Black Box test cases:

- Story #1: As an employee, I want to be able to place an order for a specific book so it can be sold to a particular student.

  a. Valid/Invalid Input/output
    - **Valid Input:** Student Number (Numeric- 9 digits), ISBN-10 (Numeric-10 digits), Employee Number (Numeric-5 digits)
    - **Valid Output:** Alphanumeric (Eg: “order placed, Order# 56690”, “unable to place order”, “employee/student num not found” etc)
    - **Invalid Input**: Characters a-z,A-Z, Special characters, Student Number (Numeric - 9 < digits < 9), ISBN-10 (Numeric- 10 < digits < 10), Employee Number (Numeric- 5 < digits < 5)
    - **Invalid Output:** program doesn’t proceed, Program crashing

  b. Equivalent Classes
    - EC1 - Student Number [100000001, 999999999]
    - EC2 - ISBN-10 [1000000001, 9999999999]
    - EC3 - Employee Number [10001, 99999]

  c. Boundary Value Analysis

Input Type	InValid	Valid	InValid
Student #	100000000	167934082	1000000000
ISBN-10	1000000000	4672895719	10000000000
Employee #	10000	15561	100000

  d. Steps for testing
    - Precondition: System (order placement software) is open/logged on to the main page
    - Input: Book ISBN-10, Student Number, Employee Number
    - Expected Output: Order Number
    - Postcondition: System will go back to its original state – main page.

Test Cases:

TC1	TC2	TC3	TC4
Input 1: 167934082	Input 1: 5	Input 1: 5	Input 1: 167937080
Input 2: 4672895719	Input 2: 2	Input 2: 467985719	Input 2: 4672235783
Input 3: 15561	Input 3: 5A7	Input 3: 15781	Input 3: 12181
Output: “Order placed, Order# 56690”	Output: INVALID	Output: INVALID	Output: “Student Number Not Found”

- Story #2: As an employee, I want to be able to reserve an in-stock book for a student so they can come and purchase it later.
- Valid/Invalid Input/output
  - **Valid Input:** Student Number (Numeric- 9 digits), ISBN-10 (Numeric-10 digits), Employee Number (Numeric-5 digits), E-mail (Alphanumeric – username@uwindsor.ca)
  - **Valid Output:** Alphanumeric (Eg: "book reserved, reservation# 56690", "unable to reserve item", "employee/student num not found" etc), email reservation sent to username@uwindsor.ca.
  - **Invalid Input**: Special characters other than @, Student Number (Numeric - 9 < digits < 9), ISBN-10 (Numeric- 10 < digits < 10), Employee Number (Numeric- 5 < digits < 5), emails outside of Uwindsor (@uwindsor.ca)
  - **Invalid Output:** program doesn't proceed, Program crashing

e. Equivalent Classes
  - EC1 - Student Number [100000001, 999999999]
  - EC2 - ISBN-10 [1000000001, 9999999999]
  - EC3 - Employee Number [10001, 99999]

f. Boundary Value Analysis

Input Type	InValid	Valid	InValid
Student #	100000000	167934082	1000000000
ISBN-10	1000000000	4672895719	10000000000
Employee #	10000	15561	100000

g. Steps for testing
  - Precondition: System (order placement software) is open/logged on to the main page
  - Input: Book ISBN-10, Student Number, Employee Number, e-mail address
  - Expected Output: Reservation number, e-mail sent!
  - Postcondition: System will go back to its original state – main page.

Test Cases:

TC1	TC2	TC3	TC4	TC5
Input 1: 167934082	Input 1: 5	Input 1: 5	Input 1: 167937082	Input 1: 167937154
Input 2: 4672895719	Input 2: 2	Input 2: 467985719	Input 2: 4672235719	Input 2: 4672235345
Input 3: 15561	Input 3: 5A7	Input 3: 15781	Input 3: 12181	Input 3: 16475
Input 4: abc12@uwindsor.ca	Input 4: abc12@gmail.com	Input 4: abc12@uwindsor.ca	Input 4: abc12@uwindsor.ca	Input 4: abcd@gmail.com
Output: "reservation made, reservation# 56690"	Output: INVALID	Output: INVALID	Output: "Student Num Not Found"	Output: INVALID

- Story #3: As an employee, I want to be able to reserve an out-of-stock book for a student so they can come and purchase it later.
- Valid/Invalid Input/output
  - **Valid Input:** Student Number (Numeric- 9 digits), ISBN-10 (Numeric-10 digits), Employee Number (Numeric-5 digits), E-mail (Alphanumeric – username@uwindsor.ca)
  - **Valid Output:** Alphanumeric (Eg: "book reserved, reservation# 56690, expected pickup date: 21-10-2022", "unable to reserve item", "employee/student num not found" etc), email reservation sent to username@uwindsor.ca.
  - **Invalid Input**: Special characters other than @, Student Number (Numeric - 9 < digits < 9), ISBN-10 (Numeric- 10 < digits < 10), Employee Number (Numeric- 5 < digits < 5), emails outside of Uwindsor (@uwindsor.ca)
  - **Invalid Output:** program doesn't proceed, Program crashing

h. Equivalent Classes
  - EC1 - Student Number [100000001, 999999999]
  - EC2 - ISBN-10 [1000000001, 9999999999]
  - EC3 - Employee Number [10001, 99999]

i. Boundary Value Analysis

Input Type	InValid	Valid	InValid
Student #	100000000	167934082	1000000000
ISBN-10	1000000000	4672895719	10000000000
Employee #	10000	15561	100000

j. Steps for testing
  - Precondition: System (order placement software) is open/logged on to the main page
  - Input: Book ISBN-10, Student Number, Employee Number, e-mail address
  - Expected Output: Reservation number, e-mail sent!
  - Postcondition: System will go back to its original state – main page.

Test Cases:

TC1	TC2	TC3	TC4	TC5
Input 1: 167934082	Input 1: 5	Input 1: 5	Input 1: 167937082	Input 1: 167937154
Input 2: 4672895719	Input 2: 2	Input 2: 467985719	Input 2: 4672235719	Input 2: 4672235345
Input 3: 15561	Input 3: 5A7	Input 3: 15781	Input 3: 12181	Input 3: 16475
Input 4: abc12@uwindsor.ca	Input 4: abc12@gmail.com	Input 4: abc12@uwindsor.ca	Input 4: abc12@uwindsor.ca	Input 4: abcd@gmail.com
Output: "Reservation made,	Output: INVALID	Output: INVALID	Output: "Student Num Not Found"	Output: INVALID

Reservation# 56690”				

- Story #4: As an employee in the bookstore, I want to sell a book to a student so they may purchase a book.
- Valid/Invalid Input/output
    - **Valid Input:** Student Number (Numeric- 9 digits), ISBN-10 (Numeric-10 digits), Employee Number (Numeric-5 digits), Student card code (Numeric-14 digits)
    - **Valid Output:** Alphanumeric (Eg: “order placed, receipt”, “unable to place order”, “employee/student num not found” etc)
    - **Invalid Input**: Characters a-z,A-Z, Special characters, Student Number (Numeric - 9 < digits < 9), ISBN-10 (Numeric- 10 < digits < 10), Employee Number (Numeric- 5 < digits < 5), Student card code (Numeric - 14 < digits < 14)
    - **Invalid Output:** program doesn’t proceed, Program crashing

k. Equivalent Classes
- EC1 - Student Number [100000001, 999999999]
- EC2 - ISBN-10 [1000000001, 9999999999]
- EC3 - Employee Number [10001, 99999]
- EC4 - Student card code [10000000000001, 99999999999999]

l. Boundary Value Analysis

Input Type	InValid	Valid	InValid
Student #	100000000	167934082	1000000000
ISBN-10	1000000000	4672895719	10000000000
Employee #	10000	15561	100000
Student card code	10000000000000	10034004500670	100000000000000

m. Steps for testing
- Precondition: System (order placement software) is open/logged on to the main page
- Input: Book ISBN-10, Student Number, Employee Number, Student Number Code
- Expected Output: Receipt
- Postcondition: System will go back to its original state – main page.

Test Cases:

TC1	TC2	TC3	TC4	TC5
Input 1: 167934082	Input 1: 5	Input 1: 5	Input 1: 167937082	Input 1: 167934082
Input 2: 4672895719	Input 2: 2	Input 2: 467985719	Input 2: 4672235719	Input 2: 4672895719

Input 3: 15561	Input 3: 5A7	Input 3: 15781	Input 3: 12181	Input 3: 15561
Input 4: 12345678910111	Input 4: 12345565677	Input 4: 12345678922611	Input 4: 12345678998761	Input 4: 12345565677
Output: “order placed, Order# 56690”	Output: INVALID	Output: INVALID	Output: “Student Num Not Found”	Output: “Card code Invalid”