# statement coverage

## EcomSystem class:

    public User getUser(String UserName)

    {

    1-    for (int i = 0; i < Accounts.size(); i++) {

    2-        User user = Accounts.get(i);

    3-        if (user.getUserName().equalsIgnorecase(UserName)) {

    4-            userIndex = i;

    5-            return user;

            }

        }

    6-    userIndex = -1;

    7-    return null;

    }

**Test case 1:**

Accounts.size() 🡺 5

(user.getUserName().equalsIgnorecase(UserName)) 🡺 true

1->2->3->4->5

**Test case 2:**

Accounts.size() 🡺 0

1->6->7

    public boolean login(String UserName, String Password)

    {

    1-    if(Accounts.isEmpty())

    2-        return false;

    3-    User foundUser = getUser(UserName);

    4-    if (foundUser != null)

    5-        if(foundUser.getPassword().equals(Password))

    6-            return true;

    7-    return false;

    }

**Test case 1:**

Accounts.isEmpty() 🡺 true

foundUser = null

1->2

**Test case 2:**

Accounts.isEmpty() 🡺 false

foundUser != null

foundUser.getPassword().equals(Password)) 🡺 true

1->3->4->5->6

**Test case 3:**

Accounts.isEmpty() 🡺 false

foundUser = null

1->3->4->7

    public boolean signup(String FN, String LN, String UserName, String Password)

    {

    1-  User foundUser = getUser(UserName);

    2-    if (foundUser == null)

        {

    3-        userIndex = Accounts.size();

    4-        CreateAccount(FN, LN, UserName, Password);

    5-        return true;

        }

    6-    return false;

    }

**Test case 1:**

foundUser = null

1->2->3->4->5

**Test case 2:**

foundUser != null

1->2->6

    public User getCurrentUser(){

        1- if(userIndex >= 0 && userIndex < Accounts.size())

        2-    return Accounts.get(userIndex);

else

        3-     return null;

    }

**Test case 1:**

userIndex = 10

Account.size() 🡺 12

1->2

**Test case 2:**

userIndex = -10

Account.size() 🡺 12

1->3

## User class:

    public boolean Visa\_Exist(String num)

    {

        1- for (Visa v : Visas)

        2-    if(num.equals(v.getVisaNumber()))

        3-        return true;

        4- return false;

    }

**Test case 1:**

Visas != null

num.equals(v.getVisaNumber())) 🡺 true

1->2->3

**Test case 2:**

Visas = null

1->4

## Visa class:

    public boolean validvisa(int year, int month)

    {

        1- Calendar cal = Calendar.getInstance();

        2- int currentYear = cal.get(Calendar.YEAR)% 100;

        3- int currentMonth = cal.get(Calendar.MONTH) + 1;

        4- if (year < currentYear || year > currentYear + 5)

        5-    return false;

        6- else if (year == currentYear && month < currentMonth)

        7-    return false;

        8- return true;

    }

**Test case 1:**

year = 2020

month = 12

1->2->3->4->5

**Test case 2:**

year = 2024

month = 01

1->2->3->4->6->7

**Test case 3:**

year = 2025

month = 07

1->2->3->4->6->8

    public boolean CVV\_check(int cvv){

        1- if(this.cvv == cvv)

        2-    return true;

        3- return false;

    }

**Test case 1:**

this.cvv = 123

cvv = 456

1->3

**Test case 2:**

this.cvv = 555

cvv = 555

1->2

## Order class:

    public void scheduleStatusUpdate(LocalTime endTime) {

        1- int unitTime = 15;

        2- long TimeDiff = getTimeDifferenceInSeconds(endTime);

        3- if(TimeDiff < unitTime)

        4-    status = "Order\_Placed";

        5- else if(TimeDiff < 2\*unitTime)

        6-    status = "Packed";

        7- else if(TimeDiff < 3\*unitTime)

        8-     status = "Dispatch";

        else

        9-     status = "Delivered";

    }

**Test case 1:**

TimeDiff = 10

1->2->3->4

**Test case 2:**

TimeDiff = 20

1->2->3->5->6

**Test case 3:**

TimeDiff = 40

1->2->3->5->7->8

**Test case 4:**

TimeDiff = 50

1->2->3->5->7->9

    public double calc\_totalPrice()

    {

        1- double totPrice = cart.getTotalPrice();

        2- if(totPrice < 1000)

        3-     shippingPrice = 100;

        else

        4-     shippingPrice = 0;

        5- return totPrice + shippingPrice;

    }

**Test case 1:**

totPrice = 900

1->2->3->5

**Test case 2:**

totPrice = 1500

1->2->4->5

## Cart class:

public void addToCart(Item item, int amount) {  
 1- if(amount > item.getStock())  
 2- return;  
 3- int currentAmount = itemsAmounts.getOrDefault(item, 0);  
 4- itemsAmounts.put(item, currentAmount + amount);  
 5- setTotalPrice();  
}

**Test case 1:**

amount = 50

item.getStock()🡺20

1->2

**Test case 2:**

amount = 10

item.getStock()🡺20

1->3->4->5

public int increaseAmount(Item item) {   
 1- int currentAmount = itemsAmounts.get(item);  
 2- int newAmount = currentAmount + 1;  
 3- if (newAmount > item.getStock())  
 4- return currentAmount;  
 5- itemsAmounts.put(item, newAmount);  
 6- setTotalPrice();  
 7- return newAmount;  
}

**Test case 1:**

newAmount = 50

item.getStock()🡺20

1->2->3->4

**Test case 2:**

newAmount = 10

item.getStock()🡺20

1->2->3->5->6->7

public int decreaseAmount(Item item) {  
 1- int currentAmount = itemsAmounts.get(item);  
 2- int newAmount = currentAmount - 1;  
 3- if (newAmount == 0)  
 4- removeFromCart(item);  
 else  
 5- itemsAmounts.put(item, newAmount);  
  
 6- setTotalPrice();  
 7- return newAmount;  
}

**Test case 1:**

currentAmount = 1

1->2->3->4->6->7

**Test case 2:**

currentAmount = 10

1->2->3->5->6->7

## Catalog class

public static Item getItem(String bookName) {  
 1- for (Item item : *allitems*)  
 2- if (item.getName() == bookName)  
 3- return item;  
 4- return null;  
}

**Test case 1:**

item.getName()=> “Jane Eyre”

bookname= “Jane Eyre”

allitems not equal NULL

1->2->3

**Test case 2:**

item.getName()=> “War and Peace”

bookname= “Jane Eyre”

allitems not equal NULL

1->2->4

# Branch coverage

## EcomSystem class:

    public User getUser(String UserName)

    {

    1-    for (int i = 0; i < Accounts.size(); i++) {

    2-        User user = Accounts.get(i);

    3-        if (user.getUserName().equalsIgnorecase(UserName)) {

    4-            userIndex = i;

    5-            return user;

            }

        }

    6-    userIndex = -1;

    7-    return null;

    }

**Test case 1:**

Accounts.size() 🡺 5

(user.getUserName().equalsIgnorecase(UserName)) 🡺 true

1->2->3->4->5

**Test case 2:**

Accounts.size() 🡺 5

(user.getUserName().equalsIgnorecase(UserName)) 🡺 false

1->2->3->6->7

    public boolean login(String UserName, String Password)

    {

    1-    if(Accounts.isEmpty())

    2-        return false;

    3-    User foundUser = getUser(UserName);

    4-    if (foundUser != null)

    5-        if(foundUser.getPassword().equals(Password))

    6-            return true;

    7-    return false;

    }

**Test case 1:**

Accounts.isEmpty() 🡺 true

foundUser = null

1->2

**Test case 2:**

Accounts.isEmpty() 🡺 false

foundUser != null

foundUser.getPassword().equals(Password)) 🡺 true

1->3->4->5->6

**Test case 3:**

Accounts.isEmpty() 🡺 false

foundUser != null

foundUser.getPassword().equals(Password)) 🡺 false

1->3->4->5->7

**Test case 4:**

Accounts.isEmpty() 🡺 false

foundUser = null

1->3->4->7

    public boolean signup(String FN, String LN, String UserName, String Password)

    {

    1-  User foundUser = getUser(UserName);

    2-    if (foundUser == null)

        {

    3-        userIndex = Accounts.size();

    4-        CreateAccount(FN, LN, UserName, Password);

    5-        return true;

        }

    6-    return false;

    }

**Test case 1:**

foundUser = null

1->2->3->4->5

**Test case 2:**

foundUser != null

1->2->6

    public User getCurrentUser(){

        1- if(userIndex >= 0 && userIndex < Accounts.size())

        2-    return Accounts.get(userIndex);

else

        3-     return null;

    }

**Test case 1:**

userIndex = 10

Account.size() 🡺 12

1->2

**Test case 2:**

userIndex = -10

Account.size() 🡺 12

1->3

## User class:

    public boolean Visa\_Exist(String num)

    {

        1- for (Visa v : Visas)

        2-    if(num.equals(v.getVisaNumber()))

        3-        return true;

        4- return false;

    }

**Test case 1:**

Visas != null

num.equals(v.getVisaNumber())) 🡺 true

1->2->3

**Test case 2:**

Visas != null

num.equals(v.getVisaNumber())) 🡺 false

1->2->4

## Visa class:

    public boolean validvisa(int year, int month)

    {

        1- Calendar cal = Calendar.getInstance();

        2- int currentYear = cal.get(Calendar.YEAR)% 100;

        3- int currentMonth = cal.get(Calendar.MONTH) + 1;

        4- if (year < currentYear || year > currentYear + 5)

        5-    return false;

        6- else if (year == currentYear && month < currentMonth)

        7-    return false;

        8- return true;

    }

**Test case 1:**

year = 2020

month = 12

1->2->3->4->5

**Test case 2:**

year = 2024

month = 01

1->2->3->4->6->7

**Test case 3:**

year = 2025

month = 07

1->2->3->4->6->8

    public boolean CVV\_check(int cvv){

        1- if(this.cvv == cvv)

        2-    return true;

        3- return false;

    }

**Test case 1:**

this.cvv = 123

cvv = 456

1->3

**Test case 2:**

this.cvv = 555

cvv = 555

1->2

## Order class:

    public void scheduleStatusUpdate(LocalTime endTime) {

        1- int unitTime = 15;

        2- long TimeDiff = getTimeDifferenceInSeconds(endTime);

        3- if(TimeDiff < unitTime)

        4-    status = "Order\_Placed";

        5- else if(TimeDiff < 2\*unitTime)

        6-    status = "Packed";

        7- else if(TimeDiff < 3\*unitTime)

        8-     status = "Dispatch";

        else

        9-     status = "Delivered";

    }

**Test case 1:**

TimeDiff = 10

1->2->3->4

**Test case 2:**

TimeDiff = 20

1->2->3->5->6

**Test case 3:**

TimeDiff = 40

1->2->3->5->7->8

**Test case 4:**

TimeDiff = 50

1->2->3->5->7->9

    public double calc\_totalPrice()

    {

        1- double totPrice = cart.getTotalPrice();

        2- if(totPrice < 1000)

        3-     shippingPrice = 100;

        else

        4-     shippingPrice = 0;

        5- return totPrice + shippingPrice;

    }

**Test case 1:**

totPrice = 900

1->2->3->5

**Test case 2:**

totPrice = 1500

1->2->4->5

## Cart class:

public void addToCart(Item item, int amount) {  
 1- if(amount > item.getStock())  
 2- return;  
 3- int currentAmount = itemsAmounts.getOrDefault(item, 0);  
 4- itemsAmounts.put(item, currentAmount + amount);  
 5- setTotalPrice();  
}

**Test case 1:**

amount = 50

item.getStock()🡺20

1->2

**Test case 2:**

amount = 10

item.getStock()🡺20

1->3->4->5

public int increaseAmount(Item item) {   
 1- int currentAmount = itemsAmounts.get(item);  
 2- int newAmount = currentAmount + 1;  
 3- if (newAmount > item.getStock())  
 4- return currentAmount;  
 5- itemsAmounts.put(item, newAmount);  
 6- setTotalPrice();  
 7- return newAmount;  
}

**Test case 1:**

newAmount = 50

item.getStock()🡺20

1->2->3->4

**Test case 2:**

newAmount = 10

item.getStock()🡺20

1->2->3->5->6->7

public int decreaseAmount(Item item) {  
 1- int currentAmount = itemsAmounts.get(item);  
 2- int newAmount = currentAmount - 1;  
 3- if (newAmount == 0)  
 4- removeFromCart(item);  
 else  
 5- itemsAmounts.put(item, newAmount);  
  
 6- setTotalPrice();  
 7- return newAmount;  
}

**Test case 1:**

currentAmount = 1

1->2->3->4->6->7

**Test case 2:**

currentAmount = 10

1->2->3->5->6->7

## Catalog class

public static Item getItem(String bookName) {  
 1- for (Item item : *allitems*)  
 2- if (item.getName() == bookName)  
 3- return item;  
 4- return null;  
}

**Test case 1:**

item.getName()=> “Jane Eyre”

bookname= “Jane Eyre”

allitems not equal NULL

1->2->3

**Test case 2:**

item.getName()=> “War and Peace”

bookname= “Jane Eyre”

allitems not equal NULL

1->2->4

# Condition coverage

## EcomSystem class:

    public User getCurrentUser(){

        1- if(userIndex >= 0 && userIndex < Accounts.size())

        2-    return Accounts.get(userIndex);

else

        3-     return null;

    }

**Test case 1:**

userIndex = 10

Account.size() 🡺 12

1->2

**Test case 2:**

userIndex = -10

Account.size() 🡺 12

1->3

**Test case 3:**

userIndex = 20

Account.size() 🡺 12

1->3

## Visa class:

    public boolean validvisa(int year, int month)

    {

        1- Calendar cal = Calendar.getInstance();

        2- int currentYear = cal.get(Calendar.YEAR)% 100;

        3- int currentMonth = cal.get(Calendar.MONTH) + 1;

        4- if (year < currentYear || year > currentYear + 5)

        5-    return false;

        6- else if (year == currentYear && month < currentMonth)

        7-    return false;

        8- return true;

    }

**Test case 1:**

year = 2020

month = 12

1->2->3->4->5

**Test case 2:**

year = 2040

month = 01

1->2->3->4->5

**Test case 3:**

year = 2024

month = 02

1->2->3->4->6->7

**Test case 4:**

year = 2026

month = 08

1->2->3->4->6->8

# Multiple Condition coverage

## EcomSystem class:

    public User getCurrentUser(){

        1- if(userIndex >= 0 && userIndex < Accounts.size())

        2-    return Accounts.get(userIndex);

else

        3-     return null;

    }

**Test case 1:**

userIndex = 10

Account.size() 🡺 12

1->2

**Test case 2:**

userIndex = -10

Account.size() 🡺 12

1->3

**Test case 3:**

userIndex = 20

Account.size() 🡺 12

1->3

## Visa class:

    public boolean validvisa(int year, int month)

    {

        1- Calendar cal = Calendar.getInstance();

        2- int currentYear = cal.get(Calendar.YEAR)% 100;

        3- int currentMonth = cal.get(Calendar.MONTH) + 1;

        4- if (year < currentYear || year > currentYear + 5)

        5-    return false;

        6- else if (year == currentYear && month < currentMonth)

        7-    return false;

        8- return true;

    }

**Test case 1:**

year = 2020

month = 12

1->2->3->4->5

**Test case 2:**

year = 2040

month = 01

1->2->3->4->5

**Test case 3:**

year = 2024

month = 07

1->2->3->4->6->8

**Test case 4:**

year = 2024

month = 01

1->2->3->4->6->7

**Test case 5:**

year = 2026

month = 08

1->2->3->4->6->8

**Test case 6:**

year = 2026

month = 02

1->2->3->4->6->8

# MCDC

## EcomSystem class:

    public User getCurrentUser(){

        1- if(userIndex >= 0 && userIndex < Accounts.size())

        2-    return Accounts.get(userIndex);

else

        3-     return null;

    }

**Test case 1:**

userIndex = 10

Account.size() 🡺 12

1->2

**Test case 2:**

userIndex = -10

Account.size() 🡺 12

1->3

**Test case 3:**

userIndex = 20

Account.size() 🡺 12

1->3

## Visa class:

    public boolean validvisa(int year, int month)

    {

        1- Calendar cal = Calendar.getInstance();

        2- int currentYear = cal.get(Calendar.YEAR)% 100;

        3- int currentMonth = cal.get(Calendar.MONTH) + 1;

        4- if (year < currentYear || year > currentYear + 5)

        5-    return false;

        6- else if (year == currentYear && month < currentMonth)

        7-    return false;

        8- return true;

    }

**Test case 1:**

year = 2020

month = 12

1->2->3->4->5

**Test case 2:**

year = 2040

month = 01

1->2->3->4->5

**Test case 3:**

year = 2024

month = 07

1->2->3->4->6->8

**Test case 4:**

year = 2024

month = 01

1->2->3->4->6->7

**Test case 5:**

year = 2026

month = 02

1->2->3->4->6->8