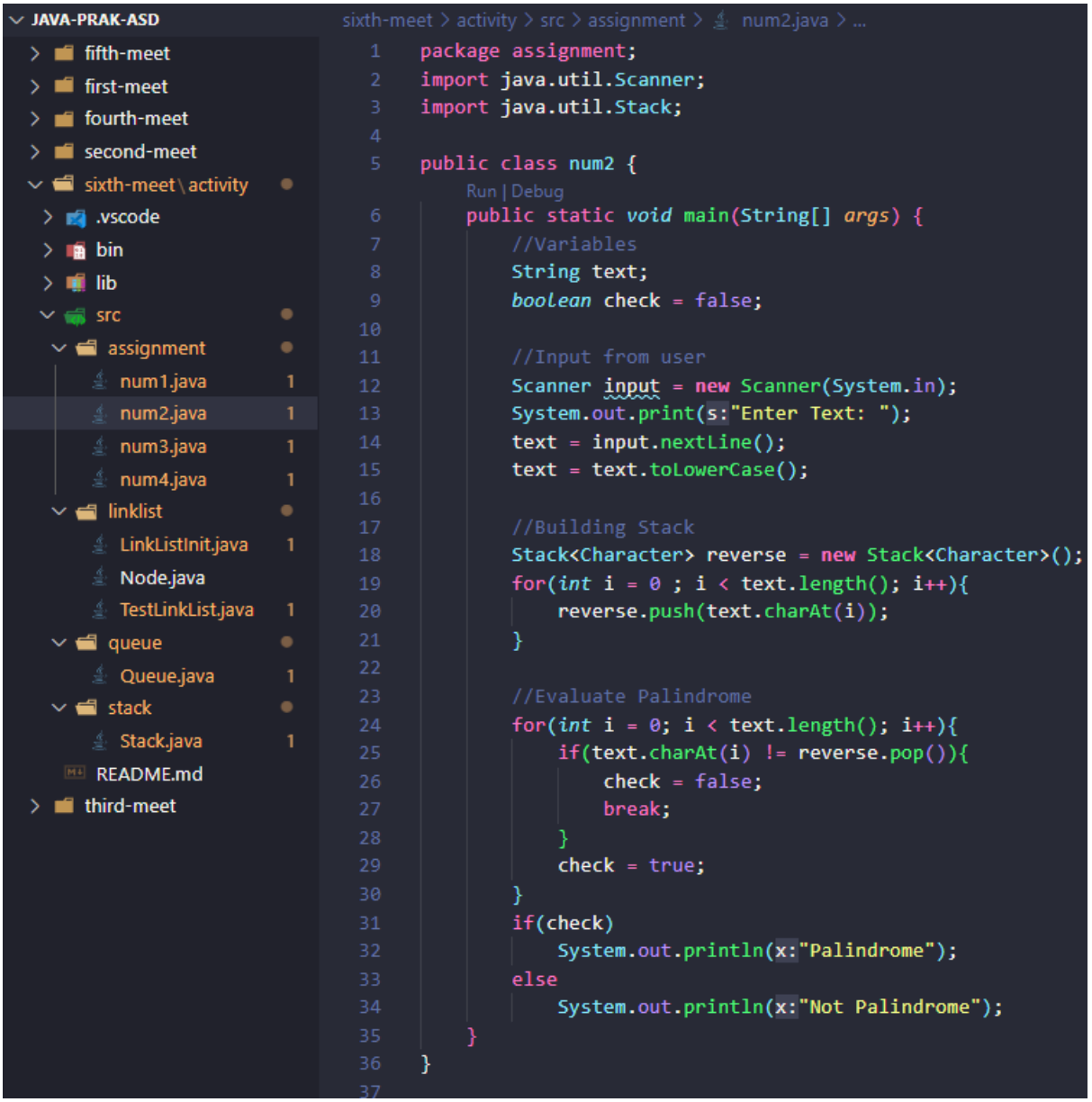




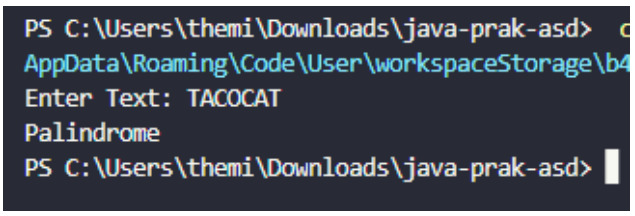


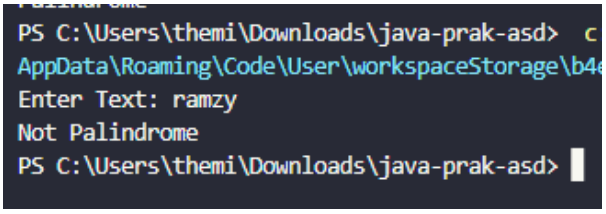


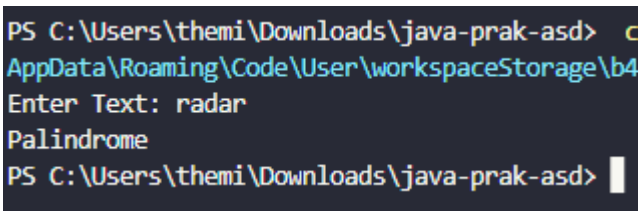
b. Create a program that checks whether a string is a palindrome (a palindrome is a string that reads from forward or backward, for example, TACOCAT).

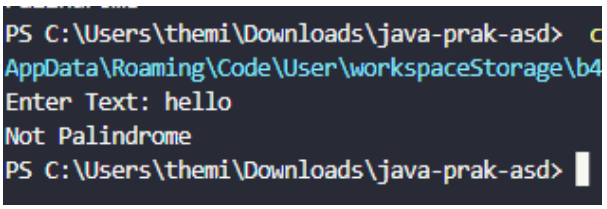


Output:









2) Implement two stacks that share an array. Suppose the first stack is called stackA and the second stack is called stackB. Add methods pushA(), pushB(), popA(), and popB(), to insert/remove elements to/from the A/B stack.

num3.java

```
package assignment;
import java.util.Arrays;
import java.util.Scanner;


class StackInit{
    public final int maxSize;
    public int[] stackArray;
    public int topA;
    public int topB;

    // Since we assumed that two stack A and B shared one array,
    // We initialize in the constructor that stack A will grow
    // from left to right, while stack B will grow from right to left
    // in order to avoid conflicting on reserving each index's element
    // while still maintaining each element of stackA and stackB in the correct
order

    public StackInit(int s){
        maxSize = s;
        stackArray = new int[maxSize];
        topA = -1;
        topB = maxSize;
    }

    public boolean isEmptyA(){
        return (topA == -1);
    }

    public boolean isEmptyB(){
        return (topB == maxSize);
    }

    public void printStackA(){
        System.out.println(Arrays.toString(stackArray));
    }

    public void printStackB(){
        System.out.println(Arrays.toString(stackArray));
    }
```

```
    public void pushA(int a){
        stackArray[++topA] = a;
    }

    public void pushB(int b){
        stackArray[--topB] = b;
    }

    public int popA(){
        return stackArray[topA--];
    }

    public int popB(){
        return stackArray[topB++];
    }

}

public class num3 {
    public static void main(String[] args){
        int stackSize;
        int stackNum;
        Scanner in = new Scanner(System.in);

        while(true){
            System.out.print("Declare size of stack:  ");
            stackSize = in.nextInt();
            if(stackSize % 2 != 0)
                System.out.println("Only accept even size (So that array can be
evenly divided by 2!)");
            else
                break;
        }

        StackInit stack = new StackInit(stackSize);

        System.out.println("Pushing elements to the stack within same array.");

        for (int i = 0; i < stackSize; i++) {
            if (i < stackSize / 2) {
                System.out.print("Enter Number to push to Stack A: ");
                stackNum = in.nextInt();
                stack.pushA(stackNum);
```

```java
            } else {
                System.out.print("Enter Number to push to Stack B: ");
                stackNum = in.nextInt();
                stack.pushB(stackNum);
            }
        }

        System.out.print("Content of Stack A (Shared Array): ");
        stack.printStackA();

        System.out.print("Content of Stack B (Shared Array): ");
        stack.printStackB();

        System.out.println("Popping elements from Stack A:");
        while (!stack.isEmptyA()) {
            System.out.print(stack.popA() + " ");
        }
        System.out.println();

        System.out.println("Popping elements from Stack B:");
        while (!stack.isEmptyB()) {
            System.out.print(stack.popB() + " ");
        }
        System.out.println();
    }
}
```

Output

```
PS C:\Users\themi\Downloads\java-prak-asd>  c:; cd 'c:\Users\themi\Dow
AppData\Roaming\Code\User\workspaceStorage\b4e68668a003ab9bb3a515b7265
Declare size of stack:  9
Only accept even size (So that array can be evenly divided by 2!)
Declare size of stack:  10
Pushing elements to the stack within same array.
Enter Number to push to Stack A: 1
Enter Number to push to Stack A: 2
Enter Number to push to Stack A: 3
Enter Number to push to Stack A: 4
Enter Number to push to Stack A: 5
Enter Number to push to Stack B: 6
Enter Number to push to Stack B: 7
Enter Number to push to Stack B: 8
Enter Number to push to Stack B: 9
Enter Number to push to Stack B: 10
Content of Stack A (Shared Array): [1, 2, 3, 4, 5, 10, 9, 8, 7, 6]
Content of Stack B (Shared Array): [1, 2, 3, 4, 5, 10, 9, 8, 7, 6]
Popping elements from Stack A:
5 4 3 2 1
Popping elements from Stack B:
10 9 8 7 6
PS C:\Users\themi\Downloads\java-prak-asd>
```

3) Implement a queue that can drop elements from the front/back queue. Add two methods: dequeuFront() and dequeueRear() which will respectively remove elements from the front and back queues.

num4.java

```
package assignment;

import java.util.Scanner;
import java.util.Arrays;

class QueueInit{
    private int maxSize;
    private int[] queueArray;
    private int front;
    private int rear;
    private int nItems;

    public QueueInit(int s){
        maxSize = s;
        queueArray = new int[maxSize];
        front = 0;
        rear = -1;
        nItems = 0;
    }

    public void enqueue(int j){
        if (rear == maxSize - 1)
            rear = -1;
        queueArray[++rear] = j;
        nItems++;
    }

    public int dequeueFront(){
        int temp = queueArray[front++];
        if(front == maxSize)
            front = 0;
        nItems--;
        return temp;
    }

    public int dequeueBack(){
        int temp = queueArray[rear--];
        if (rear == -1)
            rear = maxSize - 1;
```

```
        nItems--;
        return temp;
    }

    public boolean isEmpty(){
        return(nItems == 0);
    }

    public boolean isFull(){
        return (nItems == maxSize);
    }

    public void printQueue(){
        System.out.println(Arrays.toString(queueArray));
    }
}


public class num4 {
    public static void main(String[] args){
        int queueSize;
        int numTemp;
        int numChoice = 0;

        Scanner in = new Scanner(System.in);
        System.out.print("Enter queue size: ");
        queueSize = in.nextInt();

        QueueInit theQueue = new QueueInit(queueSize);

        while(numChoice != 4){
            System.out.println("\n 1: Enqueue \t 2: Dequeue Front  \t 3: Dequeue
Back \t 4: End");
            System.out.print("Enter command: ");
            numChoice = in.nextInt();
            if(numChoice == 1) {
                if(theQueue.isFull())
                    System.out.println("Queue is full");
                else{
                    System.out.print("Enter number: ");
                    numTemp = in.nextInt();
                    theQueue.enqueue(numTemp);
                }
            }
```

```
            else if(numChoice == 2){
                if(theQueue.isEmpty())
                    System.out.println("Queue is Empty");
                else{
                    numTemp = theQueue.dequeueFront();
                    System.out.println("Dequeued Value: " + numTemp);
                }
            }
            else if(numChoice == 3){
                if(theQueue.isEmpty())
                    System.out.println("Queue is Empty");
                else{
                    numTemp = theQueue.dequeueBack();
                    System.out.println("Dequeued Value: " + numTemp);
                }
            }

            else if(numChoice != 4){
                System.out.println("Wrong Command!");
            }
        }
        System.out.println("Program Terminated");
    }
}
```

Output

```
PS C:\Users\themi\Downloads\java-prak-asd>  c:; cd 'c:\Users\themi\Downloads\java-
AppData\Roaming\Code\User\workspaceStorage\b4e68668a003ab9bb3a515b72655776d\redhat
Enter queue size: 4

 1: Enqueue      2: Dequeue Front        3: Dequeue Back         4: End
Enter command: 1
Enter number: 1

 1: Enqueue      2: Dequeue Front        3: Dequeue Back         4: End
Enter command: 1
Enter number: 2

 1: Enqueue      2: Dequeue Front        3: Dequeue Back         4: End
Enter command: 1
Enter number: 3

 1: Enqueue      2: Dequeue Front        3: Dequeue Back         4: End
Enter command: 1
Enter number: 4

 1: Enqueue      2: Dequeue Front        3: Dequeue Back         4: End
Enter command: 1
Queue is full

 1: Enqueue      2: Dequeue Front        3: Dequeue Back         4: End
Enter command: 2
Dequeued Value: 1

 1: Enqueue      2: Dequeue Front        3: Dequeue Back         4: End
Enter command: 3
Dequeued Value: 4

 1: Enqueue      2: Dequeue Front        3: Dequeue Back         4: End
Enter command: 2
Dequeued Value: 2

 1: Enqueue      2: Dequeue Front        3: Dequeue Back         4: End
Enter command: 3
Dequeued Value: 3
Program Terminated
PS C:\Users\themi\Downloads\java-prak-asd>
```