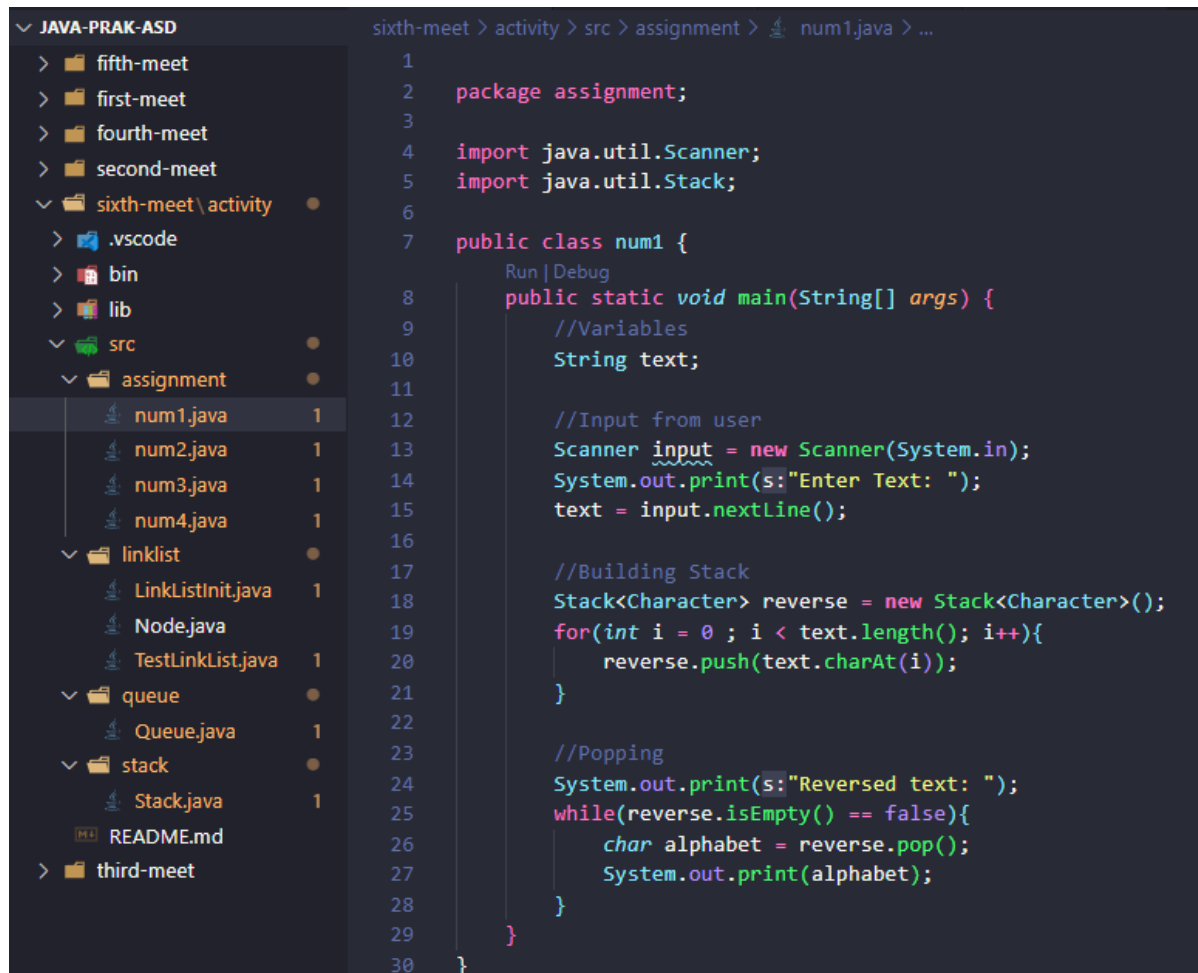


## Homework 7

Ramzy Izza Wardhana - 21/472698/PA/20322

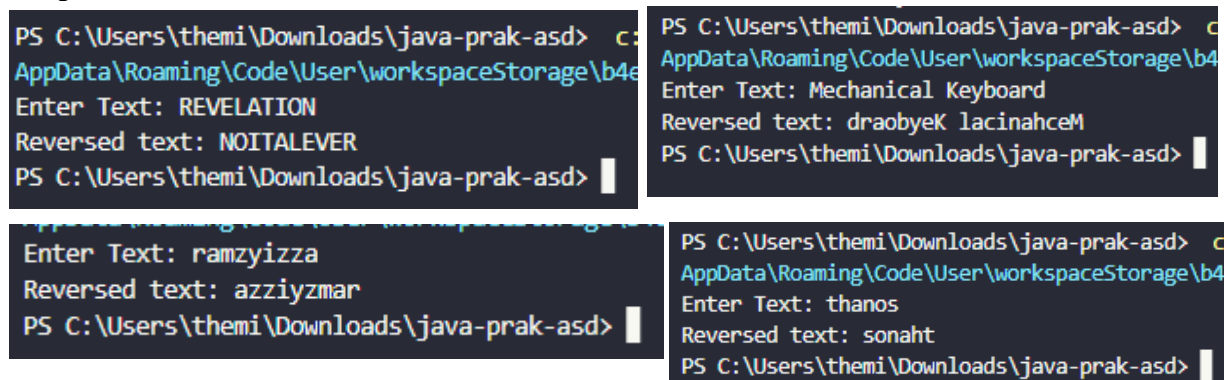
1) Do Task number 1 & 2 on the module. May use methods from the built-in Stack Class, but may not use methods from other classes (including String).

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



```
sixth-meet > activity > src > assignment > num1.java > ...
1
2 package assignment;
3
4 import java.util.Scanner;
5 import java.util.Stack;
6
7 public class num1 {
8     //Variables
9     String text;
10
11     //Input from user
12     Scanner input = new Scanner(System.in);
13     System.out.print(s:"Enter Text: ");
14     text = input.nextLine();
15
16     //Building Stack
17     Stack<Character> reverse = new Stack<Character>();
18     for(int i = 0 ; i < text.length(); i++){
19         reverse.push(text.charAt(i));
20     }
21
22     //Popping
23     System.out.print(s:"Reversed text: ");
24     while(reverse.isEmpty() == false){
25         char alphabet = reverse.pop();
26         System.out.print(alphabet);
27     }
28 }
29
30 }
```

### Output



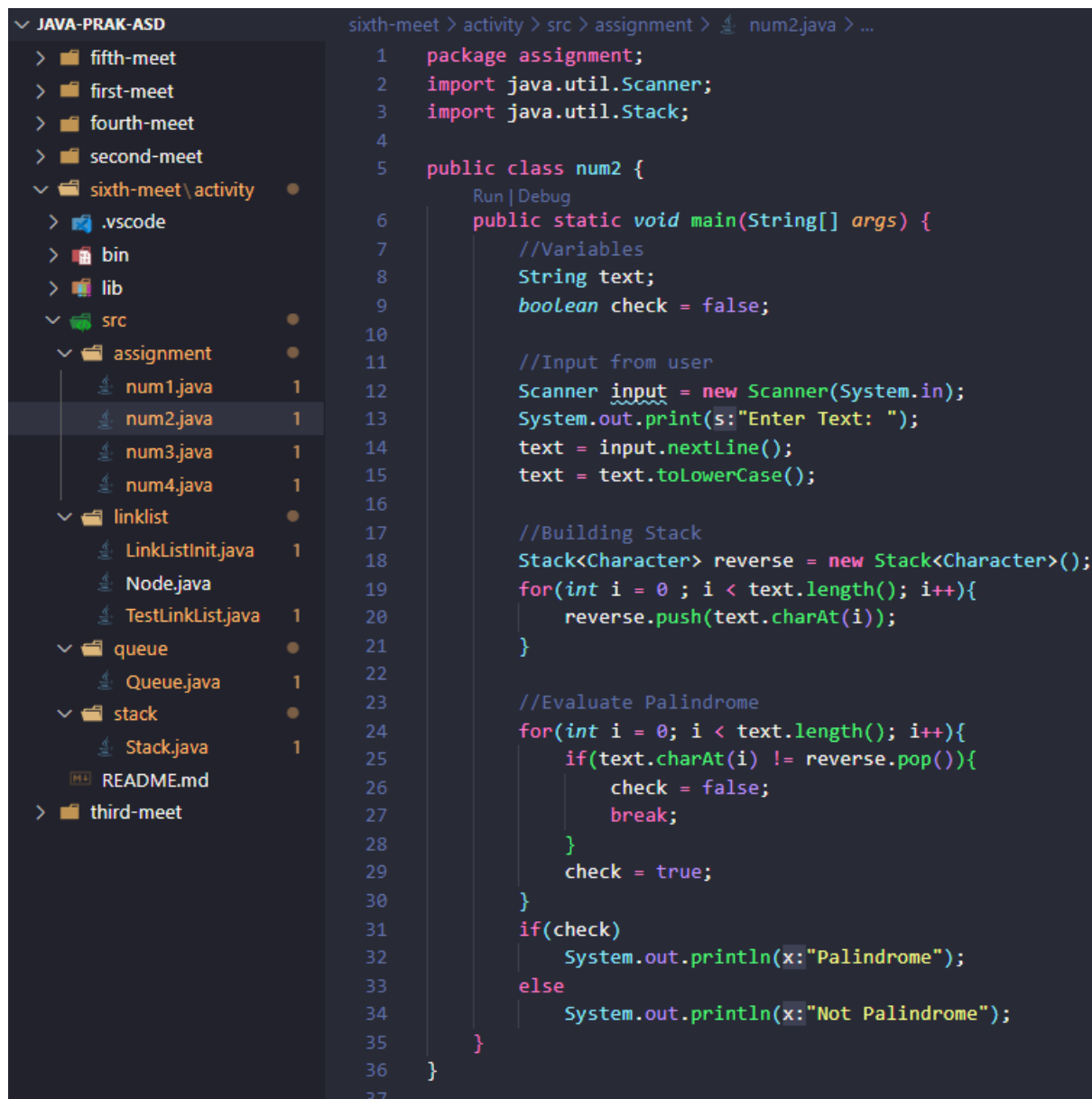
```
PS C:\Users\themi\Downloads\java-prak-asd> cd C:\Users\themi\Downloads\java-prak-asd
AppData\Roaming\Code\User\workspaceStorage\b4e
Enter Text: REVELATION
Reversed text: NOITALEVER
PS C:\Users\themi\Downloads\java-prak-asd>

PS C:\Users\themi\Downloads\java-prak-asd> cd C:\Users\themi\Downloads\java-prak-asd
AppData\Roaming\Code\User\workspaceStorage\b4e
Enter Text: Mechanical Keyboard
Reversed text: draobyek lacinahceM
PS C:\Users\themi\Downloads\java-prak-asd>

PS C:\Users\themi\Downloads\java-prak-asd> cd C:\Users\themi\Downloads\java-prak-asd
AppData\Roaming\Code\User\workspaceStorage\b4e
Enter Text: ramzyizza
Reversed text: azziyzmar
PS C:\Users\themi\Downloads\java-prak-asd>

PS C:\Users\themi\Downloads\java-prak-asd> cd C:\Users\themi\Downloads\java-prak-asd
AppData\Roaming\Code\User\workspaceStorage\b4e
Enter Text: thanos
Reversed text: sonaht
PS C:\Users\themi\Downloads\java-prak-asd>
```

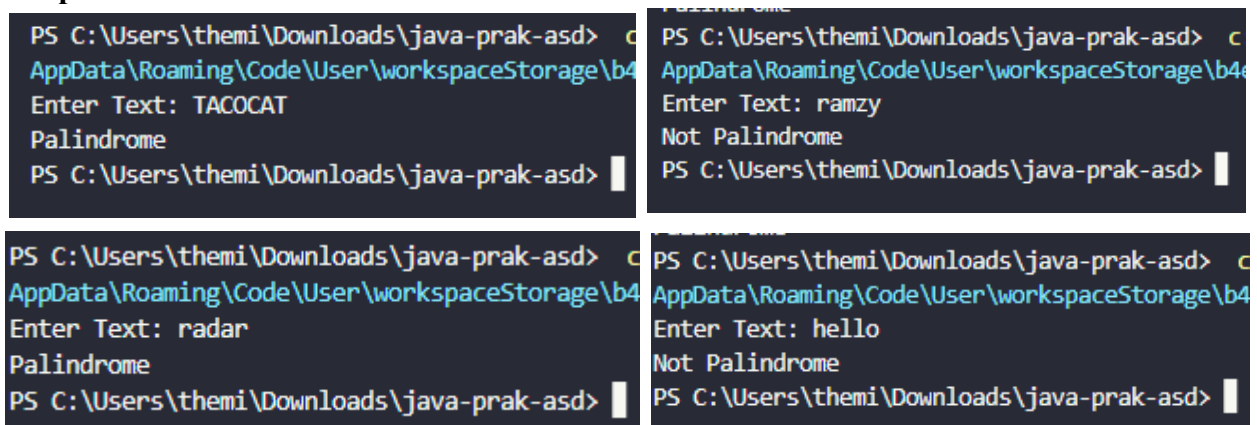
- b. Create a program that checks whether a string is a palindrome (a palindrome is a string that reads from forward or backward, for example, TACOCAT).



The screenshot shows an IDE with a project named 'JAVA-PRAK-ASD'. The file explorer on the left shows a directory structure with folders like 'fifth-meet', 'first-meet', 'fourth-meet', 'second-meet', 'sixth-meet', 'linklist', 'queue', and 'stack'. The 'sixth-meet' folder is expanded, showing a sub-folder 'activity' which contains 'num2.java'. The main editor displays the code for 'num2.java'.

```
sixth-meet > activity > src > assignment > num2.java > ...  
1 package assignment;  
2 import java.util.Scanner;  
3 import java.util.Stack;  
4  
5 public class num2 {  
6     //Variables  
7     String text;  
8     boolean check = false;  
9  
10    //Input from user  
11    Scanner input = new Scanner(System.in);  
12    System.out.print(s:"Enter Text: ");  
13    text = input.nextLine();  
14    text = text.toLowerCase();  
15  
16    //Building Stack  
17    Stack<Character> reverse = new Stack<Character>();  
18    for(int i = 0 ; i < text.length(); i++){  
19        reverse.push(text.charAt(i));  
20    }  
21  
22    //Evaluate Palindrome  
23    for(int i = 0; i < text.length(); i++){  
24        if(text.charAt(i) != reverse.pop()){  
25            check = false;  
26            break;  
27        }  
28        check = true;  
29    }  
30    if(check)  
31        System.out.println(x:"Palindrome");  
32    else  
33        System.out.println(x:"Not Palindrome");  
34 }  
35  
36  
37
```

Output:



The output shows four separate runs of the program in a Windows Command Prompt. Each run prompts the user to 'Enter Text:' and then prints either 'Palindrome' or 'Not Palindrome'.

```
PS C:\Users\them1\Downloads\java-prak-asd> c:\ProgramData\Microsoft\WindowsApps\code.exe  
AppData\Roaming\Code\User\workspaceStorage\b4  
Enter Text: TACOCAT  
Palindrome  
PS C:\Users\them1\Downloads\java-prak-asd>  
  
PS C:\Users\them1\Downloads\java-prak-asd> c:\ProgramData\Microsoft\WindowsApps\code.exe  
AppData\Roaming\Code\User\workspaceStorage\b4  
Enter Text: ramzy  
Not Palindrome  
PS C:\Users\them1\Downloads\java-prak-asd>  
  
PS C:\Users\them1\Downloads\java-prak-asd> c:\ProgramData\Microsoft\WindowsApps\code.exe  
AppData\Roaming\Code\User\workspaceStorage\b4  
Enter Text: radar  
Palindrome  
PS C:\Users\them1\Downloads\java-prak-asd>  
  
PS C:\Users\them1\Downloads\java-prak-asd> c:\ProgramData\Microsoft\WindowsApps\code.exe  
AppData\Roaming\Code\User\workspaceStorage\b4  
Enter Text: hello  
Not Palindrome  
PS C:\Users\them1\Downloads\java-prak-asd>
```

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);
            }
        }
    }
}

```

```

        } 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\them1\Downloads\java-prak-asd> c:: cd 'c:\Users\them1\Down
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\them1\Downloads\java-prak-asd>

```

3) Implement a queue that can drop elements from the front/back queue. Add two methods: dequeueFront() 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> 
```