UTM
UNIVERSITI TEKNOLOGI MALAYSIA

FACULTY OF COMPUTING
UTM Johor Bahru

SECJ1023-08

SEMESTER 2 2023/2024

PROGRAMMING TECHNIQUE II

GROUP PROJECT DELIVERABLE 2

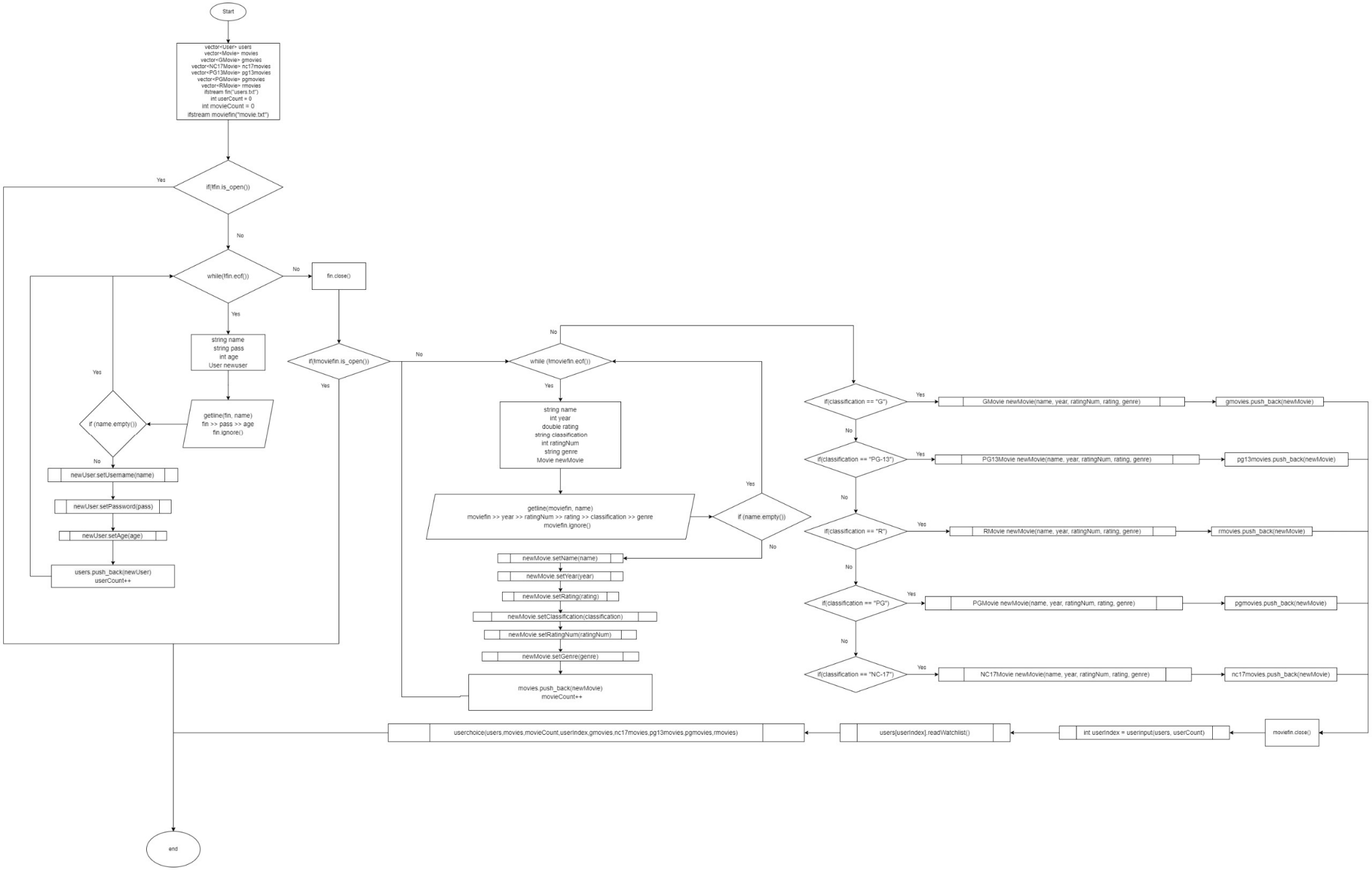
(Problem Analysis and Design)

CineMatch

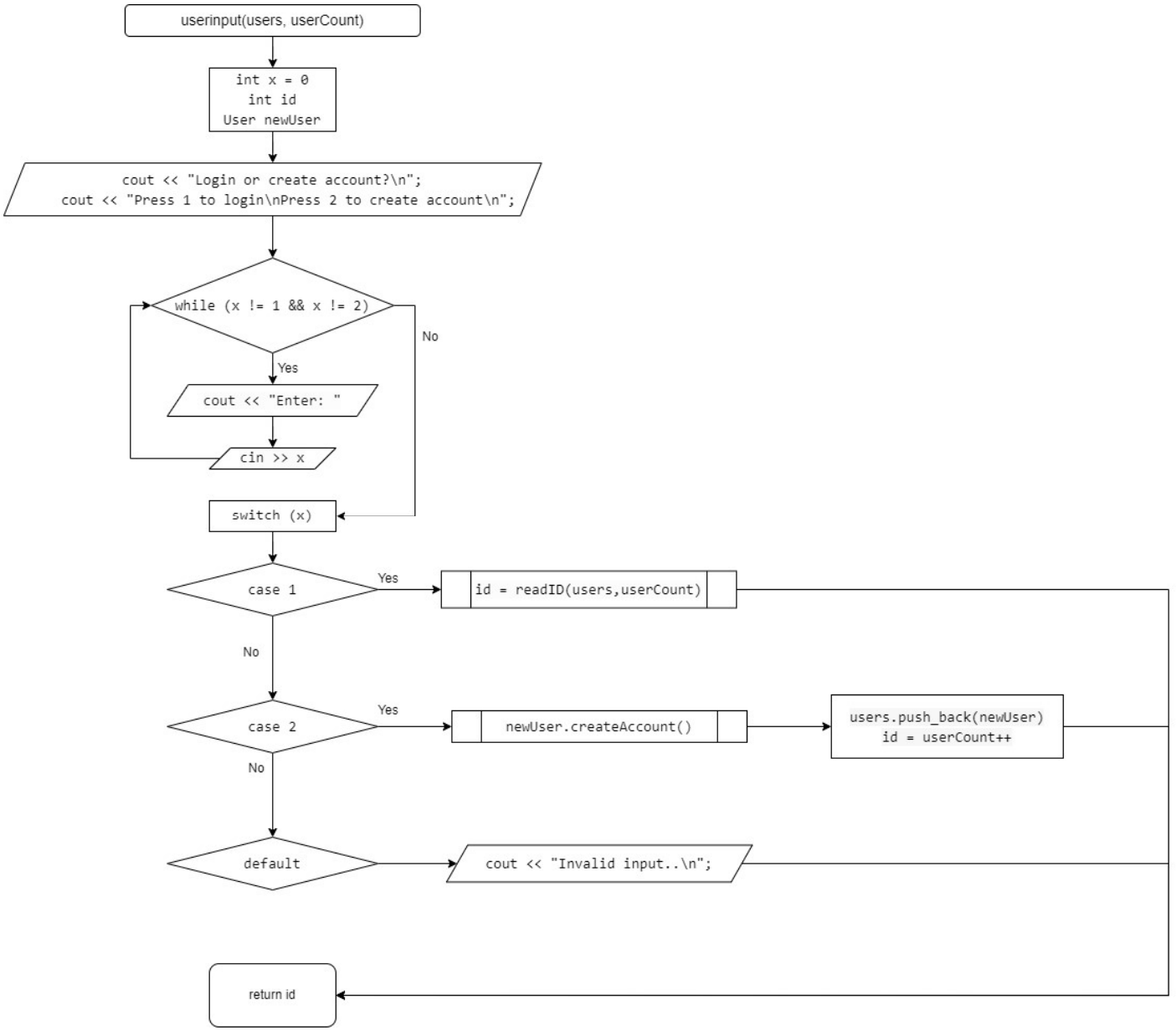
CHEW ZHUO HENG	A23CS0064
GOH CHANG ZHE	A23CS0225
CHEW CHUAN KAI	A23CS0062

Section A:Flow chart

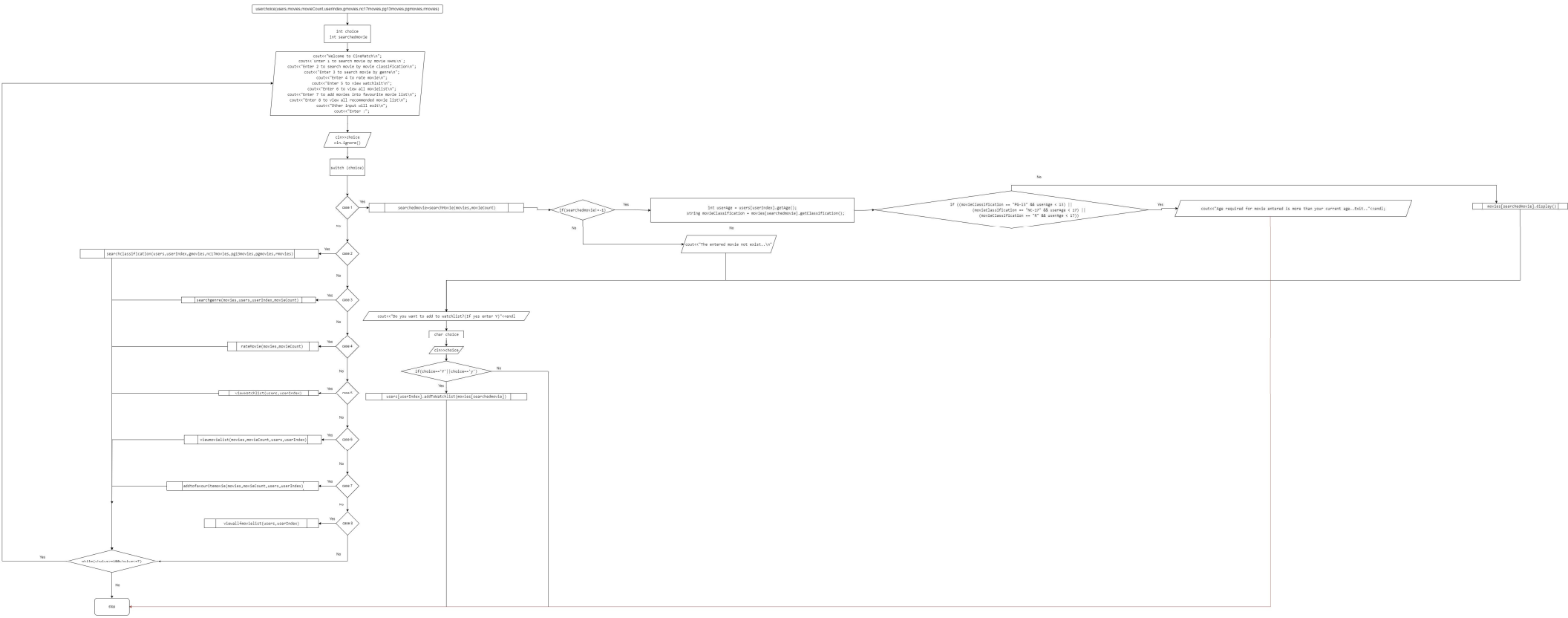
Main flowchart



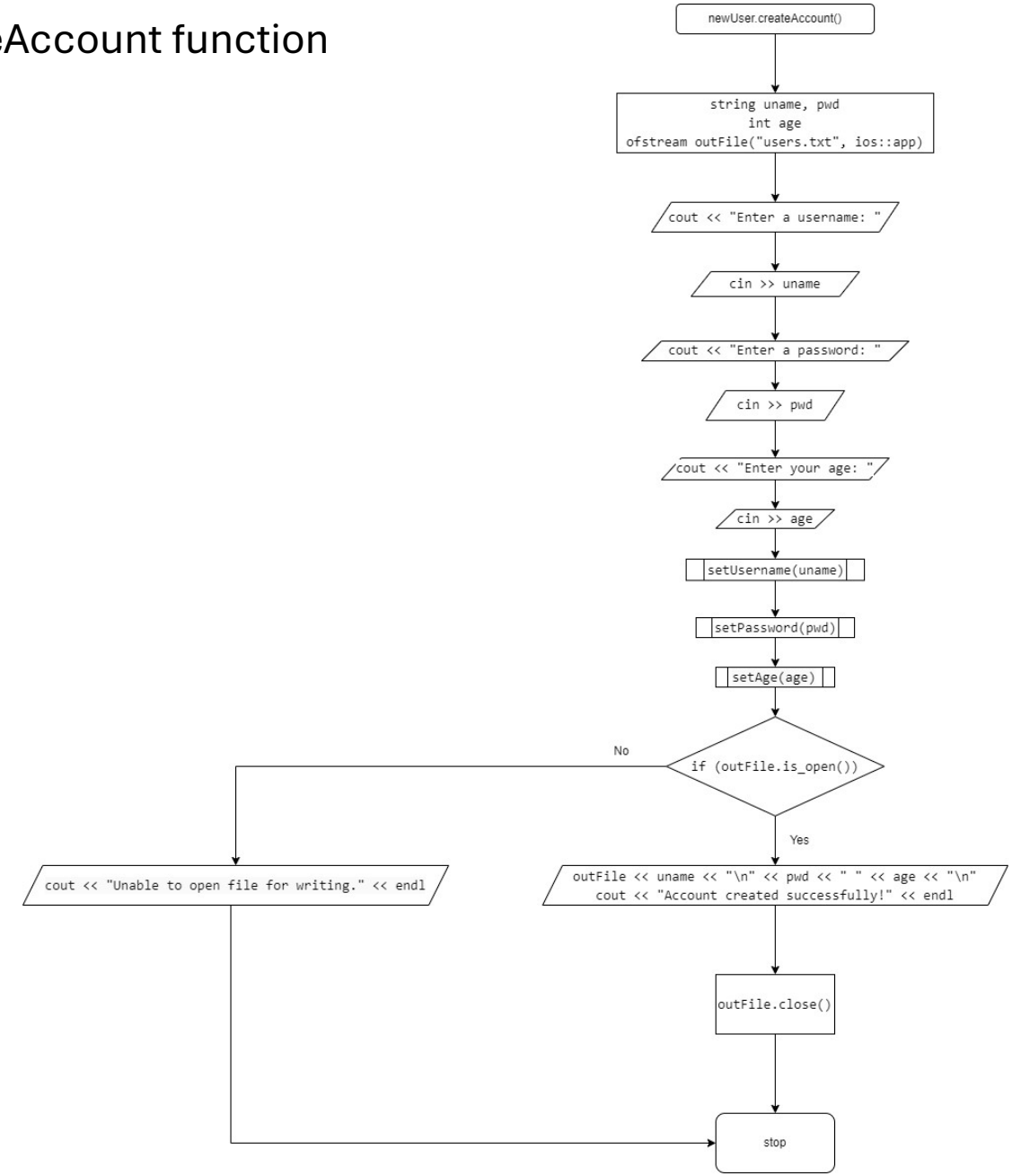
Userinput function



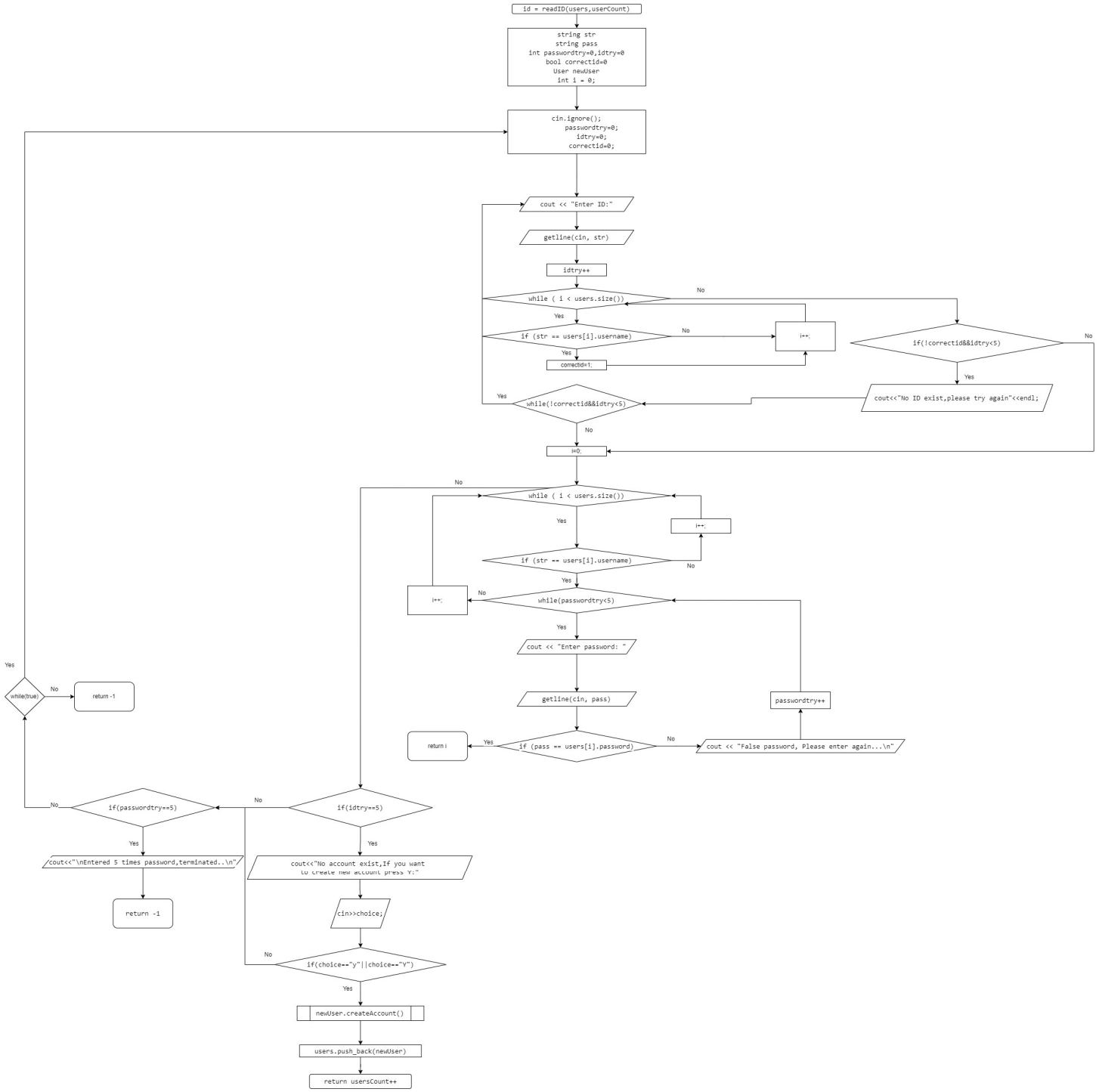
Userchoice function



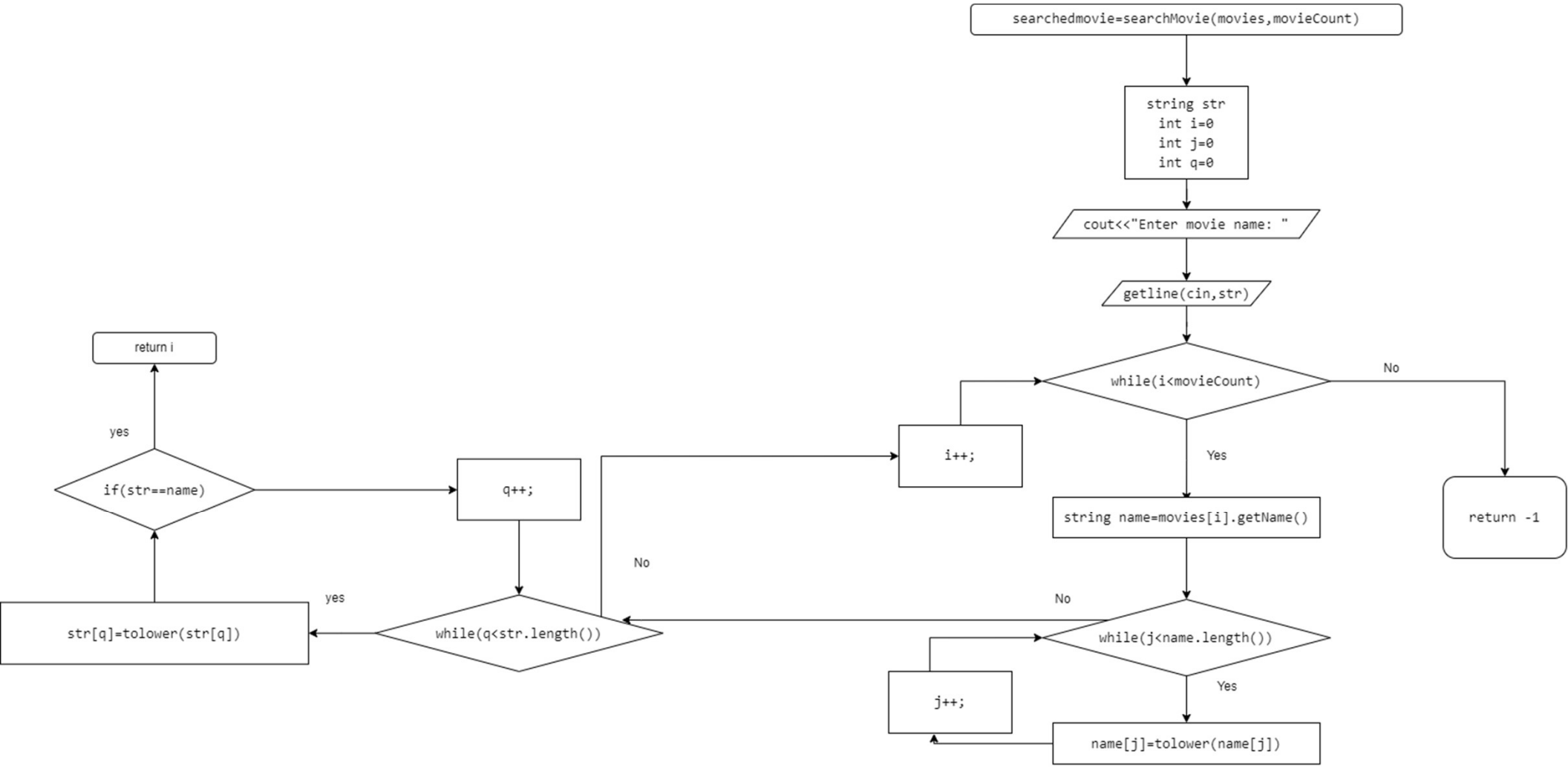
User::createAccount function



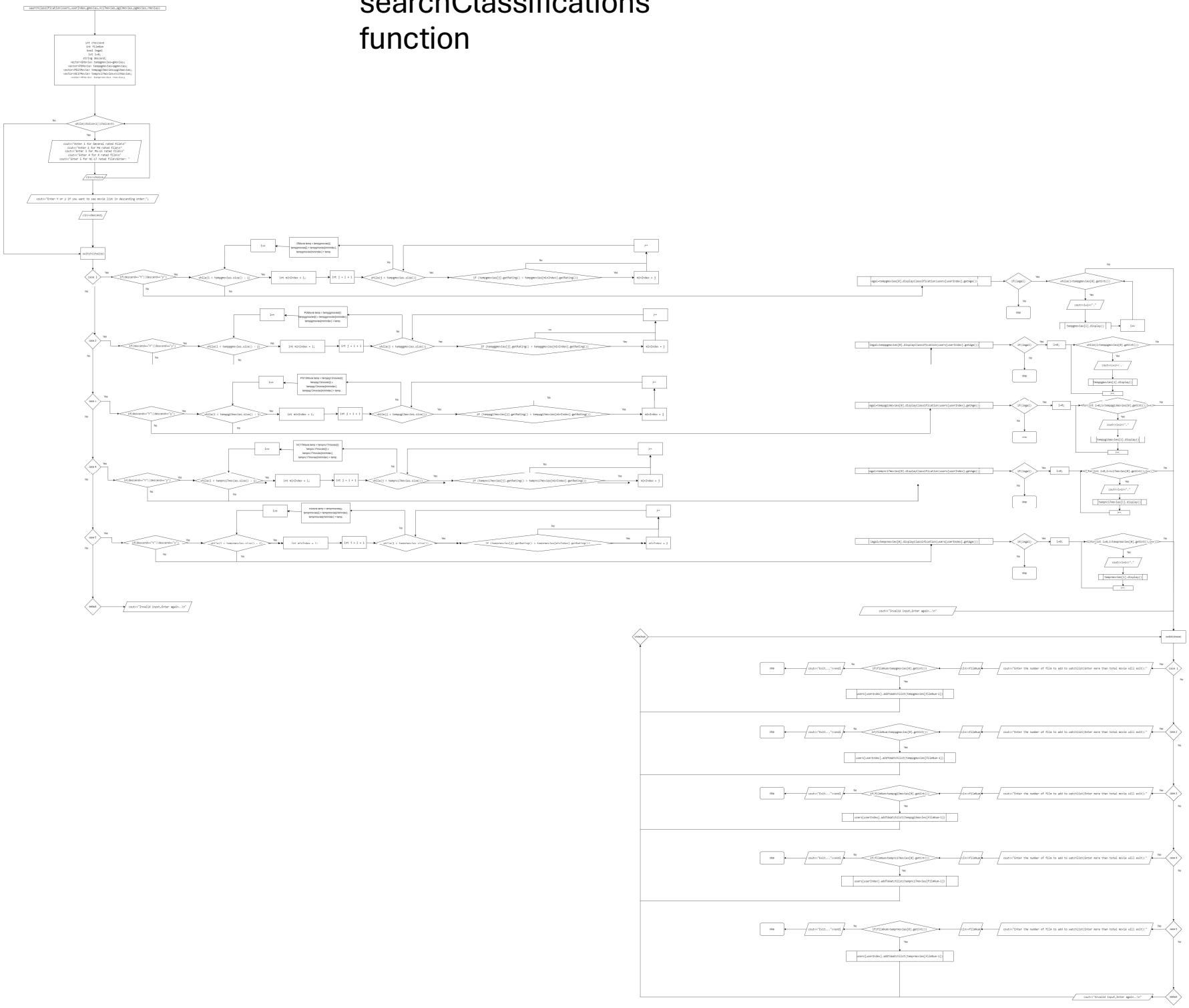
User::readID function



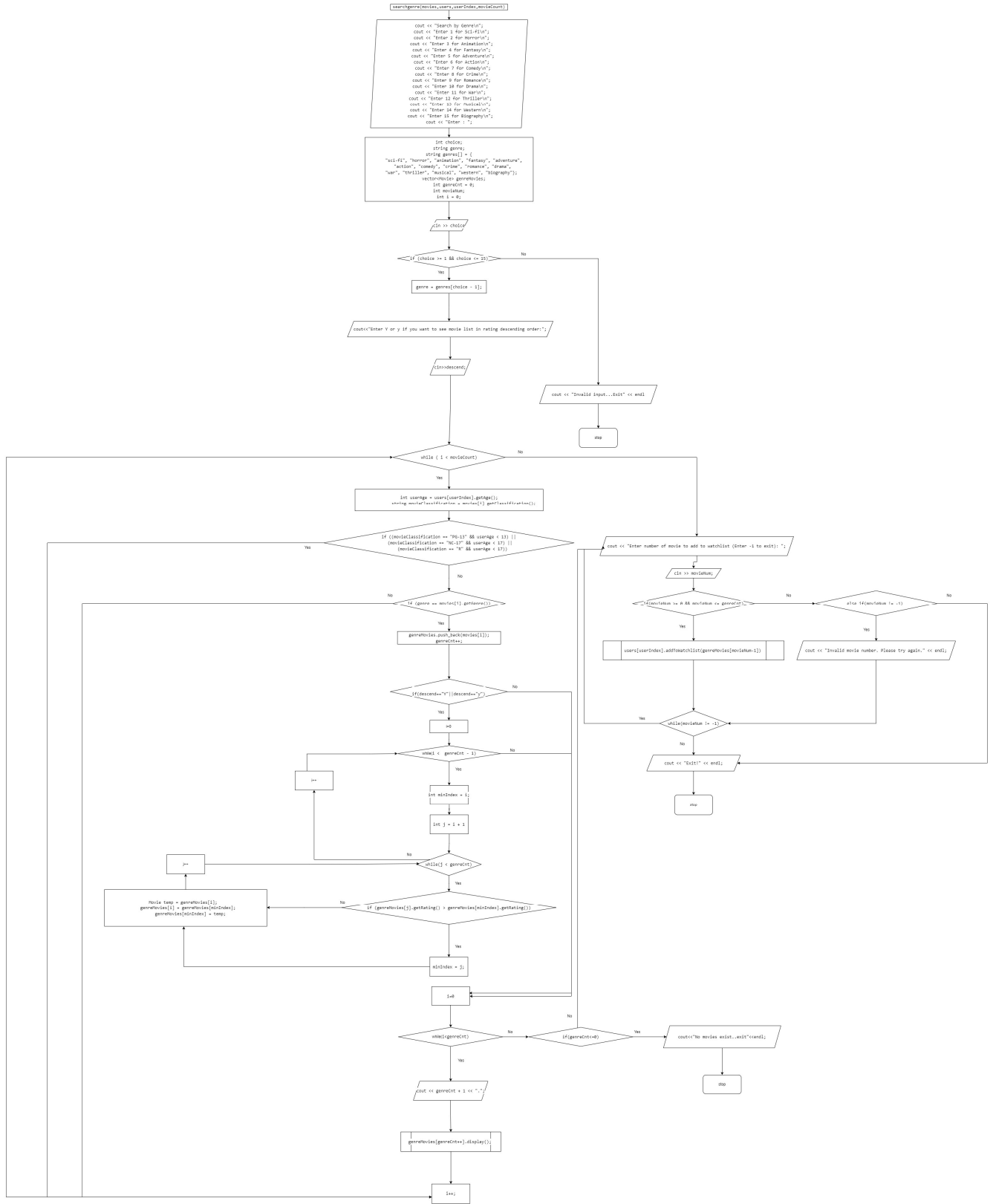
searchMovie function



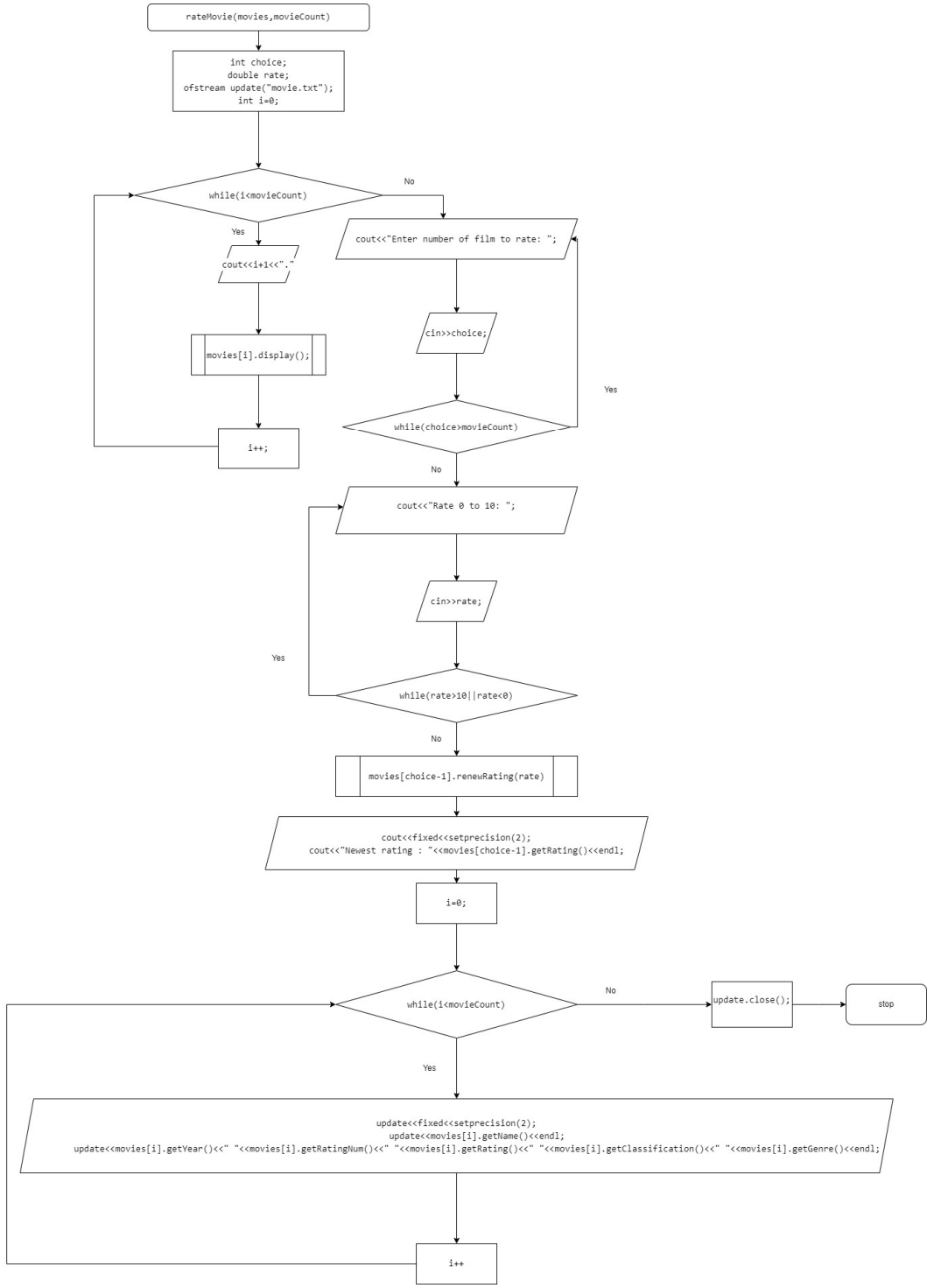
searchClassifications function



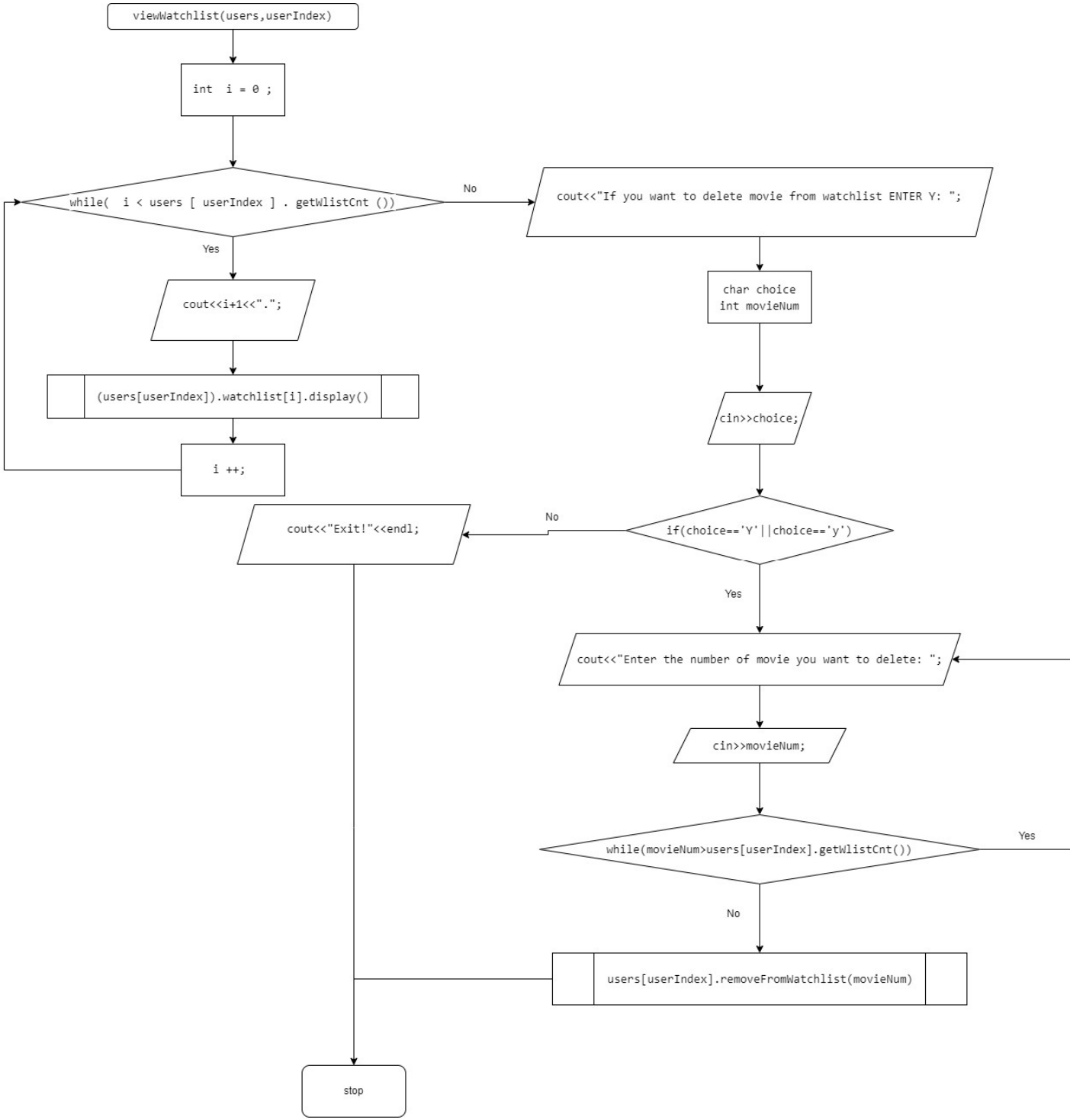
searchgenre function



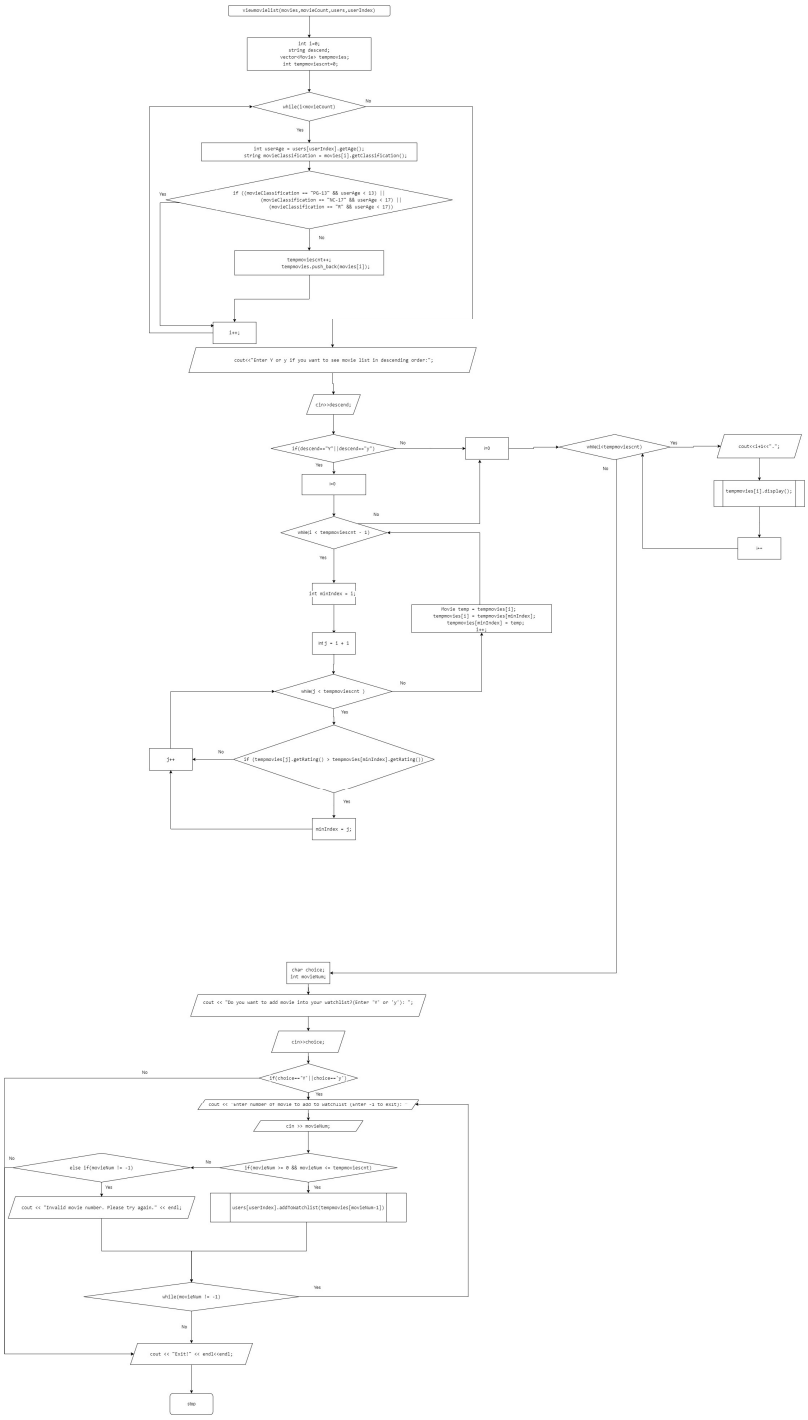
rateMovie function



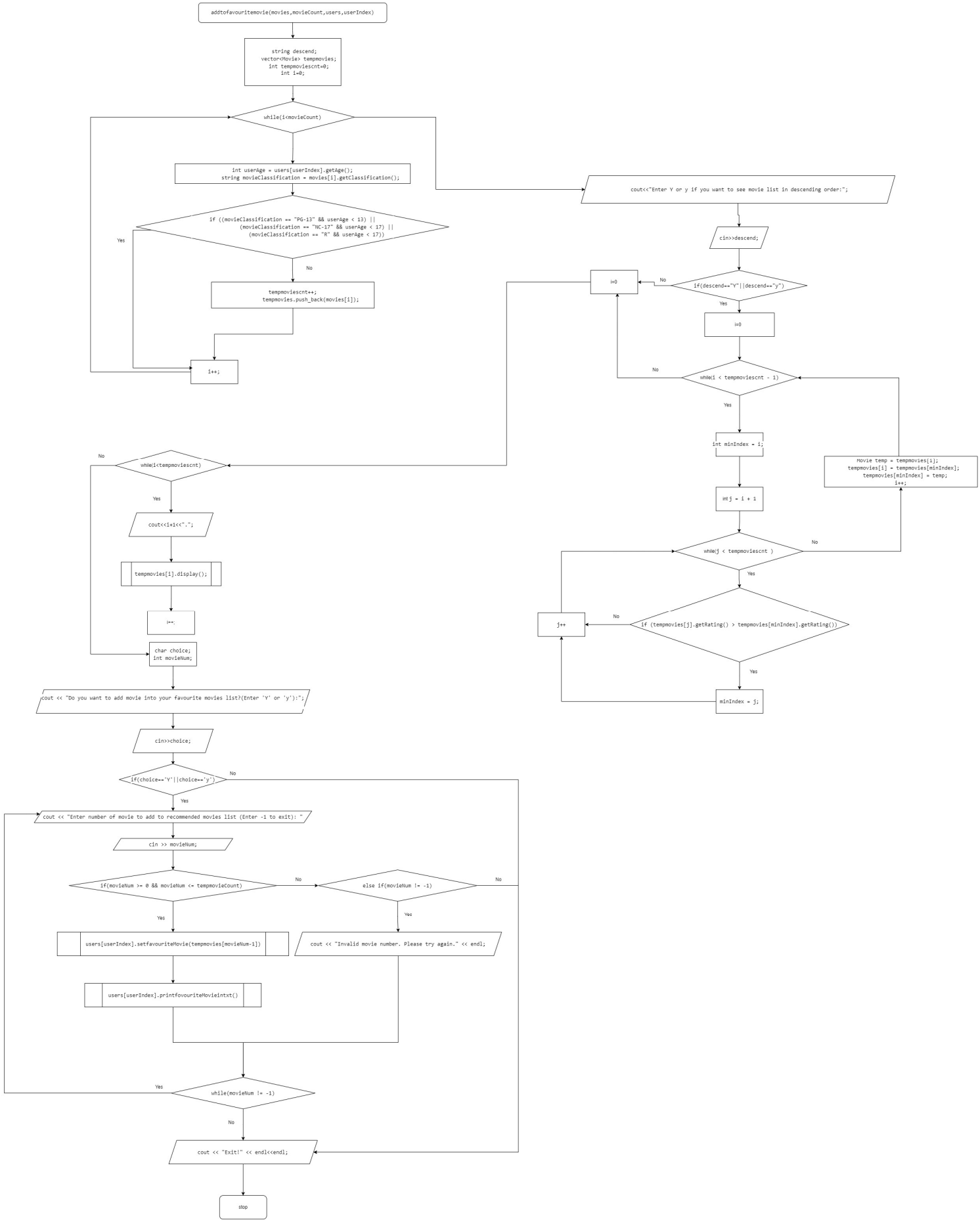
User::viewWatchlist function



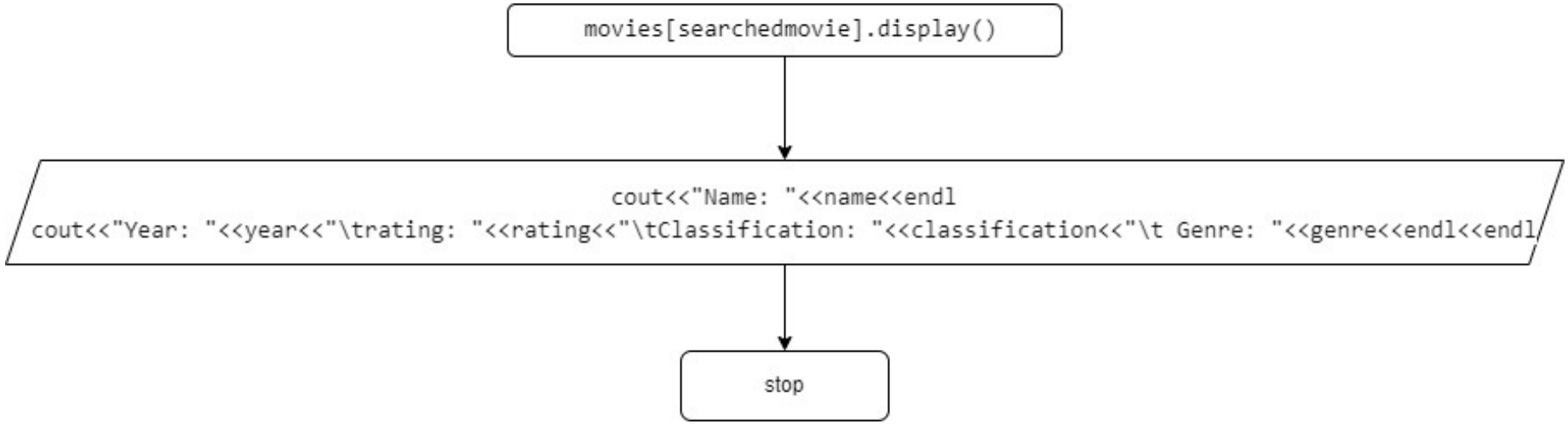
viewmovielist function



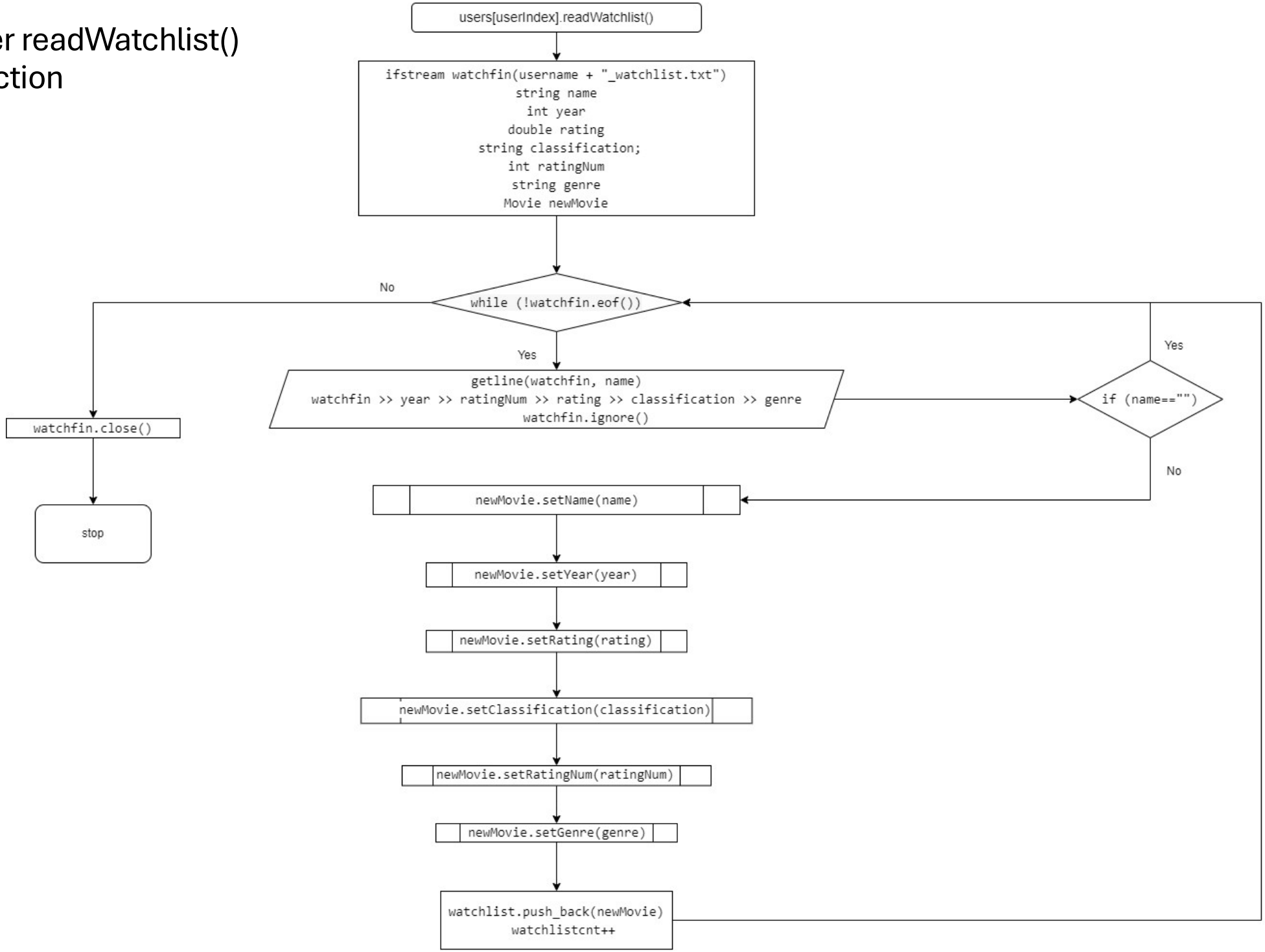
addtofavouritemovie function



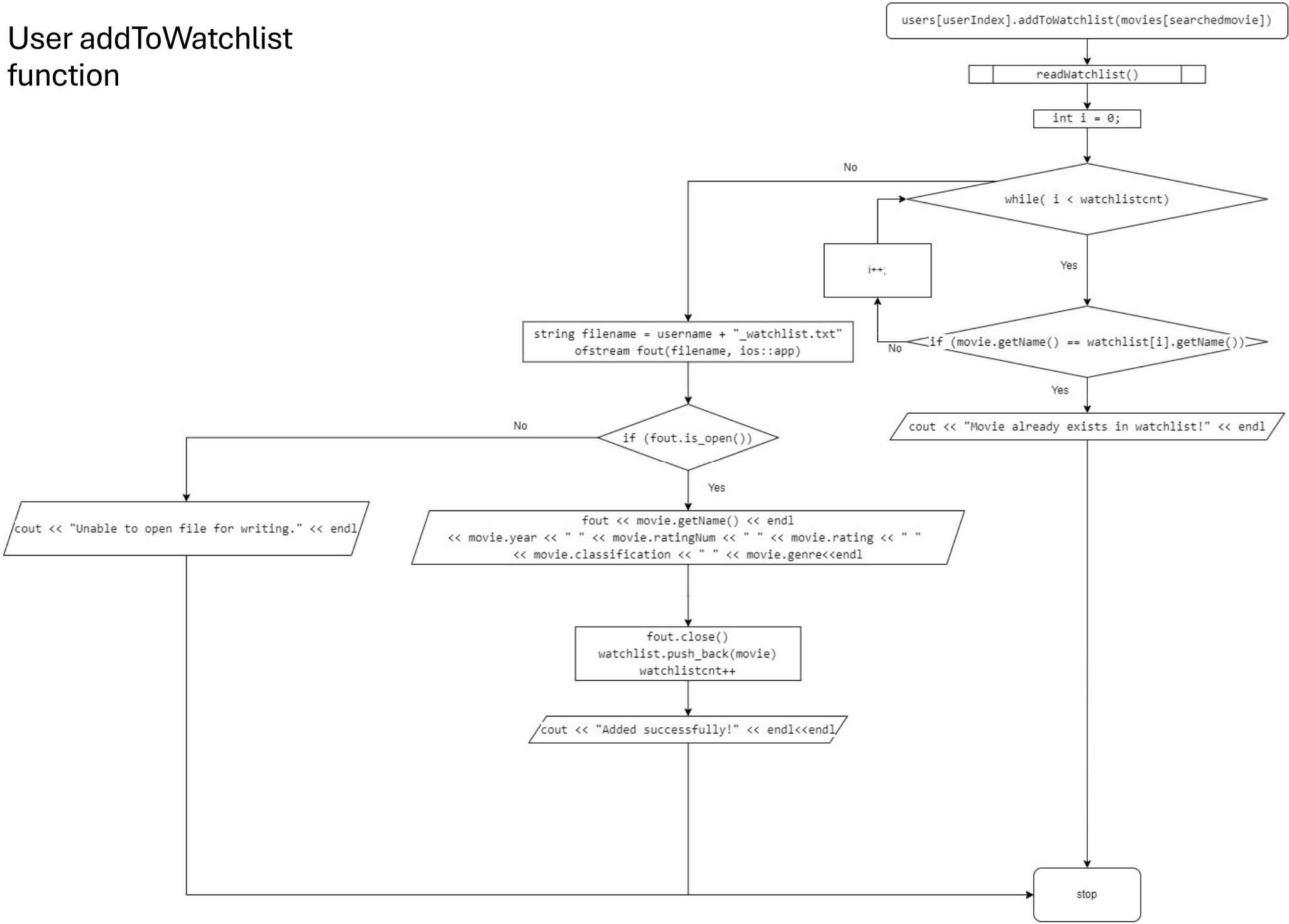
Movie::display() function



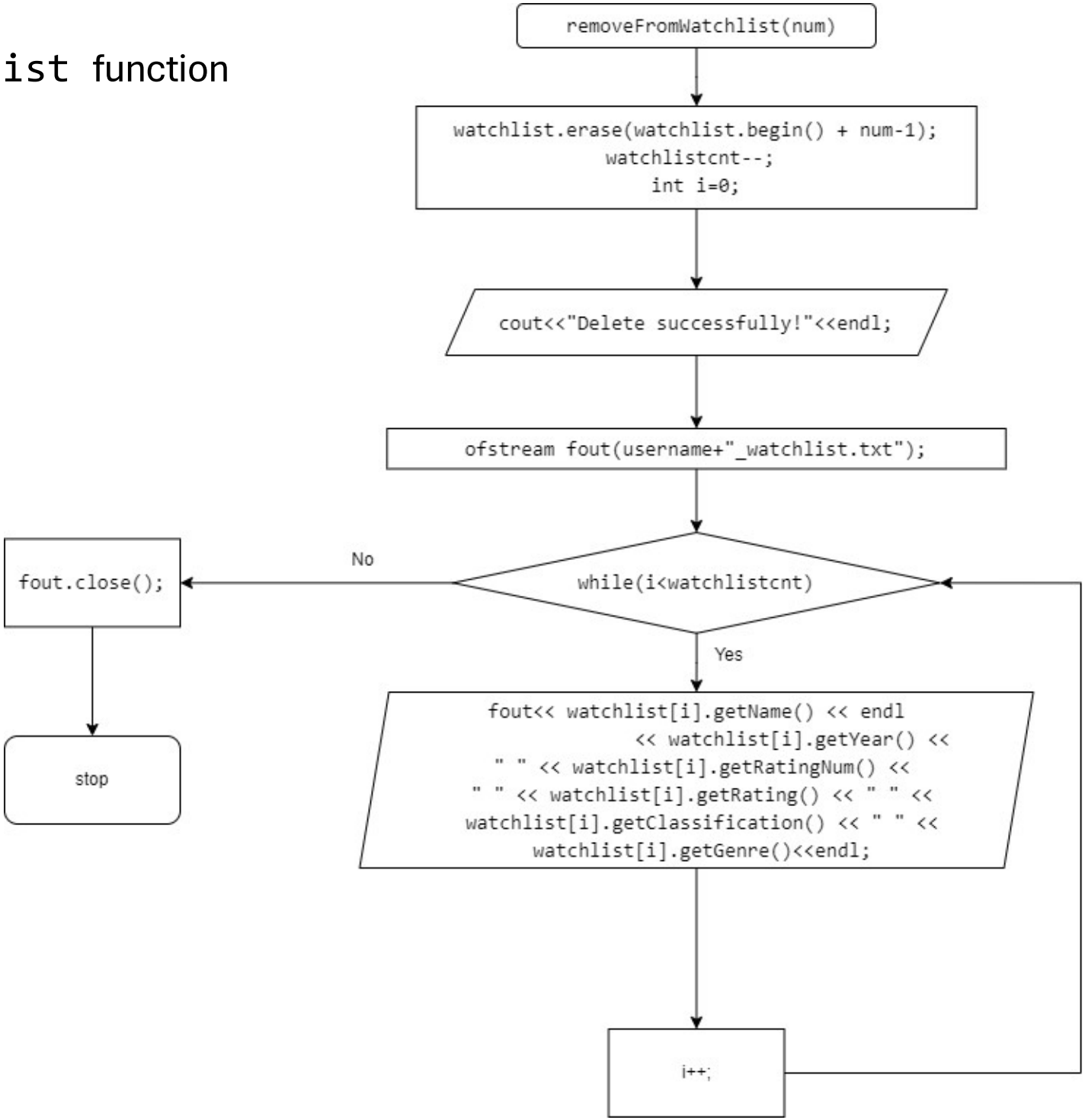
User readWatchlist() function



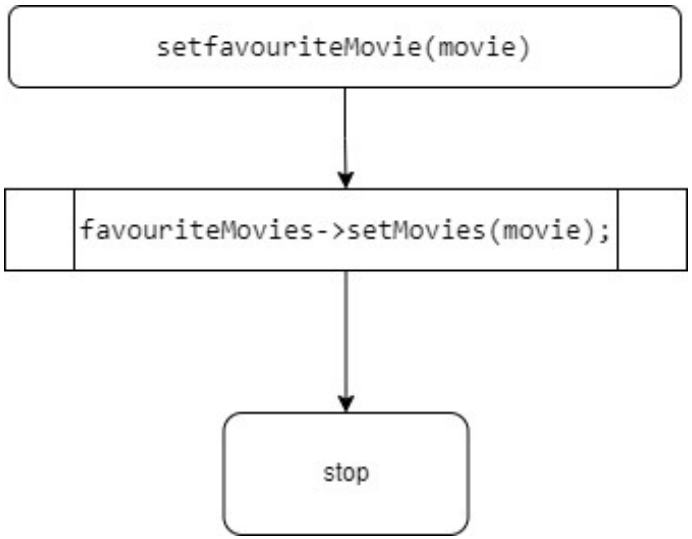
User addToWatchlist function



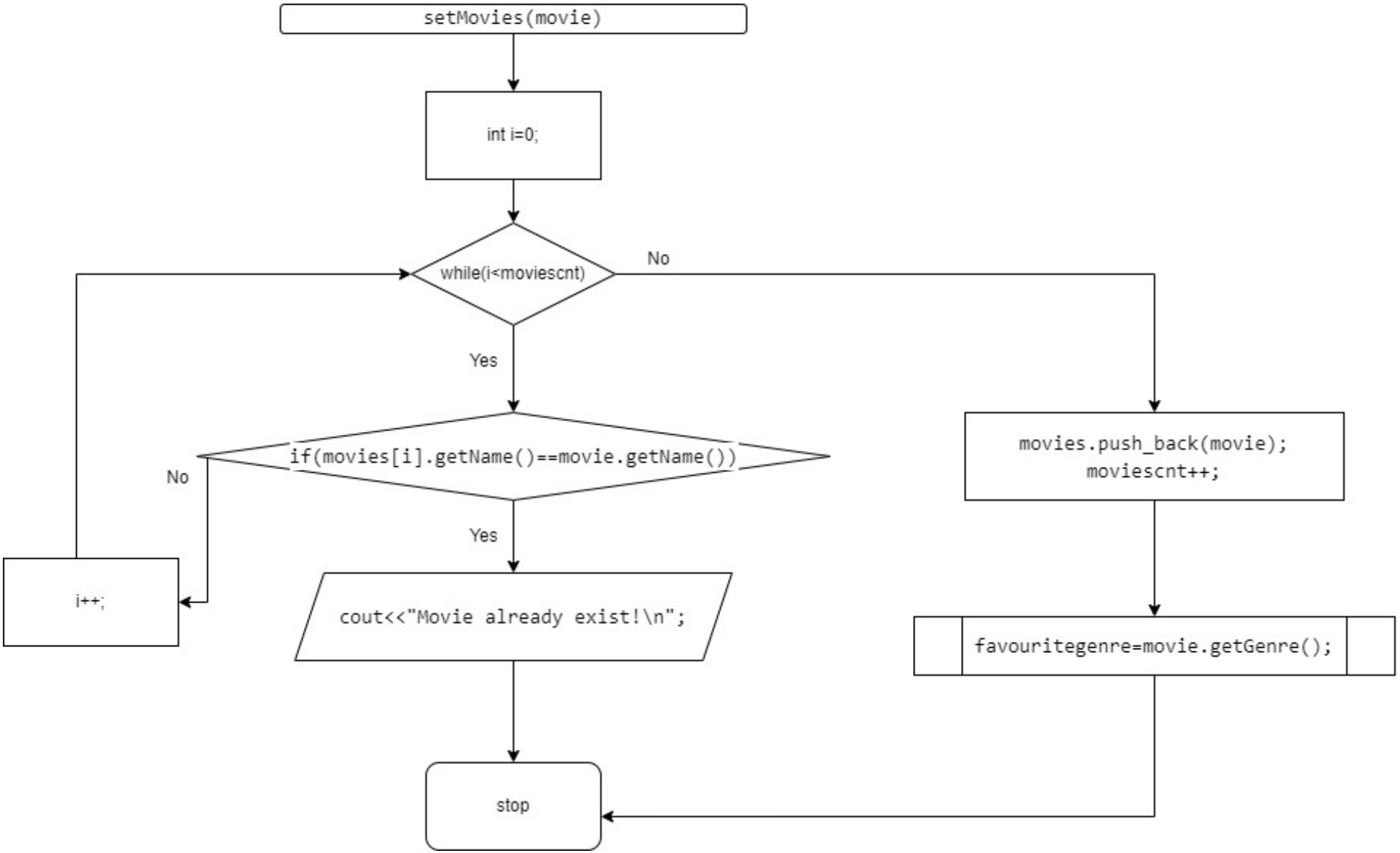
User::removeFromWatchlist function



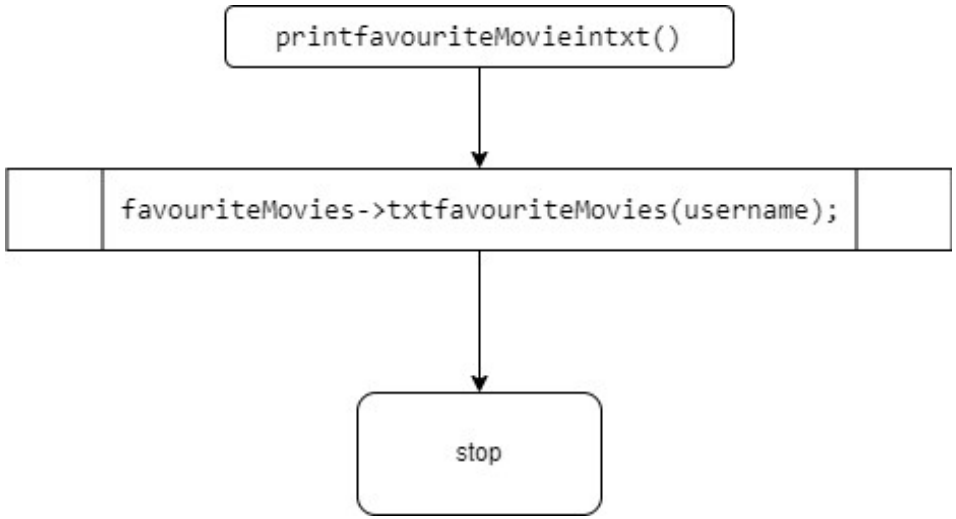
User::`setfavouriteMovie` function



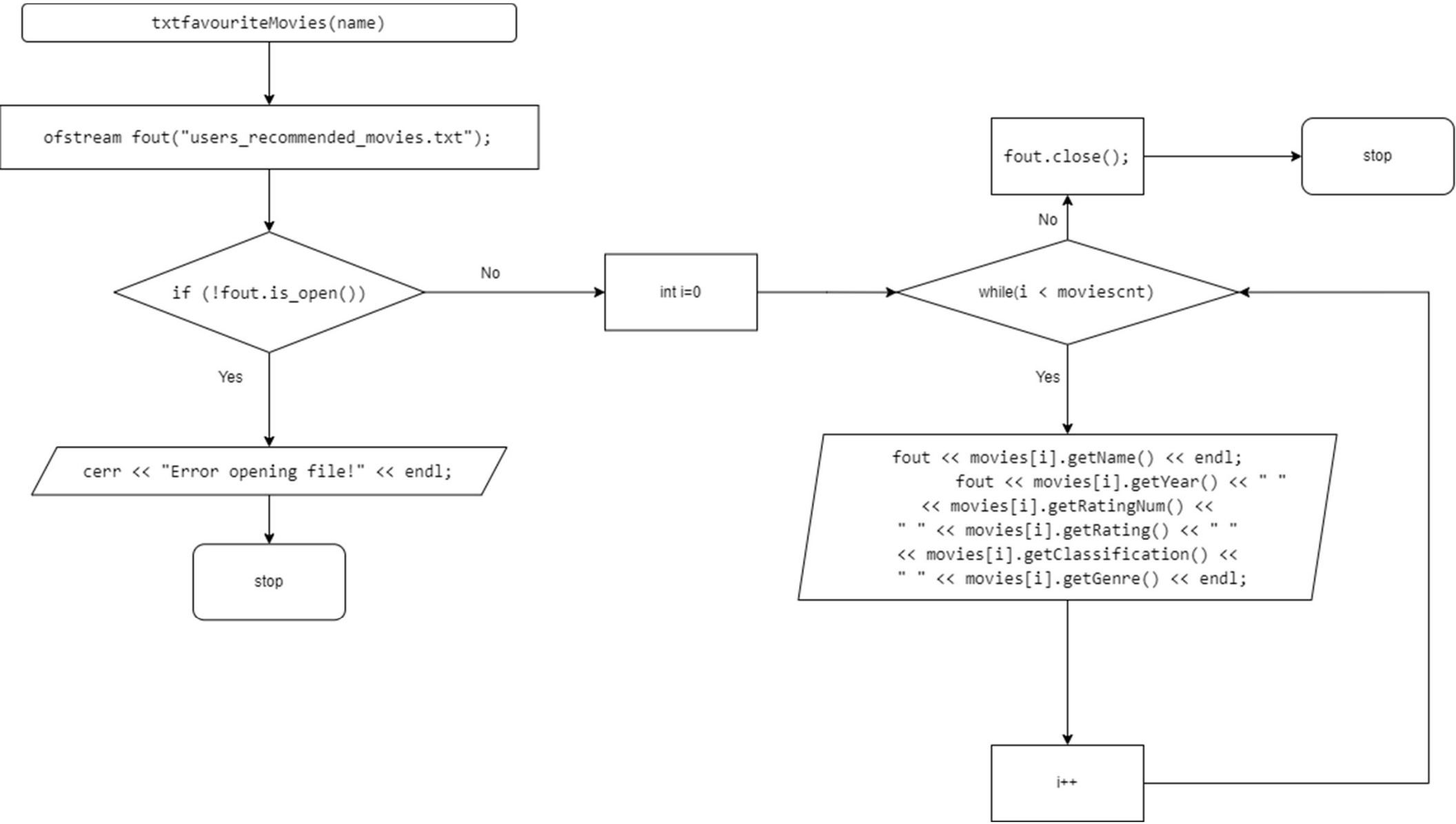
fMovie::setMovies function



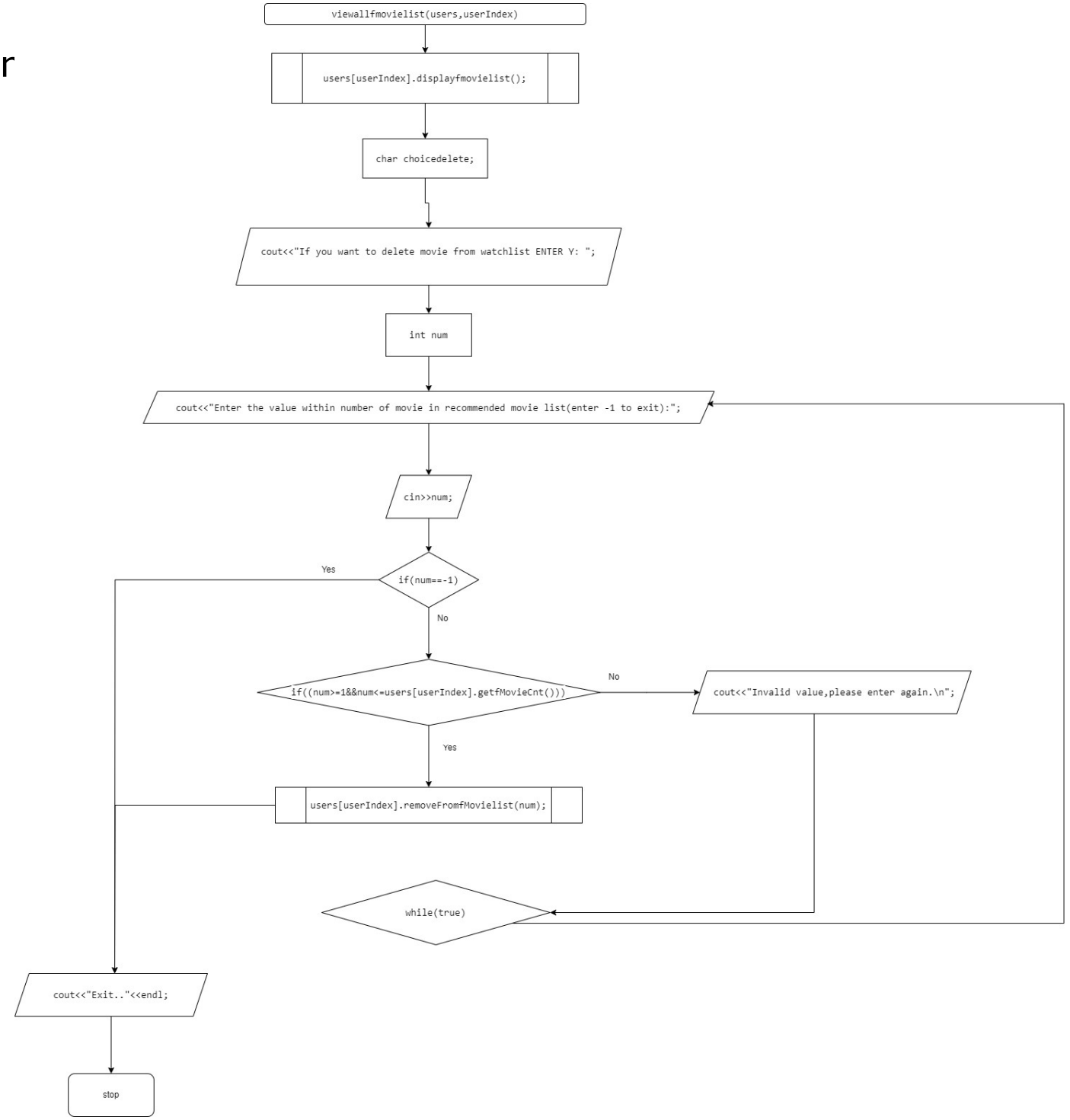
User::`printfavouriteMovieintxt` function



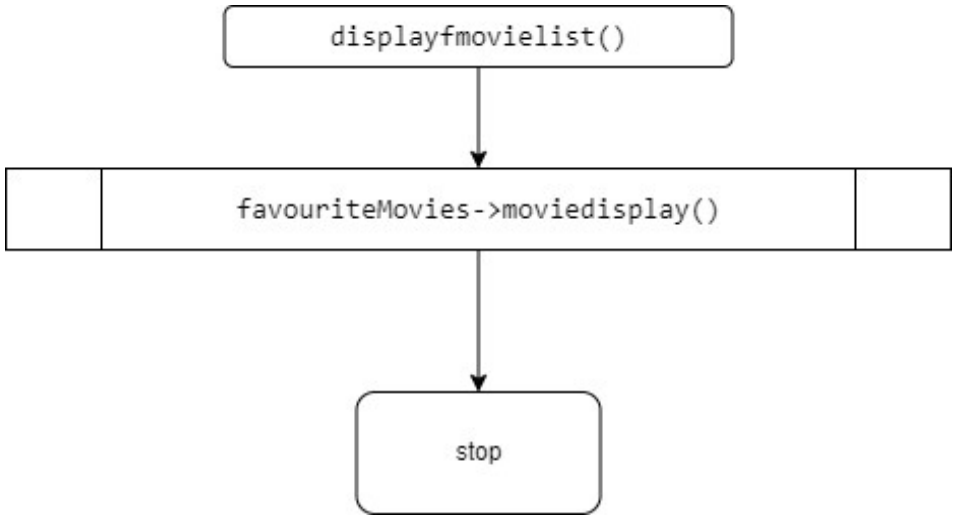
fMovie::txtfavouriteMovies function



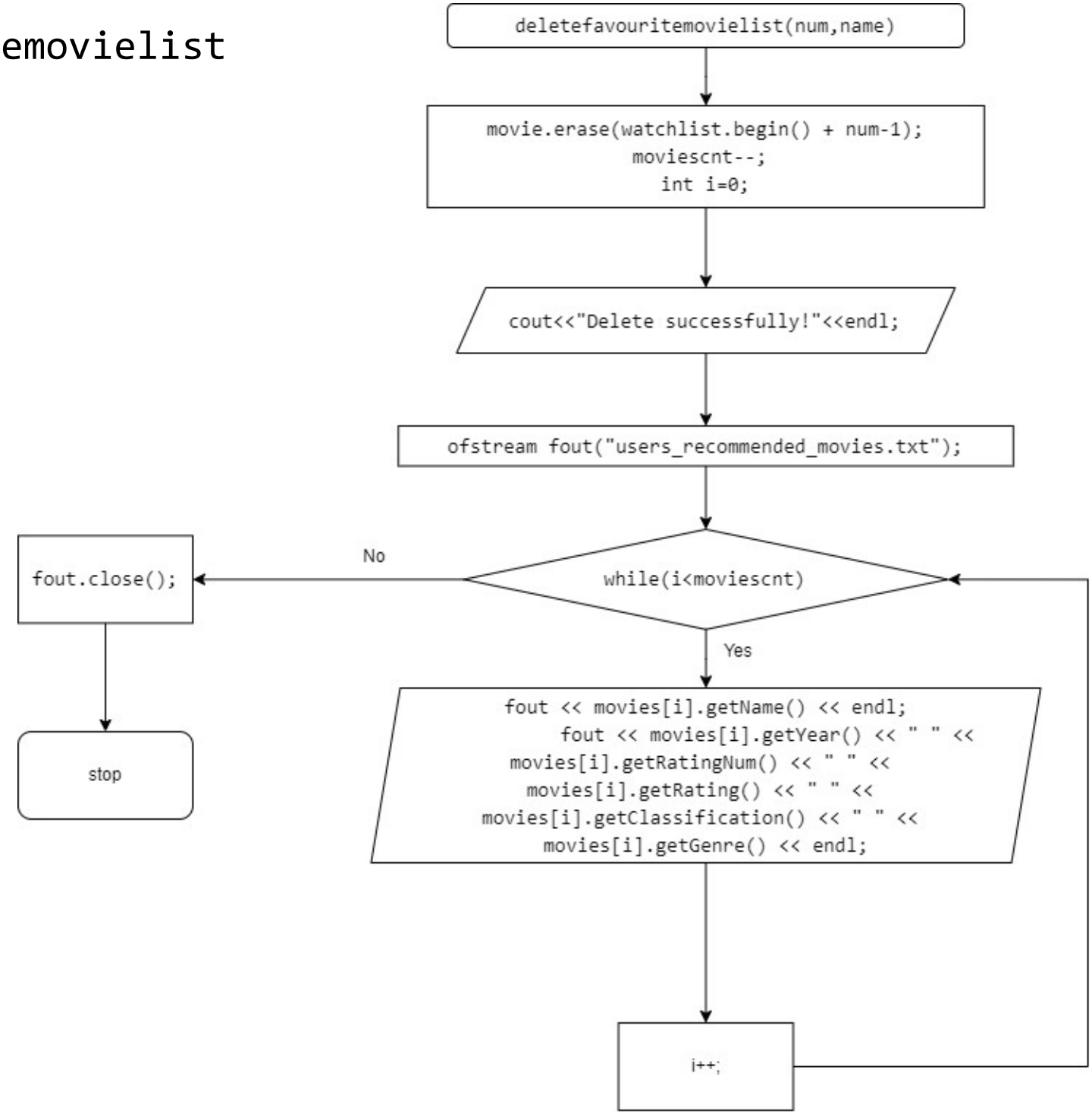
viewallfmovielist functior



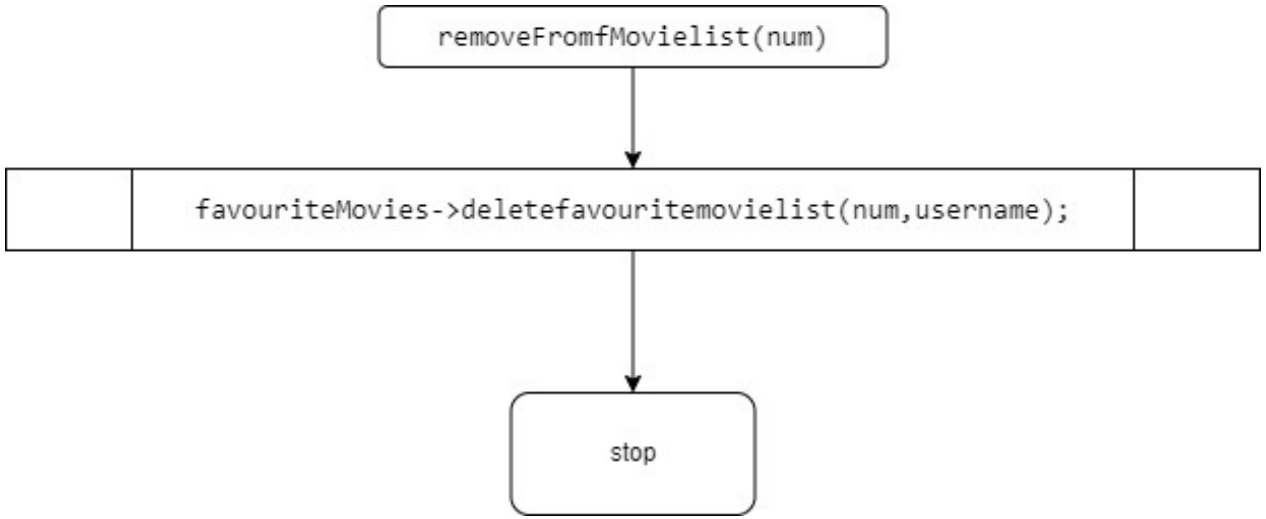
User::displayfmovielist function



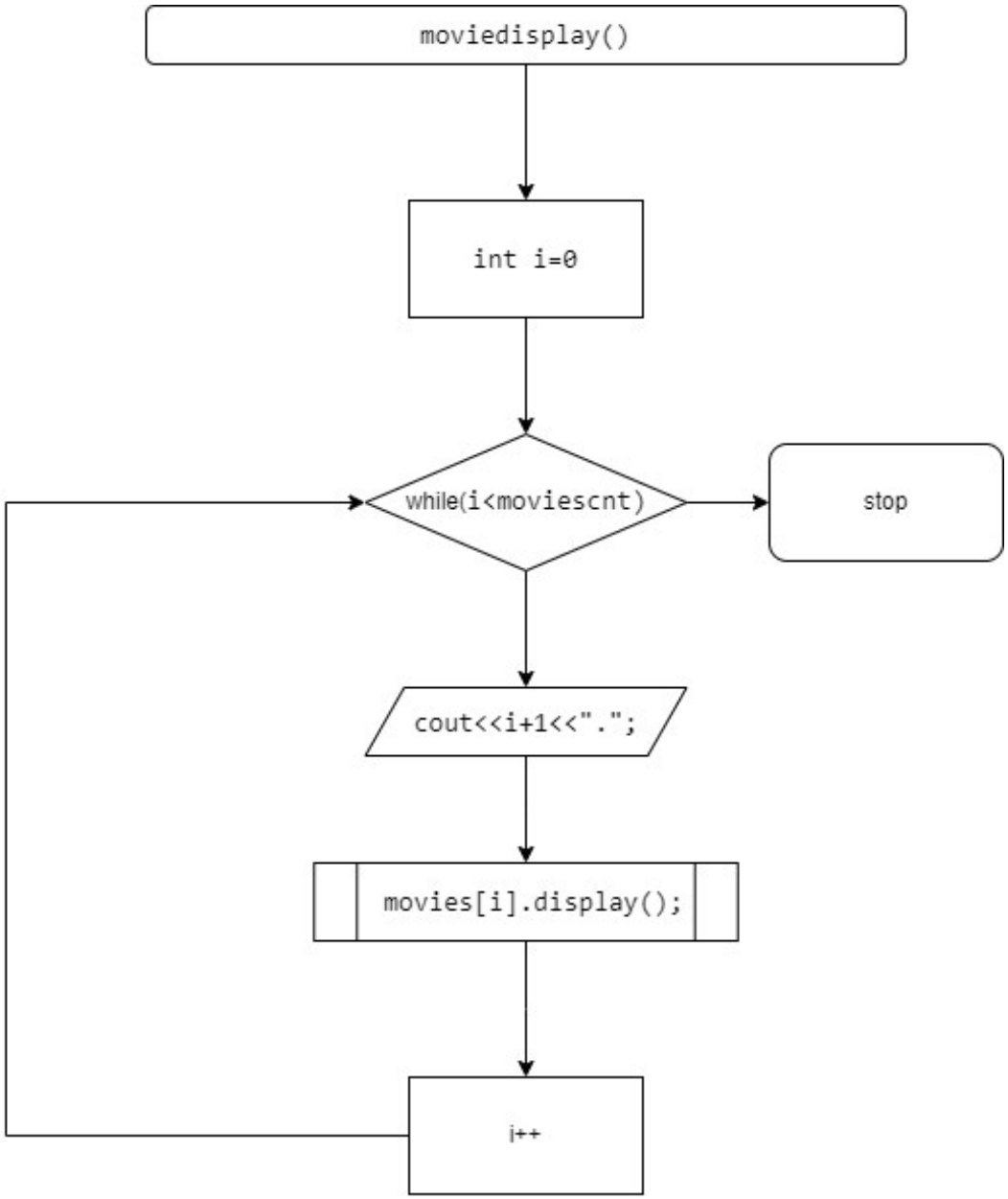
fMovie::deletefavouritemovielist function



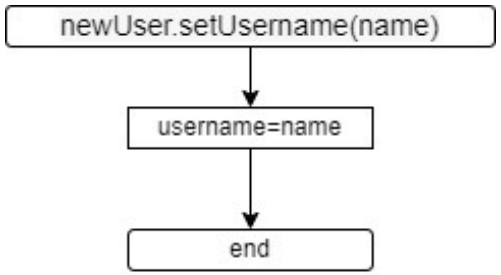
User::removeFromMovielist(num) function

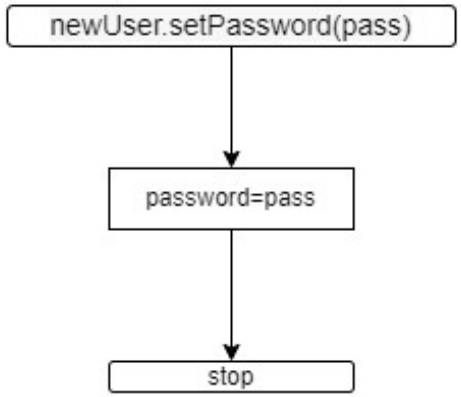


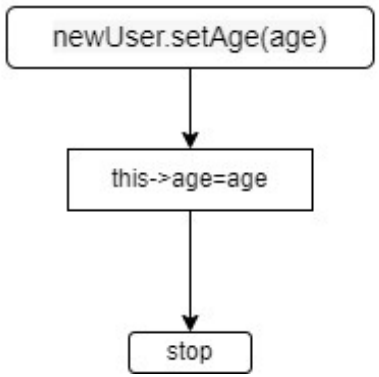
fMovie:moviedisplay function



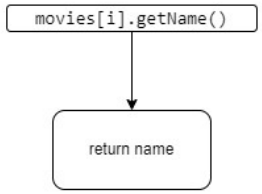
User class accessor and mutator

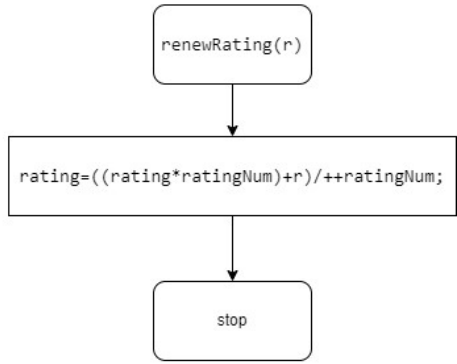


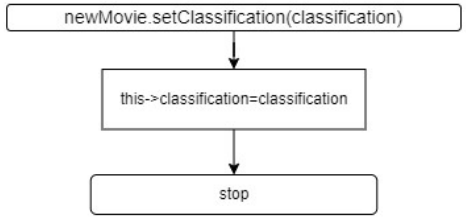


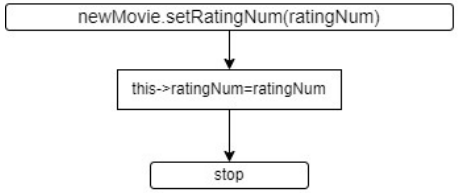


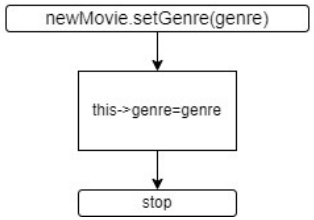
Movie class accessor,mutator and function



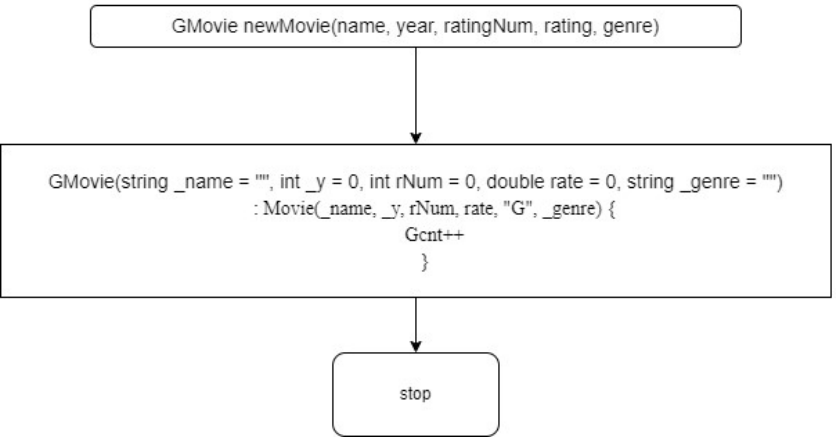


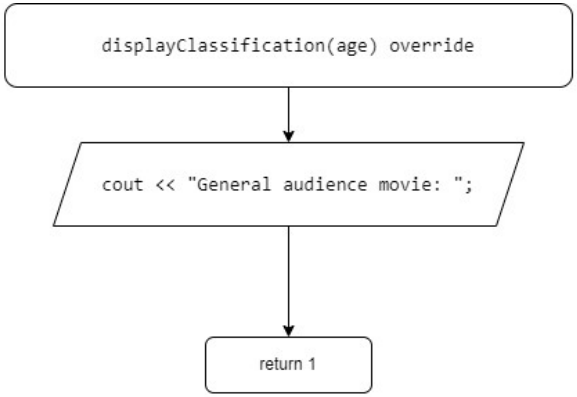


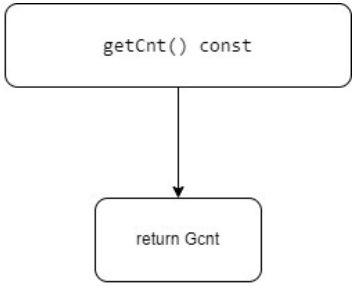




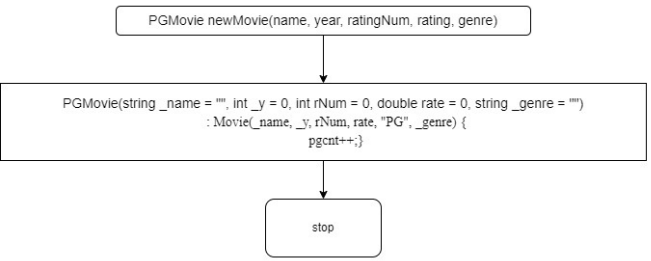
Gmovie class

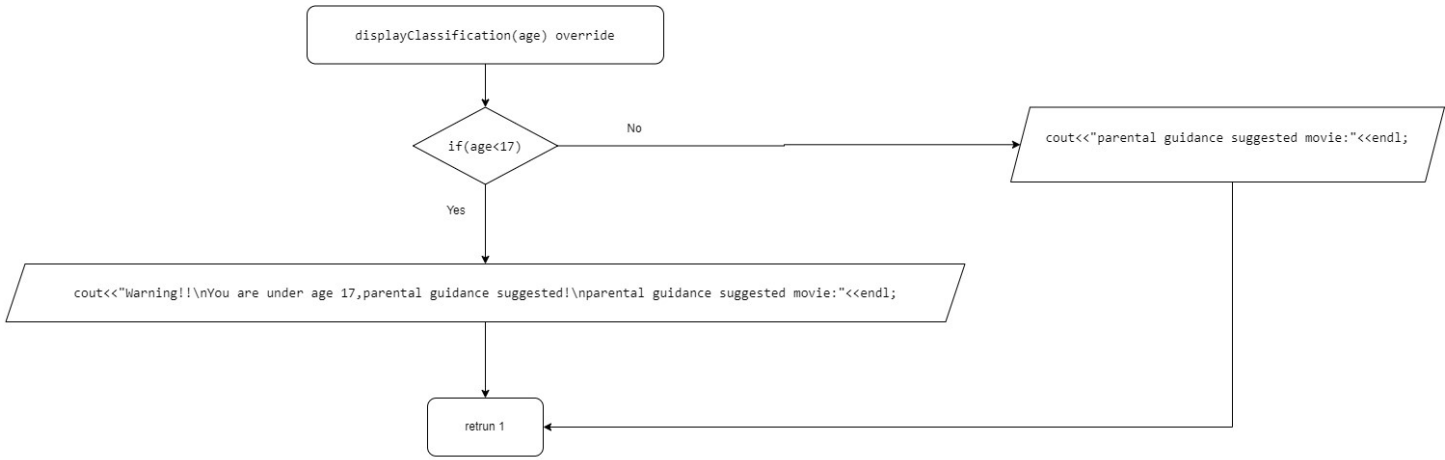


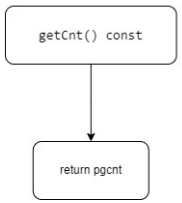




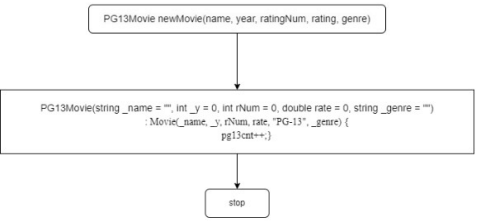
pgMovie class

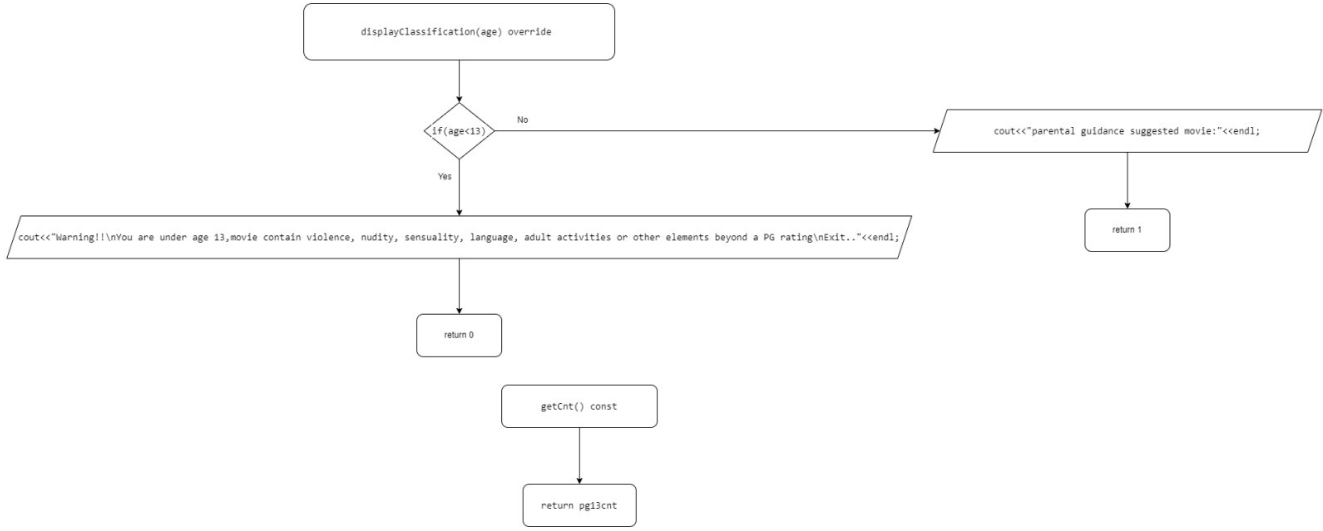




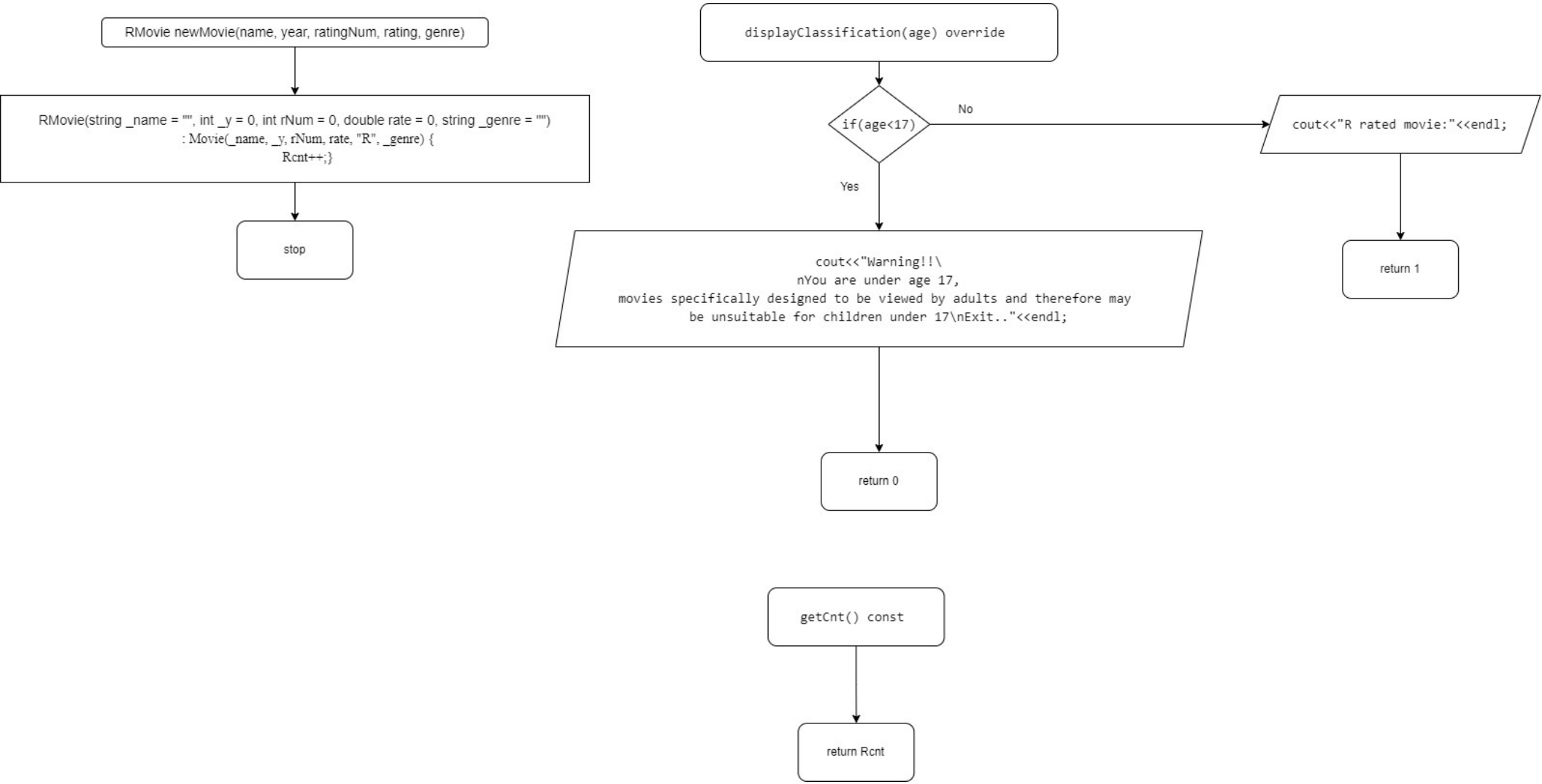


pg13Movie class

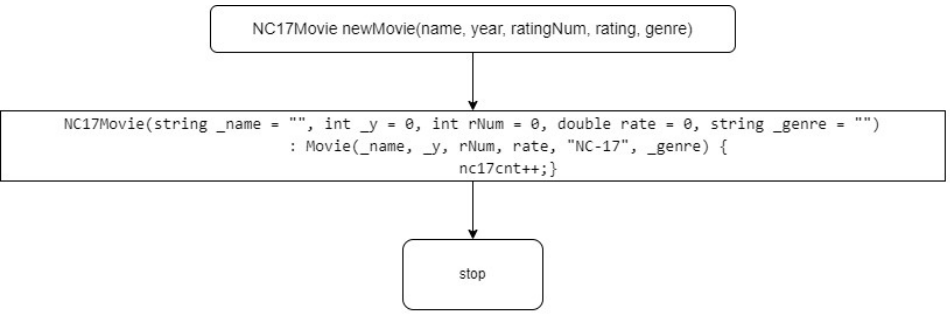


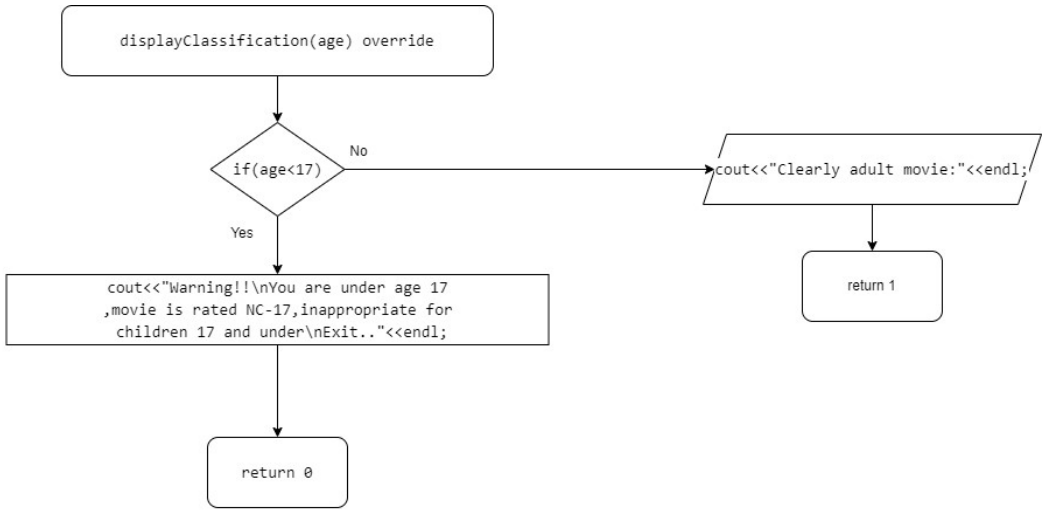


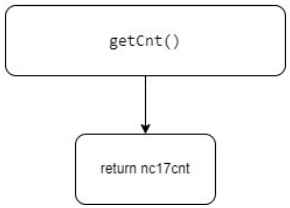
RMovie class



NC17Movie class







Section B: Problem Analysis

From the flowchart prepared in Section A, we are able to identify the objects and classes involved in the movie recommendation system project. Moreover, we can determine the class relationships, including association relationships (aggregation and composition) and inheritance relationships.

Objects

In OOP, an object is an instance of a class. Here are the specific elements for each object in our movie recommendation system:

1. Movie Object
2. Favourite Movie Object
3. General Movie Object
4. Parent Guidance (PG) Movie Object
5. PG13 Movie Object
6. NC17 Movie Object
7. Restricted Movie Object
8. User Object

Classes

A class in OOP is a blueprint for creating objects. It defines a type by encapsulating data and methods that work on the data. Here are the specific elements for each class in our movie recommendation system:

1. **Movie Class**
   Attributes: name, year, rating, classification, ratingNum, genre
   Methods: Movie(), setName(), setYear(), setRating(), setClassification(), setRatingNum(), setGenre(), renerRating(), getName(), getGenre(), getRating(), getYear(), getClassification(), getRatingNum(), display(), displayClassification()

2. **Favourite Movie Class**
   Attributes: moviescnt, favouritegenre, movies
   Methods: setMovies(), txtfavouriteMovies()

3. **General Movie Class**
   Attributes: Gcnt
   Methods: GMovie(), getCnt(), displayClassification()

4. **PG Movie Class**
   Attributes: pgcnt
   Methods: PGMovie(), getCnt(), displayClassification()

5. **PG13 Movie Class**
   Attributes: pg13cnt
   Methods: PG13Movie(), getCnt(), displayClassification()

6. **NC17 Movie Class**
   Attributes: nc17cnt
   Methods: NC17Movie(), getCnt(), displayClassification()

7. **Restricted Movie Class**
   Attributes: Rcnt
   Methods: RMovie(), getCnt(), displayClassification()

8. **User Class**
   Attributes: username, password, watchlist, watchlistcnt, age, favouriteMovies
   Methods: User(), getName(), getAge(), setUsername(), setPassword(), setAge(), getWatchlistCnt(), createAccount(), readID(), readWatchlist(), readfavouritelist(), addToWatchlist(), removeFromWatchlist(), setfavouriteMovie(), printfavouriteMovieintxt(), viewWatchlist()

Class Relationships

Association Relationships:

--Aggregation Relationships:

1. **Favourite Movie – User**
   Reason: A user can have a list of favourite movies but these movies exist independently and can be favourites of multiple users.

--Composition Relationships

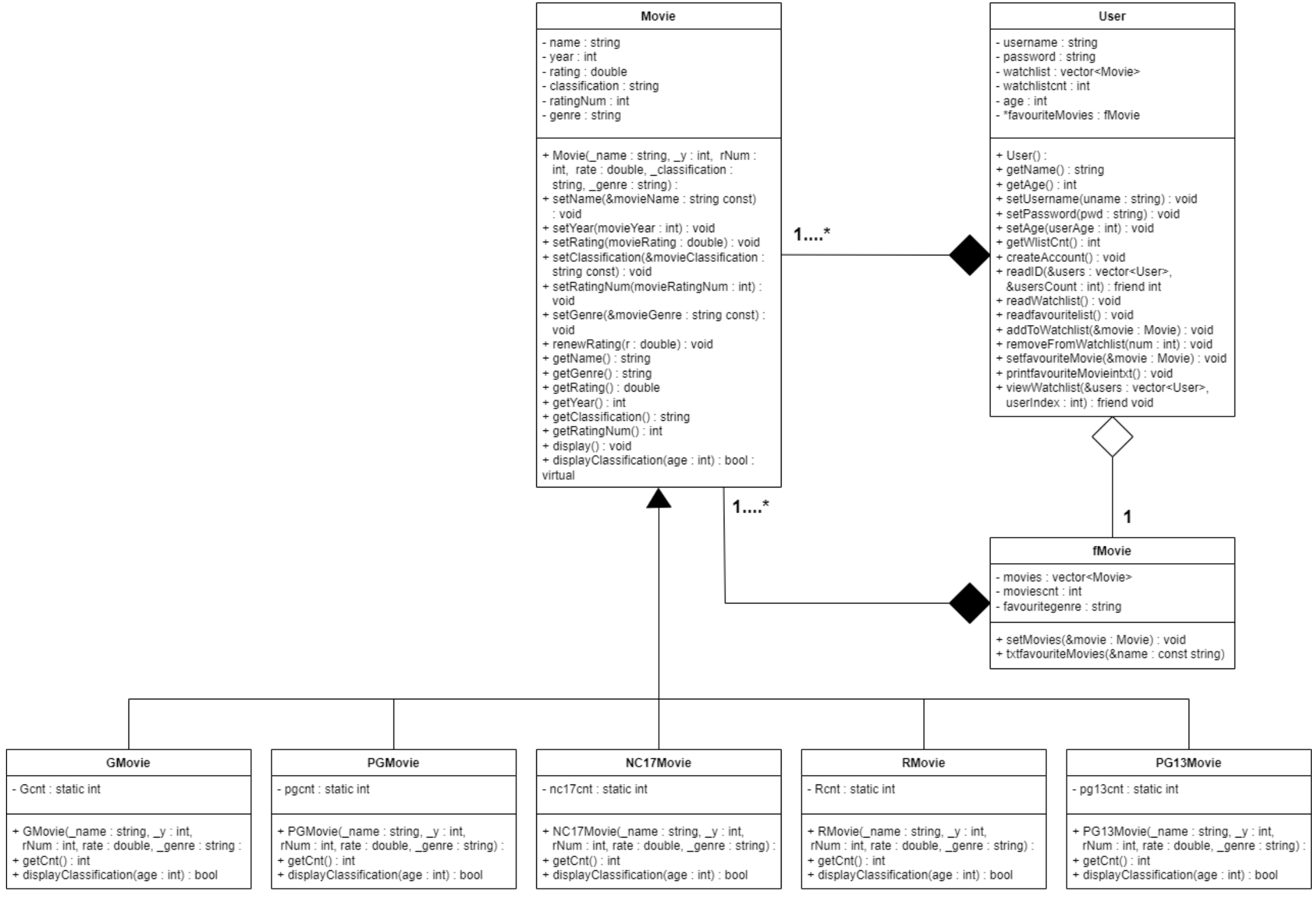
1. **User – Movie**
   Reason: User specific interactions with these movies (such as personal ratings and watchlists) because when user account is deleted, these user specific data points are also deleted.

2. **Favourite Movie – Movie**
   Reason: The presence or absence of a movie in the user's favourite movie list is directly affects the user's favourite movies, this is reflecting their personal preferences and influencing their interaction with the movie recommendation system.

Inheritance relationships:

1. **General Movie – Movie**
2. **PG Movie – Movie**
3. **PG13 Movie – Movie**
4. **NC17 Movie - Movie**
5. **Restricted Movie – Movie**

   Reason: General movie is a movie. PG movie is also a movie. Moreover, PG13 movie, NC17 movie and restricted movie are also a movie. Whereby movie is the base class. General movie, PG movie, PG13 movie, NC17 movie and restricted movie are the derived classes. Hence, general movie, PG movie, PG13 movie, NC17 movie and restricted movie are inherited from class movie.


Movie
- name : string
- year : int
- rating : double
- classification : string
- ratingNum : int
- genre : string
+ Movie(_name : string, _y : int, rNum : int, rate : double, _classification : string, _genre : string) :
+ setName(&movieName : string const) : void
+ setYear(movieYear : int) : void
+ setRating(movieRating : double) : void
+ setClassification(&movieClassification : string const) : void
+ setRatingNum(movieRatingNum : int) : void
+ setGenre(&movieGenre : string const) : void
+ renewRating(r : double) : void
+ getName() : string
+ getGenre() : string
+ getRating() : double
+ getYear() : int
+ getClassification() : string
+ getRatingNum() : int
+ display() : void
+ displayClassification(age : int) : bool : virtual
User
- username : string
- password : string
- watchlist : vector<Movie>
- watchlistcnt : int
- age : int
- *favouriteMovies : fMovie
+ User() :
+ getName() : string
+ getAge() : int
+ setUsername(uname : string) : void
+ setPassword(pwd : string) : void
+ setAge(userAge : int) : void
+ getWlistCnt() : int
+ createAccount() : void
+ readID(&users : vector<User>, &usersCount : int) : friend int
+ readWatchlist() : void
+ readfavouritelist() : void
+ addToWatchlist(&movie : Movie) : void
+ removeFromWatchlist(num : int) : void
+ setfavouriteMovie(&movie : Movie) : void
+ printfavouriteMovieintxt() : void
+ viewWatchlist(&users : vector<User>, userIndex : int) : friend void
1....*
1....*
1
fMovie
- movies : vector<Movie>
- moviescnt : int
- favouritegenre : string
+ setMovies(&movie : Movie) : void
+ txtfavouriteMovies(&name : const string)
GMovie
- Gcnt : static int
+ GMovie(_name : string, _y : int, rNum : int, rate : double, _genre : string :
+ getCnt() : int
+ displayClassification(age : int) : bool
PGMovie
- pgcnt : static int
+ PGMovie(_name : string, _y : int, rNum : int, rate : double, _genre : string) :
+ getCnt() : int
+ displayClassification(age : int) : bool
NC17Movie
- nc17cnt : static int
+ NC17Movie(_name : string, _y : int, rNum : int, rate : double, _genre : string) :
+ getCnt() : int
+ displayClassification(age : int) : bool
RMovie
- Rcnt : static int
+ RMovie(_name : string, _y : int, rNum : int, rate : double, _genre : string) :
+ getCnt() : int
+ displayClassification(age : int) : bool
PG13Movie
- pg13cnt : static int
+ PG13Movie(_name : string, _y : int, rNum : int, rate : double, _genre : string) :
+ getCnt() : int
+ displayClassification(age : int) : bool