TEST-CASES-V0.1

ΜΕΛΗ ΟΜΑΔΑΣ=

Γαρουφαλής Σπύρος	ΑΜ:1067460	5ο έτος
Κακαβά Θεοδώρα	ΑΜ:1070918	5ο έτος
Παπασπύρου Αριστέα	ΑΜ:1070739	5ο έτος
Τζερμιά Ασπασία	ΑΜ:1067455	5ο έτος
Χαραλαμποπούλου Σωτηρία	ΑΜ:1070924	5ο έτος

Editor=Γαρουφαλής Σπυρίδων

Peer Reviewer= Θεοδώρα Κακαβά

κλάσεις = 1)Προσθήκη κάρτας , 2)Προσθήκη λογαριασμού 3)Πληρωμή λογαριασμού

Με βάση αυτές τις πληροφορίες δημιουργούνται οι περιπτώσεις ελέγχου=

Προσθήκη Κάρτας

Μεταβλητή εισόδου(είδος)	**Κλάσεις ισοδυναμίας**	**Περιπτώσεις ελέγχου**
Όνομα(σύνολο τιμών)	Όνομα in {Έγκυρο ‘Ονομα} Όνομα not in {Έγκυρο ‘Ονομα}	‘Ονομα = M. PAPAS (υπάρχει στην βάση) ‘Ονομα = S. KARRA (δεν υπάρχει στην βάση)
Κάρτα ID (σύνολο τιμών)	Κάρτα ID in {Έγκυρη Κάρτα ID} Κάρτα ID not in {Έγκυρη ‘Κάρτα ID}	Κάρτα ID =1234 5678 9000 (υπάρχει στην βάση) Κάρτα ID =1234 58 (δεν υπάρχει στην βάση)
CVV (σύνολο τιμών)	CVV in {Έγκυρο CVV} CVV not in {Έγκυρο CVV}	CVV= 566(υπάρχει στην βάση) CVV=834(δεν υπάρχει στην βάση)

Ημερομηνία (σύνολο τιμών)	Ημερομηνία in {Έγκυρη Ημερομηνία} Ημερομηνία not in {Έγκυρη Ημερομηνία}	Ημερομηνία=05/23(υπάρχει στην βάση) Ημερομηνία=01/16(δεν υπάρχει στην βάση)

Να σημειωθεί οτι οι μεταβλητές εισόδου οι οποίες μπορούν να είναι έγκυρές η άκυρες είναι υλοποιημένες έτσι καθώς η βασική ιδέα είναι ότι ελέγχονται από τη βάση δεδομένων εάν είναι αποδεκτές η όχι οι τιμές που καταχωρούνται .

Προσθήκη Λογαριασμού

Μεταβλητή εισόδου(είδος)	**Κλάσεις ισοδυναμίας**	**Περιπτώσεις ελέγχου**
Όνομα(σύνολο τιμών)	Όνομα in {Έγκυρο ‘Ονομα} Όνομα not in {Έγκυρο ‘Ονομα}	‘Ονομα = Γιώργος(υπάρχει στην βάση) ‘Ονομα = Μιχαλόγλου(δεν υπάρχει στην βάση)
Επώνυμο(σύνολο τιμών)	Επώνυμο in {Έγκυρο Επώνυμο} Επώνυμο not in {Έγκυρο Επώνυμο}	Επώνυμο= Γεωργιόπουλος(υπάρχει στην βάση) Επώνυμο=Χρίστος (δεν υπάρχει στην βάση)
Αριθμός παροχής (σύνολο τιμών)	Αριθμός παροχής in {Έγκυρος Αριθμός παροχής} Αριθμός παροχής not in {Έγκυρος Αριθμός παροχής}	Αριθμός παροχής=123456789-015(υπάρχει στην βάση) Αριθμός παροχής=987427-91 (δεν υπάρχει στην βάση)

Πληρωμή Λογαριασμού

Μεταβλητή εισόδου(είδος)	**Κλάσεις ισοδυναμίας**	**Περιπτώσεις ελέγχου**
Αριθμός Παροχής (σύνολο τιμών)	Αριθμός παροχής in {Έγκυρος Αριθμός παροχής} Αριθμός παροχής not in {Έγκυρος Αριθμός παροχής}	Αριθμός παροχής=748392757-045 (υπάρχει στην βάση) Αριθμός παροχής=94-9144 (δεν υπάρχει στην βάση)
Αριθμός πληρωμής (σύνολο τιμών)	Αριθμός παροχής in {Έγκυρος Αριθμός παροχής} Αριθμός παροχής not in {Έγκυρος Αριθμός παροχής}	Αριθμός παροχής=285719503-173 (υπάρχει στην βάση) Αριθμός παροχής=98742735644535545-9 (δεν υπάρχει στην βάση)

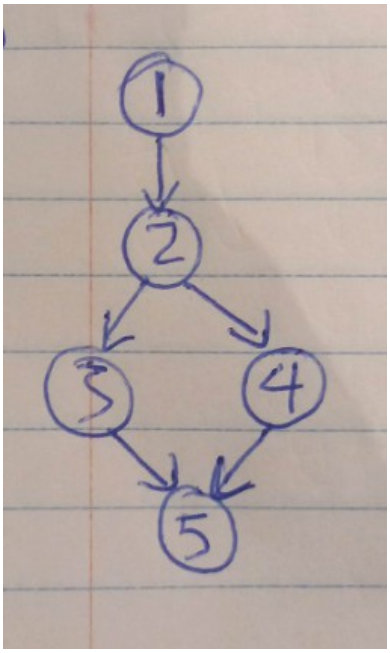
- Σε αυτές τις κλάσεις συμπεριέχονται και υπόλοιπες όπως πχ προσθήκη λογαριασμού τηλεφώνου-ρεύματος-νερού κλπ .

-Όσο αναφορά τον Έλεγχο Διαφανούς Κουτιού (white-box) επιλέχθηκε η τεχνική Δοκιμής βασικών μονοπατιών (basic path testing)=

Test case 1ο=

```
if (o == AddElectricBill.b10)
{
    String username;
        String provider_num = AddElectricBill.t1.getText();

    try {
        String storedProcedure = "{call check_prov(?, ?)}";
        CallableStatement stmt = connection.prepareCall(storedProcedure);
        stmt.setString(1, username);

        stmt.setString(2,provider_num);
        stmt.registerOutParameter(3, java.sql.Types.VARCHAR);
        stmt.execute();
        String outputParameter = stmt.getString(3);
        if (outputParameter =="ok" ) {
            // Login successful, proceed to index page
            String sql = "INSERT INTO bills (electric) VALUES (?)";
            indexP = new IndexPage();
            AddElectricBill.frame.dispose(); // Close the login page
        } else {
            // Login failed, display an error message
            System.out.println("Αποτυχία Αντιστοίχισης");
            PopupMessage popup = new PopupMessage(LogInPage.frame,"Ο λογαριασμός με αυτά τα στοιχεία δεν υπάρχει");
        }

    } catch (SQLException ex) {
        ex.printStackTrace();
    }
```

Κατασκευή γραφήματος ροής=



Όπου 1= try { ,

2= if (outputParameter == ‘ok’) ,

3= String Sql = ‘INSERT INTO bills (electric) VALUES (?)’;

IndexP= new IndexPage();

AddElectricBill.frame.dispose();

4= else {

```
System.out.println('Αποτυχία Αντιστοίχησης');

PopupMessage popup = new PopupMessage(LogInPage.frame, 'Ο λογαριασμός με αυτά τα στοιχεία δεν υπάρχει ');

}

5=catch (SQLException ex) {

  ex.printStackTrace();

}
```

-Υπολογισμός της κυκλωματικής πολυπλοκότητας (V) του γραφήματος ροής (G)=

1ος τρόπος=

V(G)= αριθμός περιοχών του G

Οπότε V(G)=2

2ος τρόπος=

V(G)= E – N + 2 (όπου E: ακμές και N: κόμβοι)

Άρα V(G)=5-5+2=2

3ος τρόπος=

V(G) = P + 1 (όπου P το πλήθος των απλών συνθηκών ελέγχου του κώδικα)

Άρα V(G) =1{απλές συνθήκες ελέγχου (if)}+1=2

Έχουμε βρεί ότι ο αριθμός της κυκλωματικής πολυπλοκότητας του γραφήματος ροής είναι 2 και με τους 3 τρόπους , άρα γνωρίζουμε ότι το άνω όριο του αριθμού των βασικών μονοπατιών που υπάρχουν είναι το 2. Δηλαδή , 2 μονοπάτια αρκούν για να ελέγξουμε όλα τα μονοπάτια. -Για τον εντοπισμό των βασικών μονοπατιών θα ακολουθήσουμε τον συνδιασμό των δύο τρόπων προσέγγισης όταν καταγράφουμε τα βασικά μονοπάτια .

Οπότε προσδιορισμός των βασικών μονοπατιών=

M1=1-2-3-5 Εφικτό=3

M2=1-2-4-5 Εφικτό=4

Συνεπώς το πρόγραμμά μπορεί να ελεγχθεί με 2 βασικά μονοπάτια.

περιπτώσεις ελέγχου=

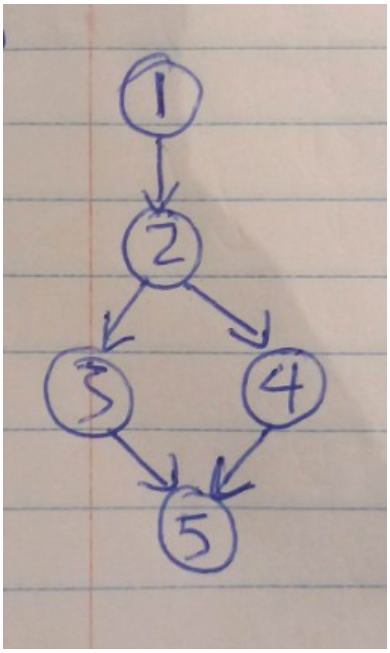
Μονοπατι	OutputParameter=	Αποτέλεσμα=
M1	‘ok’	Εντολές Βάσης
M2	!=’ok’	‘Αποτυχία αντιστοίχησης’

Test case 2ο=

```
if (o == PayElectricBill.b1)
{
    String prov_num;
        String pay_num = PayElectricBill.t1.getText();

    try {
        String storedProcedure = "{call lastest_date(?, ?)}";
        CallableStatement stmt = connection.prepareCall(storedProcedure);
        stmt.setString(1, pay_num);
        stmt.registerOutParameter(2, java.sql.Types.VARCHAR);
        stmt.execute();
        String outputParameter1 = stmt.getString(2);

        if (outputParameter1 == "ok") {
            // Login successful, proceed to index page
            card();
            indexP = new IndexPage();
            PayElectricBill.frame.dispose(); // Close the login page
        } else {
            // Login failed, display an error message
            System.out.println("Ανεπιτυχής Πληρωμή");
            PopupMessage popup = new PopupMessage(LogInPage.frame,"Ο αριθμός παροχής με τον αριθμό πληρωμής δεν αντιστοιχούν");
        }

    } catch (SQLException ex) {
        ex.printStackTrace();
    }

}
```

Κατασκευή γραφήματος ροής=

Παρομοίως με το πρώτο



Όπου 1= try { ,

2= if (outputParameter == ‘ok’) ,

3= card();

IndexP = new IndexPage();

PayElectricBill.frame.dispose();

4= else {

System.out.println(‘Αποτυχής Πληρωμή’);

PopupMessage popup = new PopupMessage(LogInPage.frame, ‘Ο αριθμός παροχής με τον αριθμό πληρωμής δεν αντιστοιχούν‘);

}

5=catch (SQLException ex) {

  ex.printStackTrace();

}

-Υπολογισμός της κυκλωματικής πολυπλοκότητας (V) του γραφήματος ροής (G)=

1ος τρόπος=

V(G)= αριθμός περιοχών του G

Οπότε V(G)=2

2ος τρόπος=

V(G)= E – N + 2 (όπου E: ακμές και N: κόμβοι)

Άρα V(G)=5-5+2=2

3ος τρόπος=

V(G) = P + 1 (όπου P το πλήθος των απλών συνθηκών ελέγχου του κώδικα)

Άρα V(G) =1{απλές συνθήκες ελέγχου (if)}+1=2

Οπότε προσδιορισμός των βασικών μονοπατιών=

M1=1-2-3-5 Εφικτό=3

M2=1-2-4-5 Εφικτό=4

Συνεπώς το πρόγραμμά μπορεί να ελεγχθεί με 2 βασικά μονοπάτια.

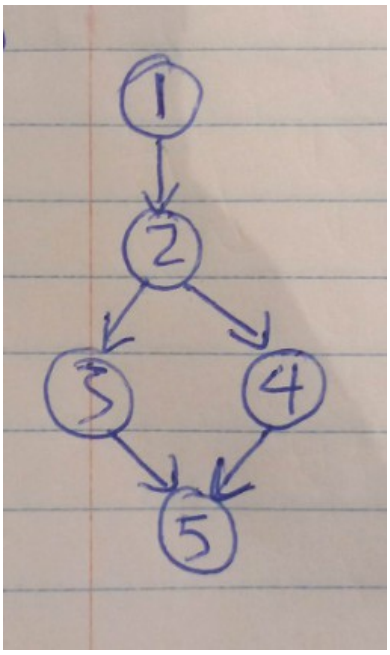
περιπτώσεις ελέγχου=

Μονοπατι	OutputParameter=	Αποτέλεσμα=
M1	‘ok’	Αλλαγή σελίδας
M2	!=’ok’	‘Ανεπιτυχής Πληρωμή’

Test case 3ο=

```
if (o == AddBankAccount.nextButton)
{
    String name = AddBankAccount.fAndLNameText.getText();
       String card_num = AddBankAccount.cardNumText.getText();
        String cvv = AddBankAccount.CVVText.getText();
        String date= AddBankAccount.expDateText.getText();
    try {
        String query = "SELECT * FROM bank WHERE name = ? AND card_num = ? AND cvv = ? AND date =?";
        PreparedStatement preparedStatement = connection.prepareStatement(query);
        preparedStatement.setString(1, name);
        preparedStatement.setString(2, card_num);
        preparedStatement.setString(3, cvv);
        preparedStatement.setString(4, date);
        ResultSet resultSet3 = preparedStatement.executeQuery();
        if (resultSet3.next()) {
            // Login successful, proceed to index page
            String sql = "INSERT INTO bank_info(card_num, date, cvv, name) VALUES (?, ? ,?, ?)";
            indexP = new IndexPage();
            AddBankAccount.frame.dispose(); // Close the login page
        } else {
            // Login failed, display an error message
            System.out.println("Αποτυχία Αντιστοίχισης");
            PopupMessage popup = new PopupMessage(LogInPage.frame,"Ο λογαριασμός με αυτά τα στοιχεία δεν υπάρχει");
        }
        resultSet3.close();
        preparedStatement.close();
    } catch (SQLException ex) {
        ex.printStackTrace();
    }
```

Κατασκευή γραφήματος ροής=



```
Όπου 1= try {     ,
 2= if (resultSet3.next() ) ,
3= String Sql = 'INSERT INTO bank_info(card_num, date, cvv, name) VALUES (?, ?, ?, ?)';
IndexP= new IndexPage();
AddBankAccount.frame.dispose();
4= else {
System.out.println('Αποτυχία Αντιστοίχησης');
PopupMessage popup = new PopupMessage(LogInPage.frame, 'Ο λογαριασμός με αυτά τα στοιχεία δεν υπάρχει ');
}
ResultsSet3.close();
PreparedStatement.close();
5=catch (SQLException ex) {
  ex.printStackTrace();
}
```

-Υπολογισμός της κυκλωματικής πολυπλοκότητας (V) του γραφήματος ροής (G)=

1ος τρόπος=

V(G)= αριθμός περιοχών του G

Οπότε V(G)=2

2ος τρόπος=

V(G)= E – N + 2 (όπου E: ακμές και N: κόμβοι)

Άρα V(G)=5-5+2=2

3ος τρόπος=

V(G) = P + 1 (όπου P το πλήθος των απλών συνθηκών ελέγχου του κώδικα)

Άρα V(G) =1{απλές συνθήκες ελέγχου (if)}+1=2

Οπότε προσδιορισμός των βασικών μονοπατιών=

M1=1-2-3-5 Εφικτό=3

M2=1-2-4-5 Εφικτό=4

Συνεπώς το πρόγραμμά μπορεί να ελεγχθεί με 2 βασικά μονοπάτια.

περιπτώσεις ελέγχου=

Μονοπατι	OutputParameter=	Αποτέλεσμα=
M1	‘ok’	Αλλαγή σελίδας
M2	!=’ok’	‘Ανεπιτυχής Πληρωμή’