SECJ1023
PROGRAMING TECHNIQUE II
SECTION 08
LECTURER: MS LIZAWATI BINTI MI YUSUF

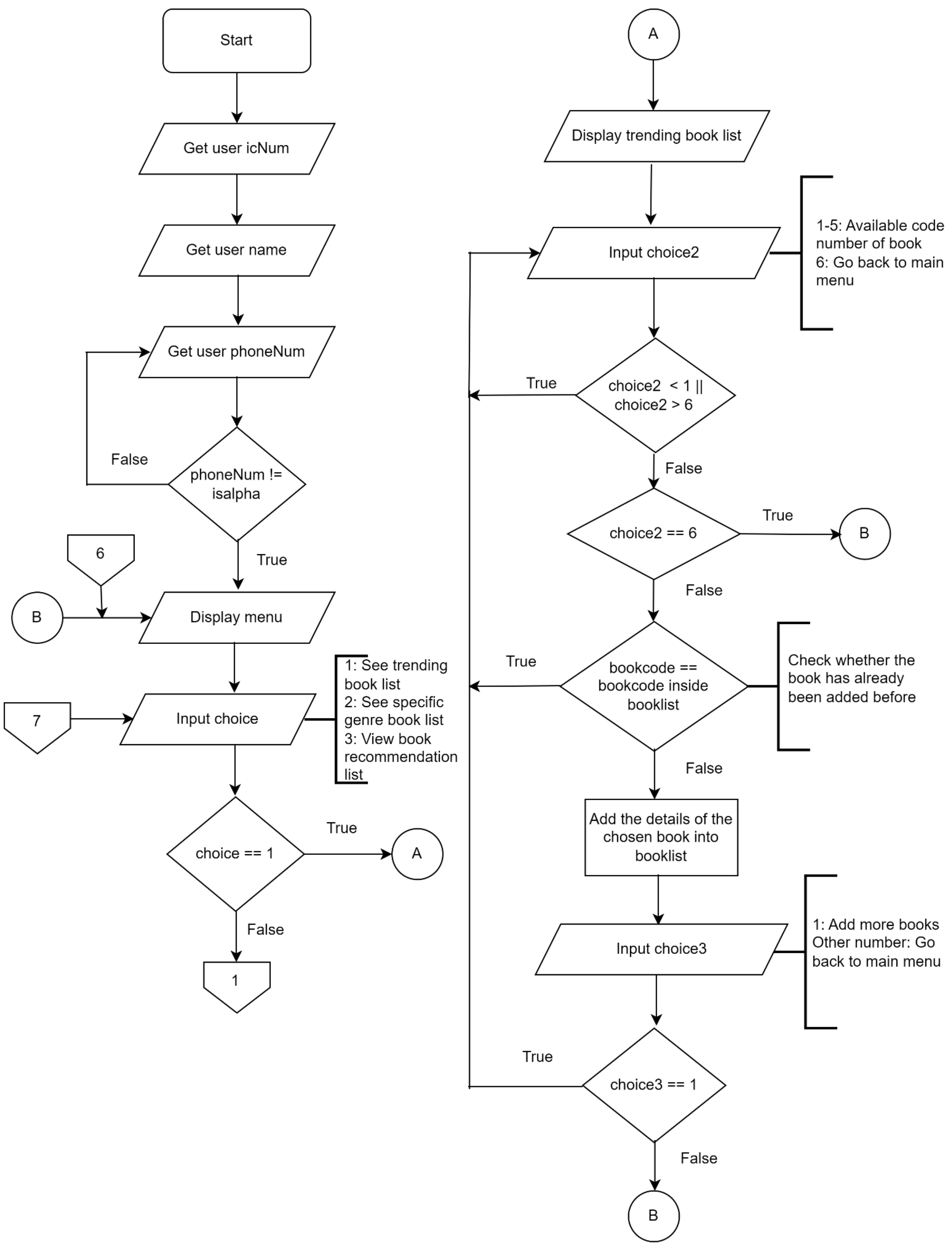
Group Project Deliverable 2: Problem Analysis and Design

Group

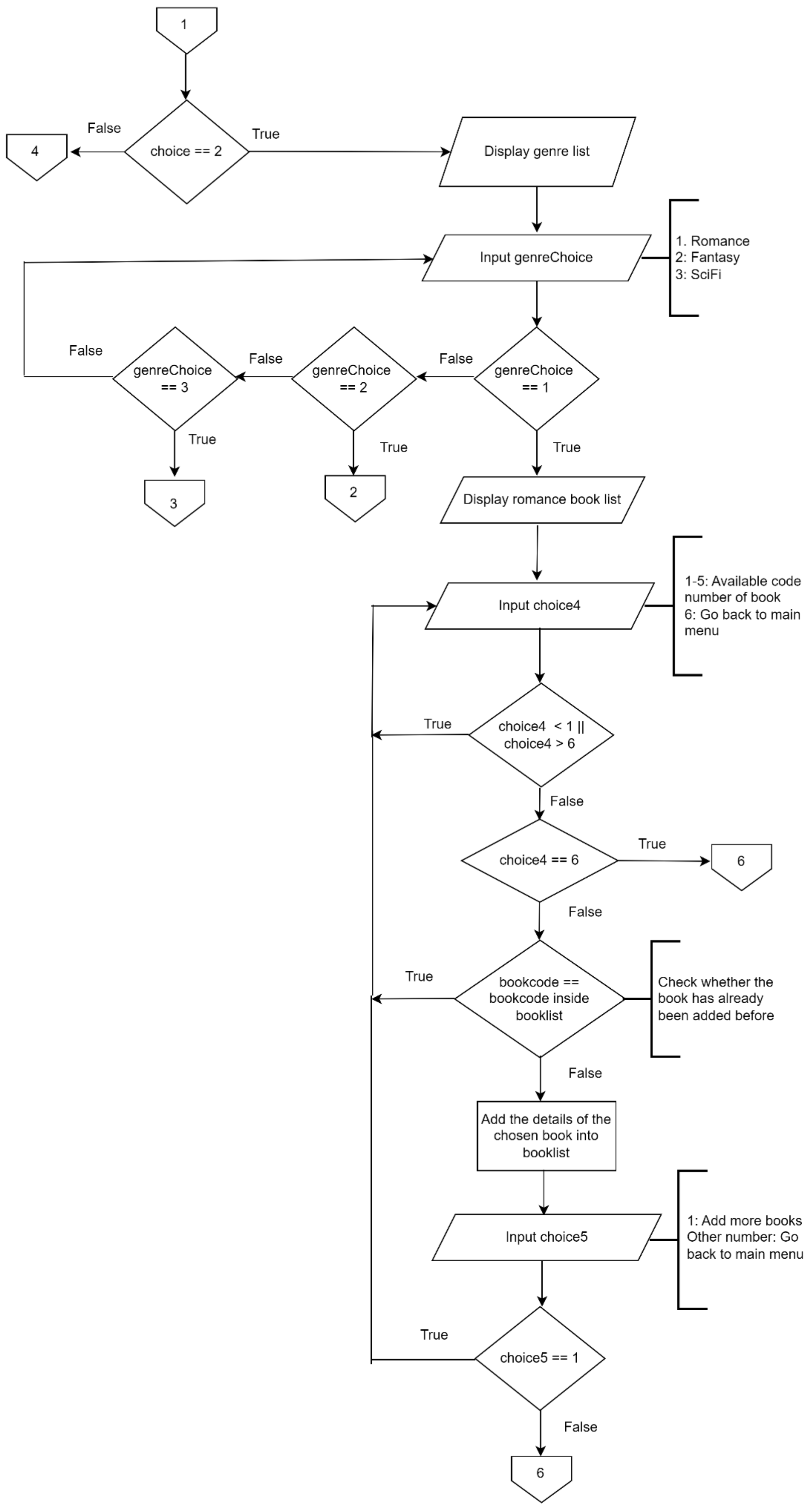
Name	Matric No
Chuah Hui Wen	A23CS0219
Lim Xin Rou	A23CS0240
Chen Wei Jay Nickolas	A23CS5028

TABLE OF CONTENT

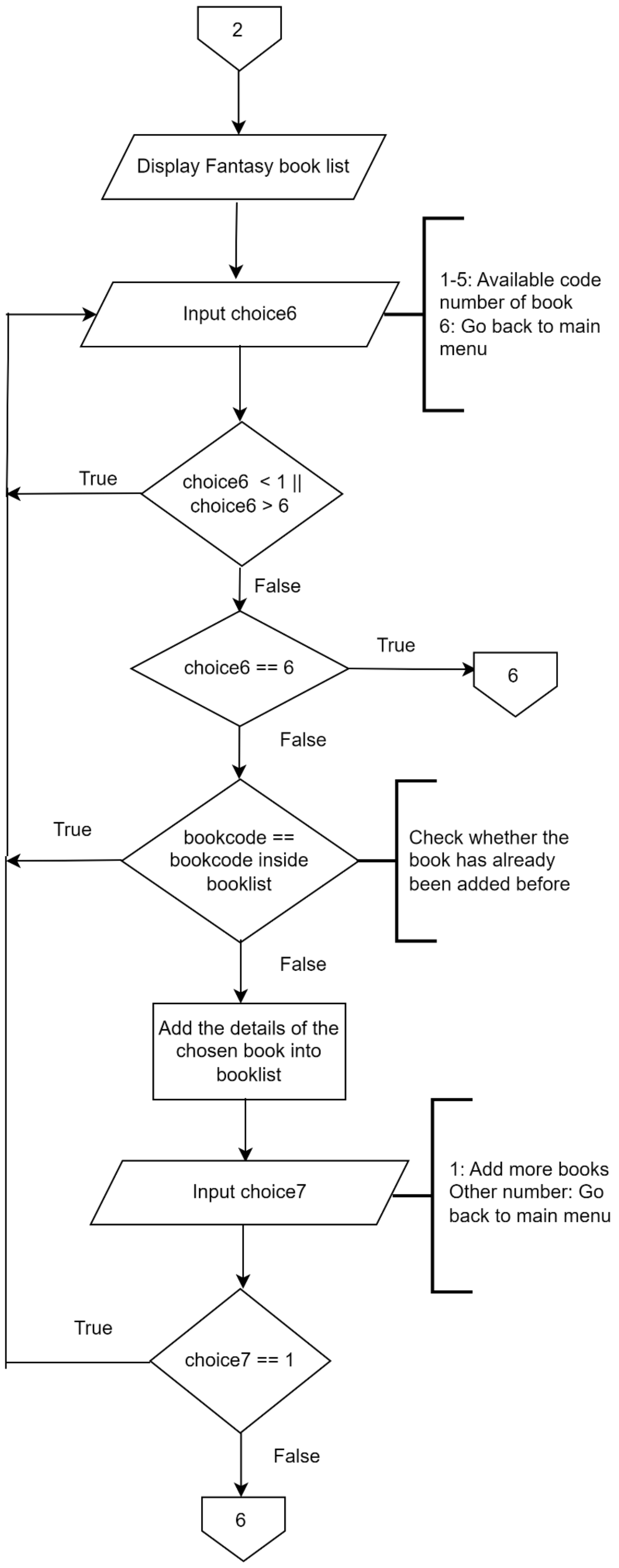
Section A: Flow Chart

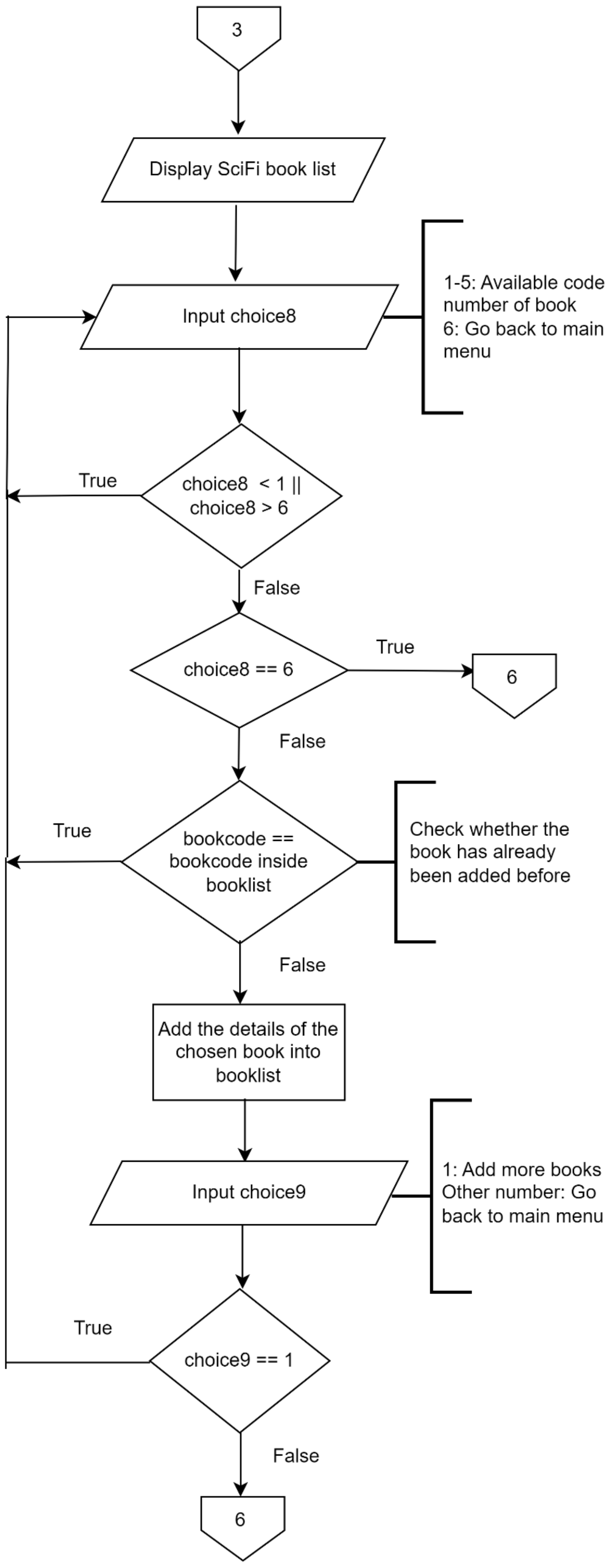


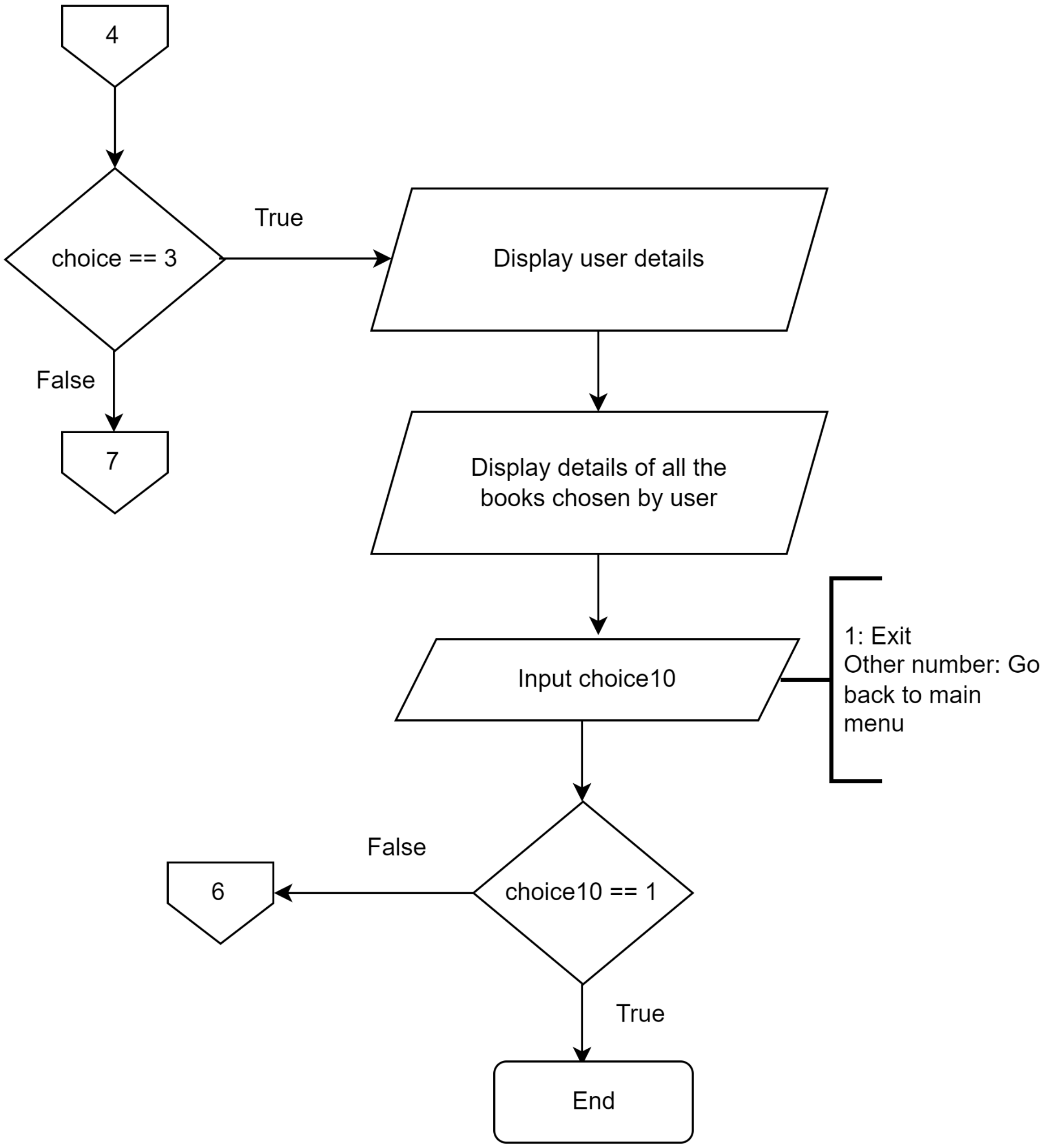
Start
Get user icNum
Get user name
Get user phoneNum
phoneNum != isalpha
False
True
6
B
Display menu
7
Input choice
1: See trending book list
2: See specific genre book list
3: View book recommendation list
choice == 1
True
A
False
1
A
Display trending book list
Input choice2
1-5: Available code number of book
6: Go back to main menu
choice2 < 1 || choice2 > 6
True
False
choice2 == 6
True
B
False
bookcode == bookcode inside booklist
True
Check whether the book has already been added before
False
Add the details of the chosen book into booklist
Input choice3
1: Add more books
Other number: Go back to main menu
choice3 == 1
True
False
B


1
choice == 2
False
True
4
Display genre list
Input genreChoice
1. Romance
2: Fantasy
3: SciFi
genreChoice == 1
False
genreChoice == 2
False
genreChoice == 3
False
True
True
True
3
2
Display romance book list
Input choice4
1-5: Available code number of book
6: Go back to main menu
choice4 < 1 || choice4 > 6
True
False
choice4 == 6
True
6
False
bookcode == bookcode inside booklist
True
Check whether the book has already been added before
False
Add the details of the chosen book into booklist
Input choice5
1: Add more books
Other number: Go back to main menu
choice5 == 1
True
False
6

1


2
Display Fantasy book list
Input choice6
1-5: Available code number of book
6: Go back to main menu
choice6 < 1 || choice6 > 6
True
False
choice6 == 6
True
6
False
bookcode == bookcode inside booklist
True
Check whether the book has already been added before
False
Add the details of the chosen book into booklist
Input choice7
1: Add more books
Other number: Go back to main menu
choice7 == 1
True
False
6


3
Display SciFi book list
Input choice8
1-5: Available code number of book
6: Go back to main menu
True
choice8 < 1 || choice8 > 6
False
choice8 == 6
True
6
False
True
bookcode == bookcode inside booklist
Check whether the book has already been added before
False
Add the details of the chosen book into booklist
Input choice9
1: Add more books
Other number: Go back to main menu
True
choice9 == 1
False
6



First of all, the user must enter their IC number, name and phone number to use the system. Then, the system presents the main menu with three options (1. See trending book list, 2. See specific genre book list, and 3. View book recommendation list). Selecting ‘1’ displays the trending book list which consists of 5 trending books, and the user can choose which book they are interested in by input number from 1 to 5. Besides, Selecting ‘2’ in the main menu will display the genre list (1. Romance, 2. Fantasy and 3. SciFi), and after picking a genre, the system will display a list of 5 books from the chosen genre. Users can also choose which book they are interested in by input number from 1 to 5. To return to the main menu, the user can enter ‘6’. When they choose a book, the system will check whether the book chosen has already been added into the recommendation book list before, if not the details of chosen book will be automatically added to the recommendation list. Users can then choose to add more books from the same list by entering ‘1’ or choose to return to the main menu by entering a different number. Selecting ‘3’ in the main menu will display the recommendation list that was created by the user, which includes user details and the details of all the books chosen by the user. After that, the user can choose to exit the system by entering number ‘1’, or choose to return to the main menu by entering any other number.

<u>Section B: Problem Analysis</u>

i)

<u>The objects involved in the project</u>

1. class Publisher
   - 5 publisher objects
2. class Books
   - Fantasy Class : 5 fantasy objects
   - Romance Class : 5 romance objects
   - SciFi Class : 5 scifi objects
3. class User
   - 1 user object
4. class Booklist
   - 1 booklist object

<u>The classes involved in the project</u>

1. class Publisher

   Attributes :
   - string publisherName
   - string country

   publisherName and country are private member variables of the class. They are accessible only within the class and its member functions.

   Methods:
   - Publisher() is a default constructor
   - Publisher(string n, string c) is a parameterized constructor that initializes publisherName and country with the provided values
   - ~Publisher() is a destructor
   - string getPublisherName() is an accessor that returns the value of publisherName
   - void setPublisherName(string n) is a mutator that set value for the publisherName
   - string getCountry() is an accessor that returns the value of country
   - void setCountry(string c) is a mutator that set value for the country

2. class Book

   Attributes :

   - string bookCode
   - string bookTitle
   - string genre
   - int yearPublish
   - Publisher* publisher

   All attributes are protected member variables of the class. They are accessible within the class and its derived classes. bookCode, bookTitle, genre, and yearPublish store information about the book, while publisher is a pointer to a Publisher object, indicating the publisher of the book.

   Methods :

   - Book() is a default constructor
   - Book(string bc, string bt, string g, int yp, Publisher* p) is a parameterized constructor that initializes bookCode, bookTitle, genre, yearPublish and publisher with the provided values
   - ~Book() is a destructor
   - string getBookCode() is an accessor that returns the value of bookCode
   - string getBookTitle() is an accessor that returns the value of bookTitle
   - string getGenre() is an accessor that returns the value of genre
   - int getYearPublish() is an accessor that returns the value of yearPublish
   - Publisher* getPublisher() is an accessor that returns a pointer that points to the book's publisher
   - void setBookCode(string bc) is a mutator that sets the value of bookCode
   - void setBookTitle(string bt) is a mutator that sets the value of bookTitle
   - void setGenre(string g) is a mutator that sets the value of genre
   - void setYearPublish(int yp) is a mutator that sets the value of yearPublish
   - void setPublisher(Publisher *p) is a mutator that sets the value of publisher

- virtual void display() is a virtual function that displays the information of the book. It is marked as virtual, indicating that it can be overridden by derived classes.

3. class Romance

   Attributes :

   - string mainCoupleName

   mainCoupleName is a private member variable of the Romance class. mainCoupleName is accessible only within the class and its member functions.

   Methods :

   - Romance() is a default constructor
   - Romance(string bc, string bt, int yp, Publisher* p, string mc) is a parameterized constructor that initializes the Romance object with the provided values
   - ~Romance() is a destructor
   - string getMainCoupleName() is an accessor that returns the value of mainCoupleName
   - void setMainCoupleName(string mc) is a mutator that sets the value of mainCoupleName
   - void display() is a function that overrides the display() function of the base class Book. It first calls the display() function of the Book class to display book information, and then prints the mainCoupleName.

4. class Fantasy

   Attributes :

   - string creatureType

   creatureType is a private member variable of the Fantasy class. creatureType is accessible only within the class and its member functions.

   Methods :

   - Fantasy() is a default constructor

- Fantasy(string bc, string bt, int yp, Publisher* p, string ct) is a parameterized constructor that initializes the Fantasy object with the provided values.
- ~Fantasy() is a destructor
- string getCreatureType() is a accessor that returns the value of creatureType
- void setCreatureType(string ct) is a mutator that sets the value of creatureType
- void display() is a function that overrides the display() function of the base class Book. It first calls the display() function of the Book class to display book information, and then prints the creatureType

5. class SciFi
   Attributes :
   - string scientificConcept

   scientificConcept is a private member variable of the SciFi class. scientificConcept is accessible only within the class and its member functions.

   Methods :
   - SciFi() is a default constructor
   - SciFi(string bc, string bt, int yp, Publisher* p, string sc) is a parameterized constructor that initializes the SciFi object with the provided values.
   - string getScientificConcept() is an accessor that returns the value of scientificConcept
   - void setScientificConcept(string ct) is a mutator that sets the value of scientificConcept
   - void display() is a function that overrides the display() function of the base class Book. It first calls the display() function of the Book class to display book information, and then prints the scientificConcept

6. class Booklist
   Attributes :
   - Book* books[100]
   - int count

   Both attributes are private member variables of the Booklist class. Both attributes are accessible only within the class and its member functions. books is an array of pointers to Book objects. The array can hold up to 100 Book pointers. count is an integer that tracks the number of Book objects currently in the list.

Methods:

- Booklist() is a constructor
- ~Booklist() is a destructor
- int getCount() is an accessor that returns the count of books
- Book* getBook(int index) is an accessor that returns a pointer that points to the book
- void addBook(Book* b) is a mutator that adds book objects to the booklist object
- void setCount(int c) is a mutator that set the value of count
- bool isBookInList(Book* b) is a method that checks whether the book that the user wants to add is in the book list already
- void display() is a method that displays the the list of books

7. class User
   Attributes :
   - string name
   - string phoneNum
   - string icNum
   - Booklist booklist

   All attributes are private member variables of the User class. All attributes are accessible only within the class and its member functions.

   Methods :
   - User() is a default constructor
   - User(string ic, string n, string pn) is a parameterized constructor that initializes a User object with the given IC number, name, and phone number.
   - ~User() is a destructor
   - string getName() is an accessor that return the name of the user

- string getPhoneNum() is an accessor that returns the phoneNum of the user
- string getIcNum() is an accessor that return the icNum of the user
- void setName(string n) is a mutator that sets the name of the user
- void setPhoneNum(string p) is a mutator that sets the phoneNum of the user
- void setIcNum(string ic) is a mutator that sets the icNum of the user
- void displayLogin() is a method to prompt the user to enter their IC number, name, and phone number
- void displayBooklist() is a method to display the user details and user's list of books.
- void addBookToBooklist(Book* book) is a method that adds the user’s interested book to their booklist

ii) class relationships

Association Relationships

1. Book and Publisher

Book and Publisher will have an aggregation relationship. The Book class holds a pointer to a Publisher object. A Book has a Publisher. The Publisher object is created outside the Book and is passed to it, showing a "has-a" relationship. Publisher can exist independently. If the Book object is destroyed, the Publisher object will not be destroyed.

2. Booklist and Book

Booklist and Book will have an aggregation relationship. The Booklist class aggregates Book pointers, showing a "has-a" relationship, meaning booklist has a collection of books. Books can exist independently. If the Booklist object is destroyed, the Book object will not be destroyed.

3. User and Booklist

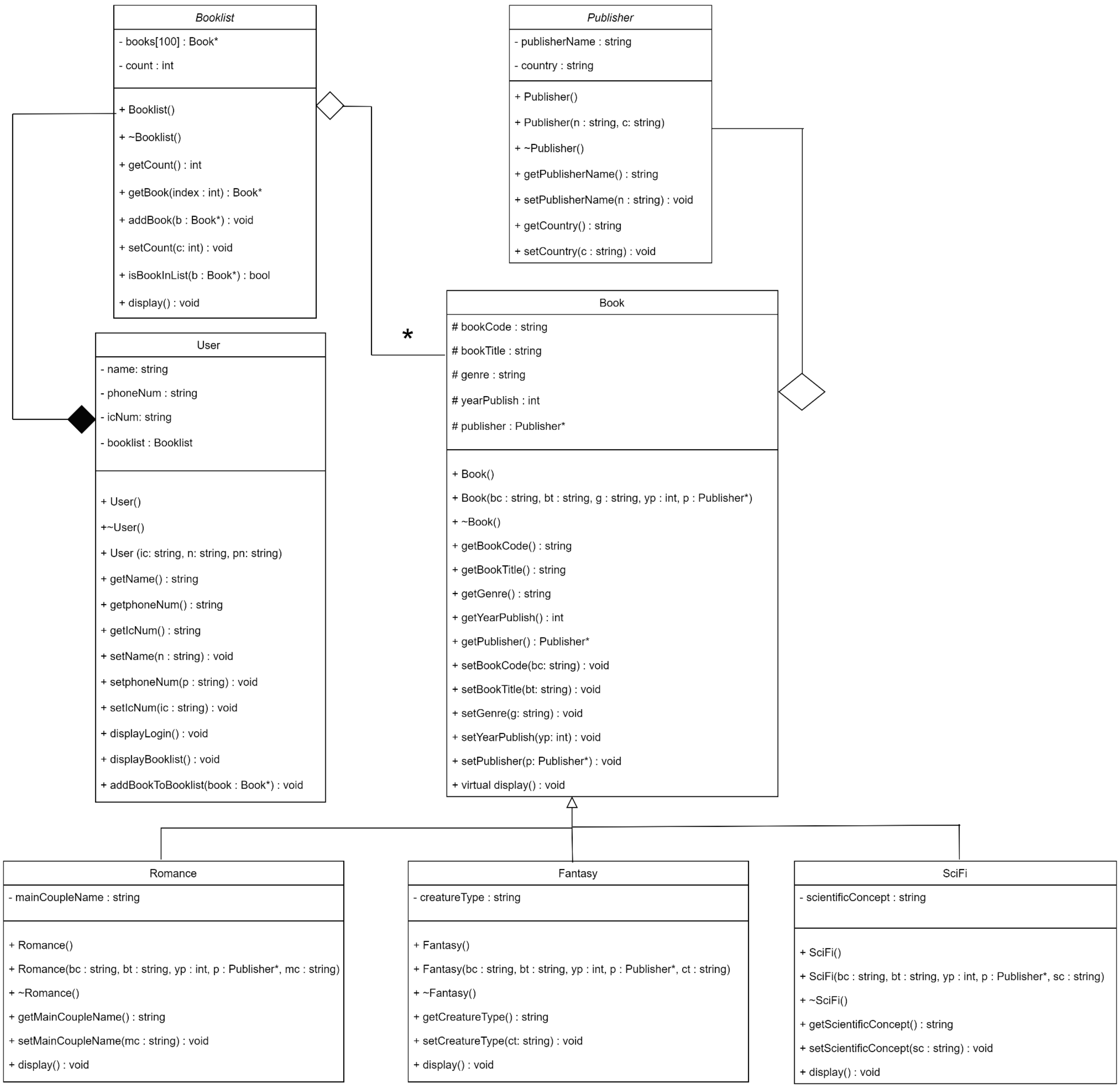
User and Booklist will have a composition relationship. A User consists of a Booklist object, indicating that the Booklist cannot exist independently if no User. Booklist strongly depends on User. If the User object is destroyed, then the Booklist object is also destroyed.

Inheritance Relationships

1. Book, Romance, Fantasy, SciFi

Inheritance among Book, Romance, Fantasy, and SciFi is suitable for representing an is-a relationship, where each specific genre is a type of book. Inheritance is used because it allows these specific genres to reuse common book attributes and methods defined in the Book class while also specifying their genre during instantiation.

Section C: Class Diagram


Booklist
- books[100] : Book*
- count : int
+ Booklist()
+ ~Booklist()
+ getCount() : int
+ getBook(index : int) : Book*
+ addBook(b : Book*) : void
+ setCount(c: int) : void
+ isBookInList(b : Book*) : bool
+ display() : void
Publisher
- publisherName : string
- country : string
+ Publisher()
+ Publisher(n : string, c: string)
+ ~Publisher()
+ getPublisherName() : string
+ setPublisherName(n : string) : void
+ getCountry() : string
+ setCountry(c : string) : void
*
User
- name: string
- phoneNum : string
- icNum: string
- booklist : Booklist
+ User()
+~User()
+ User (ic: string, n: string, pn: string)
+ getName() : string
+ getphoneNum() : string
+ getIcNum() : string
+ setName(n : string) : void
+ setphoneNum(p : string) : void
+ setIcNum(ic : string) : void
+ displayLogin() : void
+ displayBooklist() : void
+ addBookToBooklist(book : Book*) : void
Book
# bookCode : string
# bookTitle : string
# genre : string
# yearPublish : int
# publisher : Publisher*
+ Book()
+ Book(bc : string, bt : string, g : string, yp : int, p : Publisher*)
+ ~Book()
+ getBookCode() : string
+ getBookTitle() : string
+ getGenre() : string
+ getYearPublish() : int
+ getPublisher() : Publisher*
+ setBookCode(bc: string) : void
+ setBookTitle(bt: string) : void
+ setGenre(g: string) : void
+ setYearPublish(yp: int) : void
+ setPublisher(p: Publisher*) : void
+ virtual display() : void
Romance
- mainCoupleName : string
+ Romance()
+ Romance(bc : string, bt : string, yp : int, p : Publisher*, mc : string)
+ ~Romance()
+ getMainCoupleName() : string
+ setMainCoupleName(mc : string) : void
+ display() : void
Fantasy
- creatureType : string
+ Fantasy()
+ Fantasy(bc : string, bt : string, yp : int, p : Publisher*, ct : string)
+ ~Fantasy()
+ getCreatureType() : string
+ setCreatureType(ct: string) : void
+ display() : void
SciFi
- scientificConcept : string
+ SciFi()
+ SciFi(bc : string, bt : string, yp : int, p : Publisher*, sc : string)
+ ~SciFi()
+ getScientificConcept() : string
+ setScientificConcept(sc : string) : void
+ display() : void