Test Case for Use

Case 6

```
attend.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String subjectName = null;
        String querySubject = "SELECT subject name FROM subjects
WHERE idsubjects = ?";
        try (Connection connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/softengin23
24", "root", "W45@jqr#8CX");
             PreparedStatement preparedStatementSubject =
connection.prepareStatement(querySubject)) {
            preparedStatementSubject.setString(1, id); // Assuming id
is the subject ID
            try (ResultSet resultSet =
preparedStatementSubject.executeQuery()) {
                if (resultSet.next()) {
                    subjectName = resultSet.getString("subject name");
4
                } else {
                    return; // Exit the action listener if no subject
is found
                }
            }
        } catch (SQLException ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(null, "Database error: " +
ex.getMessage());
           return; // Exit the action listener if there is a
database error
        }
        String queryMeeting = "SELECT every1, startTime, status FROM
meetings schedule subjects WHERE meetingsAbout = ?";
        try (Connection connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/softengin23
24", "root", "W45@jqr#8CX");
             PreparedStatement preparedStatementMeeting =
connection.prepareStatement(queryMeeting)) {
```

```
preparedStatementMeeting.setString(1, subjectName); 8
            try (ResultSet resultSet =
preparedStatementMeeting.executeQuery()) {
                if (resultSet.next()) {
                    String every1 = resultSet.getString("every1");
                    String startTimeStr =
resultSet.getString("startTime");
                    String status = resultSet.getString("status");
                    // Check the status
                    if ("canceled".equalsIgnoreCase(status)) {
                        JOptionPane.showMessageDialog(null, "The
lecture has been canceled. Let me redirect you to studentMenu");
                            Timer timer = new Timer(1000, r \rightarrow {} {}
                                dispose();
                                new studentMenuu(null, user);
                            });
                            timer.setRepeats(false);
                            timer.start();
                    } else if ("scheduled".equalsIgnoreCase(status))
{
                        // Get the current day name and time
                        String todayDayName =
DayOfWeek.from(LocalDate.now()).name().toLowerCase();
                        LocalTime startTime =
LocalTime.parse(startTimeStr, DateTimeFormatter.ofPattern("h:mm a",
Locale.ENGLISH));
                        LocalTime currentTime = LocalTime.now();
                        // Calculate the difference in minutes
                        long minutesDifference =
java.time.Duration.between(currentTime, startTime).toMinutes();
                        if
(every1.toLowerCase().contains(todayDayName) && minutesDifference >
30) {
                                 JOptionPane.showMessageDialog(null,
"The lecture will begin in " + minutesDifference + " minutes.");
```

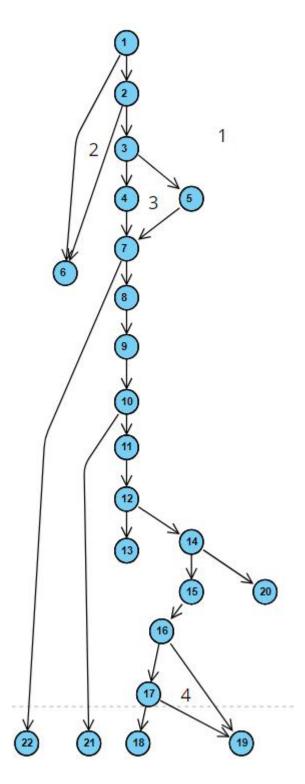
} else {

```
dispose(); 19
    new LobbyGUIStudent(null, user, id);
} else {
    dispose(); 20
    new LobbyGUIStudent(null, user, id);
} else {
    21

JOptionPane.showMessageDialog(null, "No meeting schedule found.");
} catch (SQLException ex) {
    ex.printStackTrace(); 22
    JOptionPane.showMessageDialog(null, "Database error: " + ex.getMessage());
}};
```

Για τι**ς** συναρτ**ή**σει**ς** addUserToLobby, removeUserFromLobby και acceptAllStudents που καλο**ύ**νται με την ε**ί**σοδο του Φοιτητ**ή** στο Λ**ό**μπι και **ό**ταν ο Καθηγητ**ής** δ**έ**χεται τον Φοιτητ**ή** αντ**ί**στοιχα, δεν χρει**ά**στηκε να δημιουργηθε**ί** κ**ά**ποιο**ς** γρ**ά**φο**ς**, καθ**ώς** η ρο**ή** του κ**ώ**δικ**ά** του**ς** ε**ί**ναι ενια**ί**α και συνεχ**ής**.

Στην επ**ό**μενη σελ**ί**δα παρουσι**ά**ζεται ο Γρ**ά**φο**ς** για το actionListener..



Σύμφωνα με τον τύπο: V(g) = e-n+2 =

= 24-22+2

= 4 περιοχές, όπως φαίνεται και στον γράφο. Εφαρμόζουμε τον αλγόριθμο για εύρεση μονοπατιών. Ξεκινάμε με το συντομότερο έγκυρο μονοπάτι:

M1: 1-6

Ακολουθούν τα υπόλοιπα μονοπάτια που προκύπτουν με προσθήκη ακμών στο Μ1.

M2: 1-2-3-4-7-8-9-10-11-12-13 (Alternative Flow 2)

 $\texttt{M3:} \ 1-2-3-4-7-8-9-10-11-12-14-15-16-19 \ (\textbf{steps of Main Flow})$

M4: 1-2-3-4-7-8-9-10-11-12-14-15-16-17-18 (Alternative Flow 1)

Pa	Περι	Περίπτωση	Αναμεν ό με
th	γραφή	Ελέγχου	v o
	-		Αποτ έ λεσμ
			α (Έξοδος
			Προγράμμα
			τος)
M	Έλεγχ	Connection conn,	JOptionPane. showMe
2	ος	resultSet.next(),	ssageDialog(null,
	για	("canceled".equalsIgn	The lecture has
	σ ύ νδε	oreCase(status)) ==	been canceled. Let
	ση	true	me redirect you to
	στη		studentMenu");
	Βάση,		
	αν ά κτ		
	ηση		
	δεδο		
	μ έ νων		
	για		
	την		
	Κλ ή ση		
	, η		
	κ λ ή σ η		
	έ Χει		

M Έλεγχ Connection conn, για ("scheduled".equalsIg noreCase(status) == ση true, στη everyl.toLowerCase(). ανάκτ η δεδο μένων για την Κλήση, , ο χρήστ ης εισέρ χετα ι στο Λόμπι Μ4 Έλεγχ Connection conn, ος resultSet.next(), ("canceled".equalsIgnore (σase(status) == false, ποη δεδο κατα κατα κατα κατα κατα κατα κατα κατ				
M Έλεγχ Connection conn,		αкυρ		
3 ος για σύνδε ση τesultSet.next(), ("scheduled".equalsIg noreCase(status) == true, everyl.toLowerCase(). Bάση, ανάκτ ηση δεδο μένων για την Κλήση, , ο χρήστ ης εισέρ χετα ιστο Λόμπι M4 Έλεγχ Connection conn, ος για τατο Κάση τος κατα κατα κατα κατα κατα κατα κατα κατ		ωθεί		
γ ι α		Έλεγχ	Connection conn,	new
σύνδε noreCase (status) == true, everyl.toLowerCase (). Bάση, ανάκτ false each of the fal	3	ο ς	resultSet.next(),	LobbyGUIStudent(nu
true, στη Βάση, ανάκτ ηση δεδο μένων για την Κλήση , ο χρήστ ης εισέρ χετα ι στο Λόμπι Μ4 Έλεγχ Connection conn, ος για τος για ("canceled".equalsIgnore ὑπαρξ Case(status) == false, "The lecture will begin in " +		για	("scheduled".equalsIg	11, user, id);
στη everyl.toLowerCase(). Βάση, ανάκτ πηση δεδο μένων για την Κλήση, ο χρήστ πος εισέρ χετα ι στο Λόμπι Μ4 Έλεγχ Connection conn, ος resultSet.next(), για ("canceled".equalsIgnore ύπαρξ ("case(status) = false, "The lecture will begin in " + "The lecture will be		σ ύ νδε	noreCase(status) ==	
Bάση, ανάκτ ηση δεδο μένων για την Κλήση , ο χρήστ ης εισέρ χετα ιστο Λόμπι M4 Έλεγχ Connection conn, ος resultSet.next(), για ("canceled".equalsIgnore ύπαρξ Case(status) = false, στο λομπι Βάση, ανάκτ επίση Επίση Επίση Επίση Ενωνον Αόμπι Ιοτο Αόμπι Ενωνον Ενείγι. tollowercase(). contains (todayDayName == false για την Κλήση , ο Χρήστ ης εισέρ χετα ιστο Αόμπι Μ4 Έλεγχ Connection conn, σς resultSet.next(), για ("canceled".equalsIgnore ύπαρξ Case(status) == false, begin in " +		ση	true,	
ανάκτ ηση δεδο μένων για την Κλήση , ο χρήστ ης εισέρ χετα ιστο Λόμπι Μ4 Έλεγχ Connection conn, ος για ("canceled".equalsIgnore ὑπαρξ Case(status) = false,			every1.toLowerCase().	
α ν άκ τ == false		Βάση,	contains(todayDayName	
δεδο $μένων$ $για$ $την$ $Κλήση$, $ο$ $χρήστ$ $ης$ $εισέρ$ $χετα$ $ιστο$ $Λόμπι$ M4 $Έλεγχ$ Connection conn, $ος$ $για$ $("canceled".equalsIgnore$ $ΰπαρξ$ Case(status) == false, $σεω$ $σ$		αν ά κτ		
μένων $για$ $την$ $Κλήση$, $ο$ $χρήστ$ $ης$ $εισέρ$ $χετα$ $ιστο$ $Λόμπι$ M4 $Έλεγχ$ Connection conn, $ος$ $για$ $για$ $για$ $("canceled".equalsIgnore ύπαρξ Case(status) = false, ("π)$				
για $την$ $Κλήση$, $ο$ $χρήστ$ $ης$ $εισέρ$ $χετα$ $ιστο$ $Λόμπι$ M4 $Έλεγχ$ Connection conn, $ος$ $για$ $("canceled".equalsIgnore)$ $ύπαρξ$ Case(status) = false, $("angle below of the content of$		δεδο		
την $Κλήση$, $ο$ $χρήστ$ $ης$ $εισέρ$ $χετα$ $ιστο$ $Λόμπι$ M4 $Έλεγχ$ Connection conn, $ος$ $για$ $ΰπαρξ$ Case(status) = false, $στο$ $σ$		μένων		
$K \lambda \dot{\eta} \sigma \eta$, ο $X \rho \dot{\eta} \sigma \tau$ $\eta \varsigma$ $\varepsilon \iota \sigma \dot{\varepsilon} \rho$ $X \varepsilon \tau \alpha$ $\iota \sigma \tau \sigma$ $\Lambda \dot{\sigma} \mu \pi \iota$ M4 $E \lambda \varepsilon \gamma X$ Connection conn, $\sigma \varsigma$ resultSet.next(), $\gamma \iota \alpha$ ("canceled".equalsIgnore $\dot{\upsilon} \pi \alpha \rho \xi$ Case(status) == false, $\sigma \dot{\tau} \dot{\tau} \dot{\tau} \dot{\tau} \dot{\tau} \dot{\tau} \dot{\tau} \dot{\tau}$		για		
χ ρ $\hat{\eta}$ σ τ η ζ ε ι σ $\hat{\varepsilon}$ ρ χ ε τ α ι σ τ ρ χ ε τ α ι σ τ ρ χ ε χ		την		
$x \rho \dot{\eta} \sigma \tau$ $\eta \varsigma$ $\epsilon \iota \sigma \dot{\epsilon} \rho$ $x \epsilon \tau \alpha$ $\iota \sigma \tau \sigma$ $\Lambda \dot{\sigma} \mu \pi \iota$ M4 Έλεγ x Connection conn, $\sigma \varsigma$ $\gamma \iota \alpha$ $\sigma \tau \sigma$ $\sigma \tau \sigma$ $\sigma \tau \sigma$ $\sigma \tau \sigma \tau \sigma \sigma$		Κλ ή ση		
		, 0		
εισέρ χετα ιστο Λόμπι M4 Έλεγχ Connection conn, ος resultSet.next(), για ("canceled".equalsIgnore ὑπαρξ Case(status) == false, begin in " +		χρ ή στ		
x ε τ α t σ τ ο Λ $\dot{\mathbf{o}}$ μ π t M4 $\dot{\mathbf{E}}$ λ ε γ x Connection conn, t \mathbf{o} \mathbf{c} resultSet.next(), t \mathbf{o} \mathbf{c} resultSet.next(), t \mathbf{o} \mathbf{c} ("canceled".equalsIgnore \mathbf{c}		ης		
λόμπι M4 $ Ελεγχ $ Connection conn, $ ος $ resultSet.next(), $ για $ ("canceled".equalsIgnore $ ύπαρξ $ Case(status) == false, $ στο $ begin in " +		εισ έ ρ		
Λ ό μ π ι $M4$ Έ λ ε γ χ Connection conn, O γ ι α O γ ι α O γ ι α O γ α ρ ξ Case(status) == false, O γ ι α O γ ι α O γ ι α O γ ι α O γ O γ ι α O γ		χετα		
M4 Έλεγχ Connection conn, JOptionPane. show \circ resultSet. next(), ssageDialog(null "The lecture will begin in " +		ι στο		
		Λόμπι		
γ ι α	M4	Έλεγχ	Connection conn,	JOptionPane. showMe
Ú π α ρ ξ		ο ς	resultSet.next(),	ssageDialog(null,
(" 1 1 1 1" 1 T		για	("canceled".equalsIgnore	"The lecture will
		ύ παρξ		begin in "+
		η	("scheduled".equalsIgnor	minutesDifference
γ ρ α μ eCase(status)) == true + " minutes.");		γραμ		+ " minutes.");
μής everyl.toLowerCase().con tains(todayDayName) &&		μ ής		
κάμερ minutesDifference > 30		κάμερ		
$\alpha \varsigma \eta = true$		ας η		
οποία		ο π ο ία		

ε ί ναι κλει στ ή	
κ λ ε ι	
στή	