



DATA STRUCTURE & ALGORITHM

PROJECT

THE TRIO MEMBERS

- 1. EDDY KOH WEI HEN (A22EC0154)**
 - 2. NUR HAFIZAH BINTI JAFRI (A22EC5022)**
 - 3. NURSYUHADA BINTI BADREN (A22EC0253)**
- 



CONTENT



01

PROJECT DESCRIPTION

02

PROJECT DESIGN

03

DATA STRUCTURE OF THE
CONCEPTS EMPLOYED

04

EXECUTION OF THE PROJECT



PROJECT DESCRIPTION

THIS PROJECT IS AN AGENT-BASED HOTEL BOOKING SYSTEM THAT USES CLASSES TO REPRESENT CUSTOMERS, A QUEUE TO MANAGE RESERVATIONS, AND A STACK TO HANDLE ACCEPTED BOOKING REQUESTS. THERE ARE TWO CHARACTERS THAT USE THE SYSTEM, WHO ARE THE AGENT AND THE HOTEL MANAGER. THE AGENT IS THE PERSON WHO HELPS THE CUSTOMERS TO DO A BOOKING REQUEST AND THE BOOKING LIST WILL PASS TO THE HOTEL MANAGER. THE HOTEL MANAGER WILL ACCEPT THE BOOKING REQUEST FROM THE LIST PASSED BY THE AGENT. CUSTOMERS ARE CHARACTERIZED BY ATTRIBUTES SUCH AS NAME, AGE, IC NUMBER, PHONE NUMBER, ROOM NUMBER, AND CHECK-IN DATE.

THE QUEUE CLASS MANAGES THE RESERVATION LIST AND OFFERS OPERATIONS SUCH AS ADDING RESERVATIONS, SEARCHING FOR CUSTOMER DATA, DISPLAYING THE RESERVATION LIST, AND PROCESSING BOOKINGS BY HOTEL AGENTS.

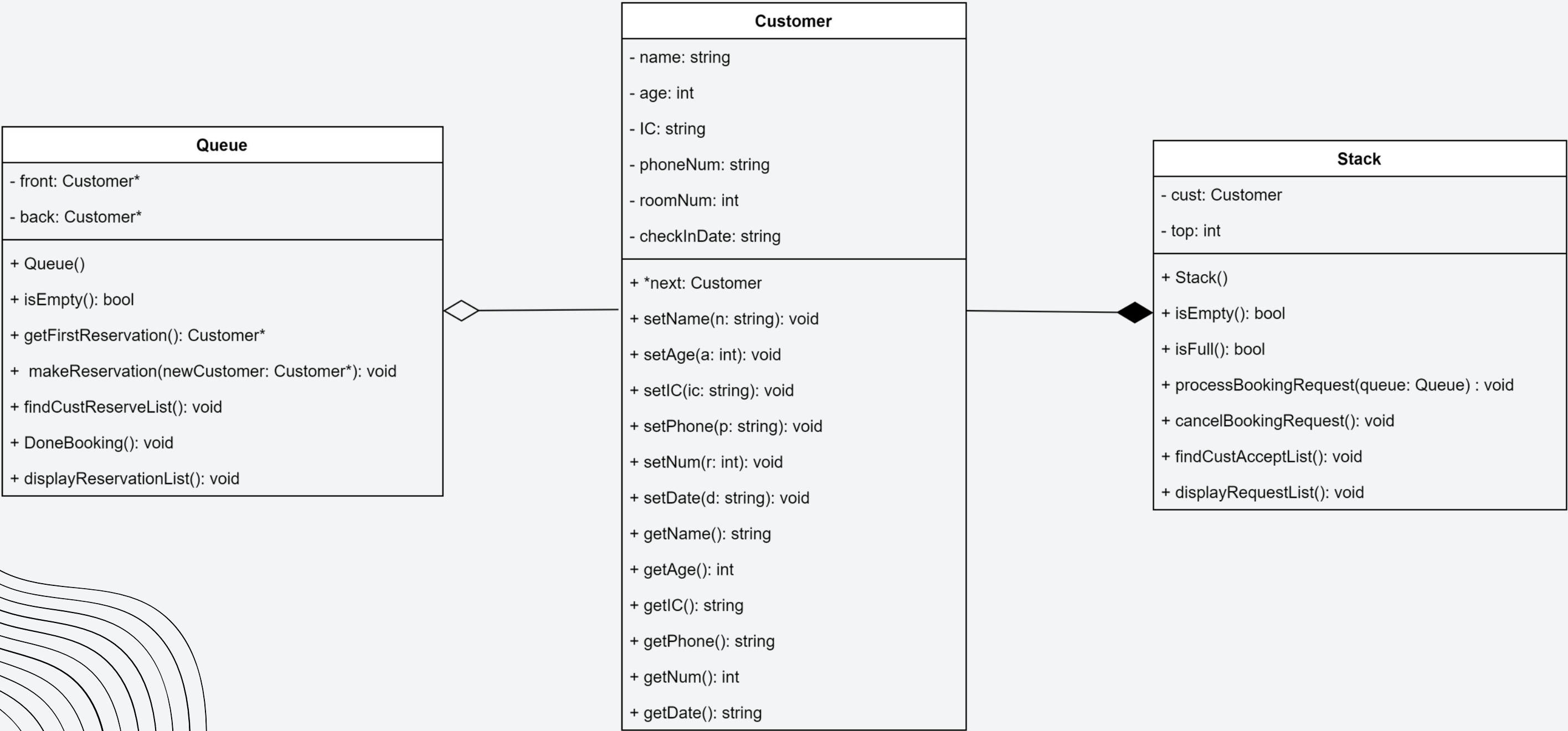
THE STACK CLASS HANDLES THE LIST OF ACCEPTED BOOKING REQUESTS, ALLOWING THE HOTEL MANAGER TO ACCEPT NEW REQUESTS, CANCEL BOOKINGS, FIND CUSTOMER DATA FROM THE ACCEPTED LIST, AND DISPLAY THE ACCEPTED LIST.

THE MAIN FUNCTION INITIALIZES CUSTOMER DATA FROM A FILE NAMED "PROJECT.TXT" AND PROVIDES A USER INTERFACE THAT ALLOWS INDIVIDUALS TO ACT AS EITHER AGENTS OR HOTEL MANAGERS. A LOOP ENSURES CONTINUOUS INTERACTION UNTIL THE USER CHOOSES TO EXIT, WITH FUNCTIONS BEING CALLED BASED ON USER CHOICES. THE PROGRAM EMPLOYS BOTH QUEUE AND STACK CLASSES TO MANAGE RESERVATIONS AND ACCEPT BOOKING REQUESTS.

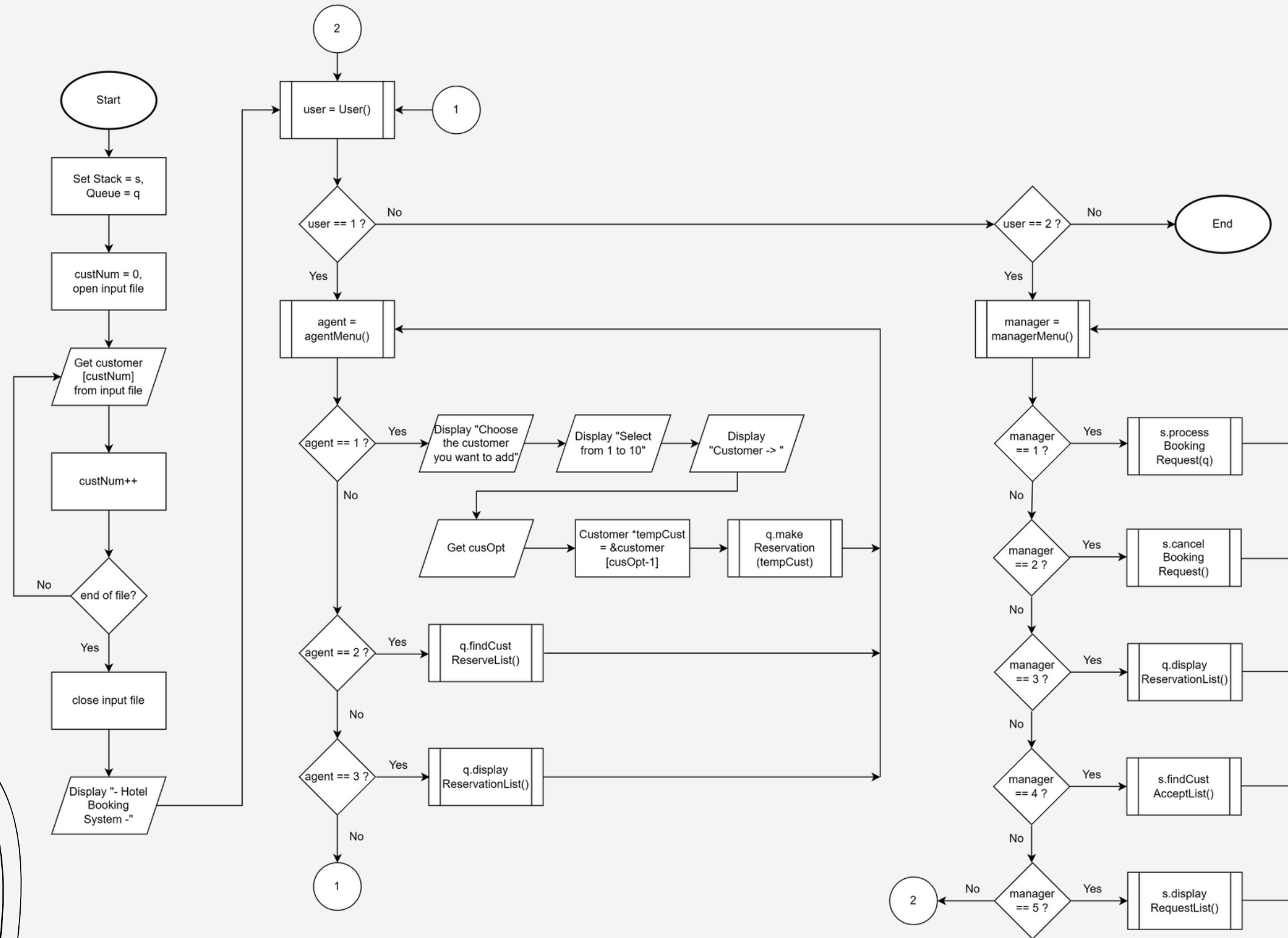


PROJECT DESIGN

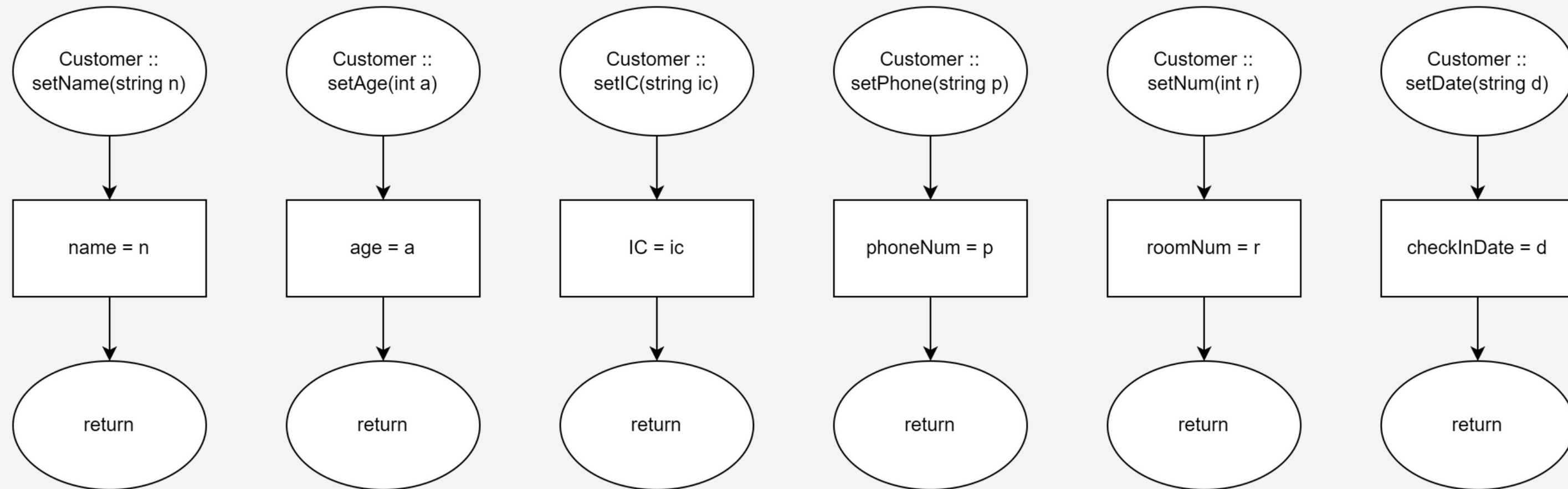
CLASS DIAGRAM



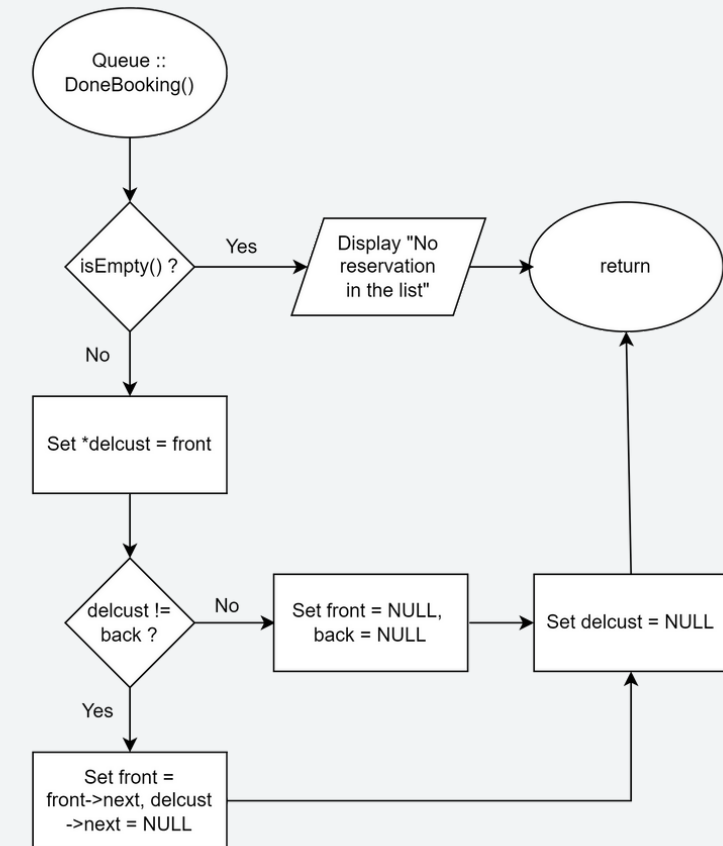
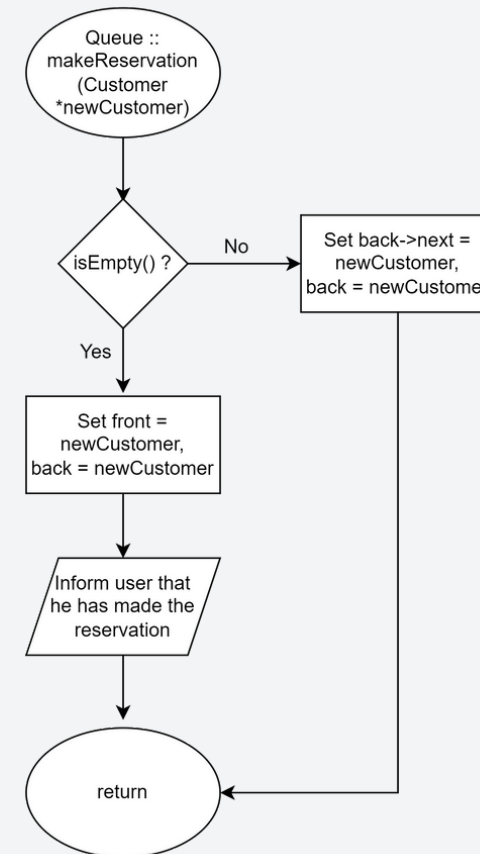
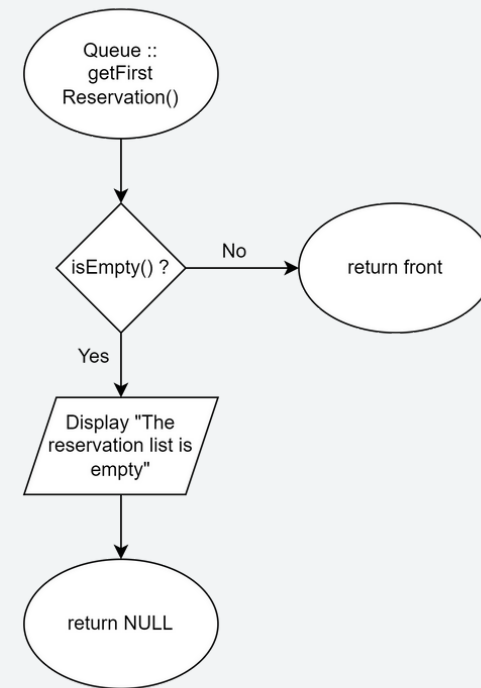
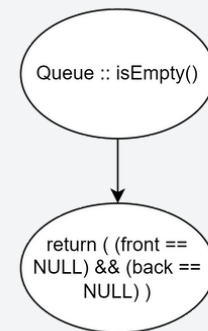
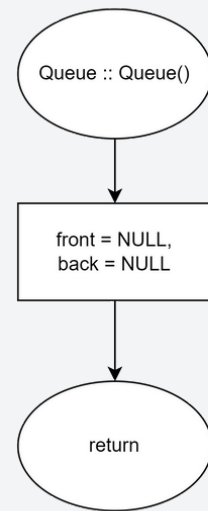
FLOW CHART : MAIN BODY



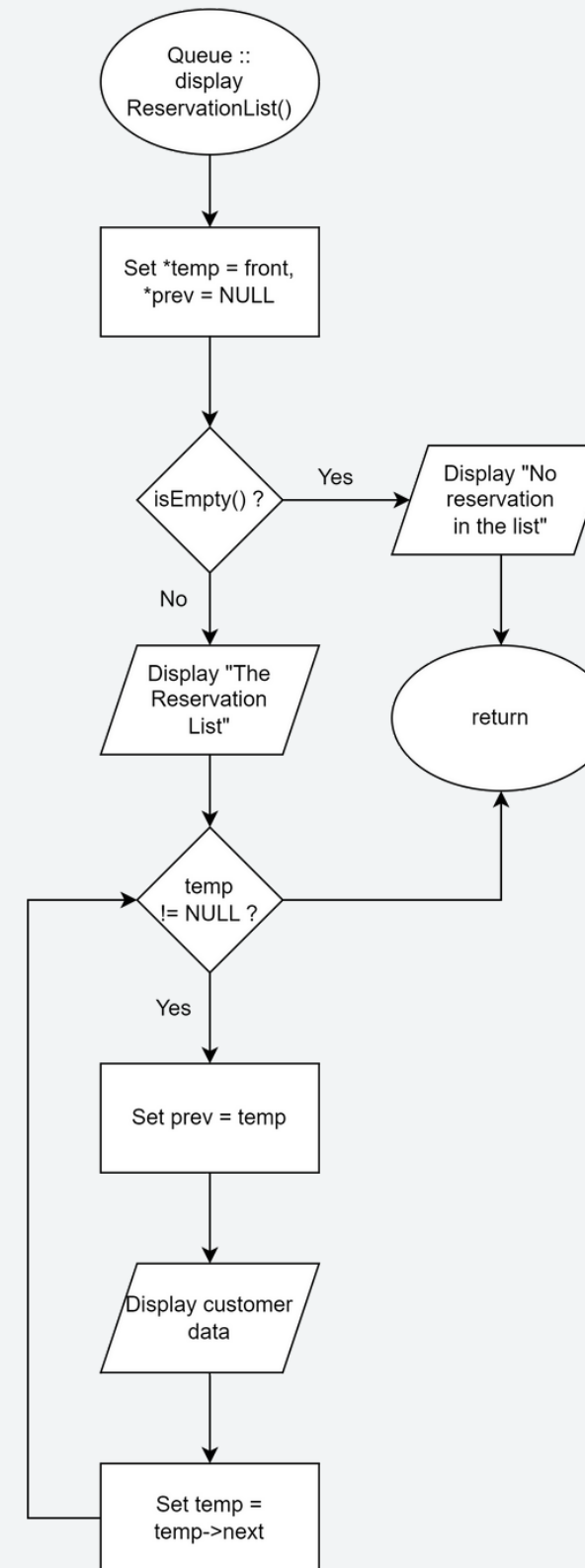
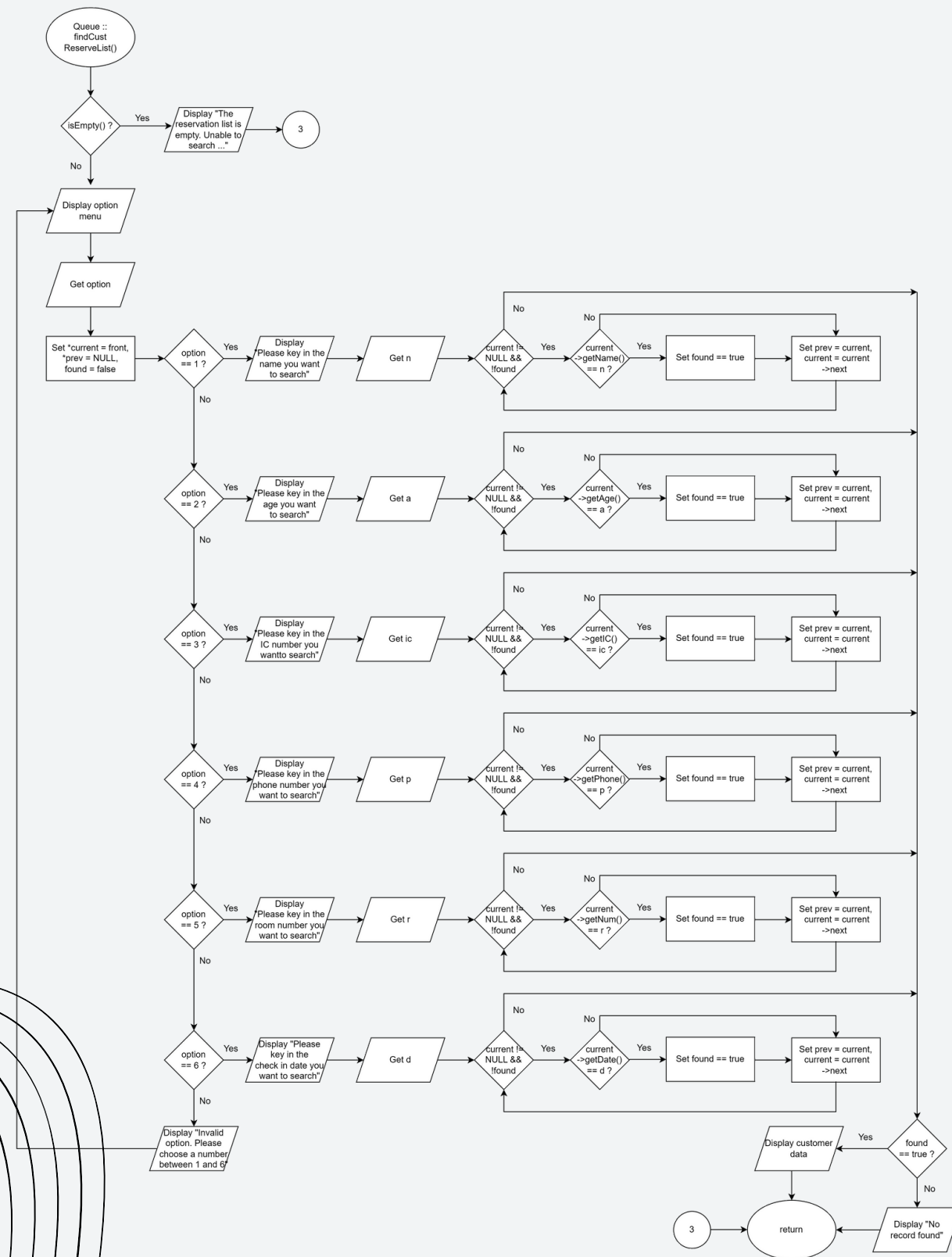
FLOW CHART : CLASS CUSTOMER



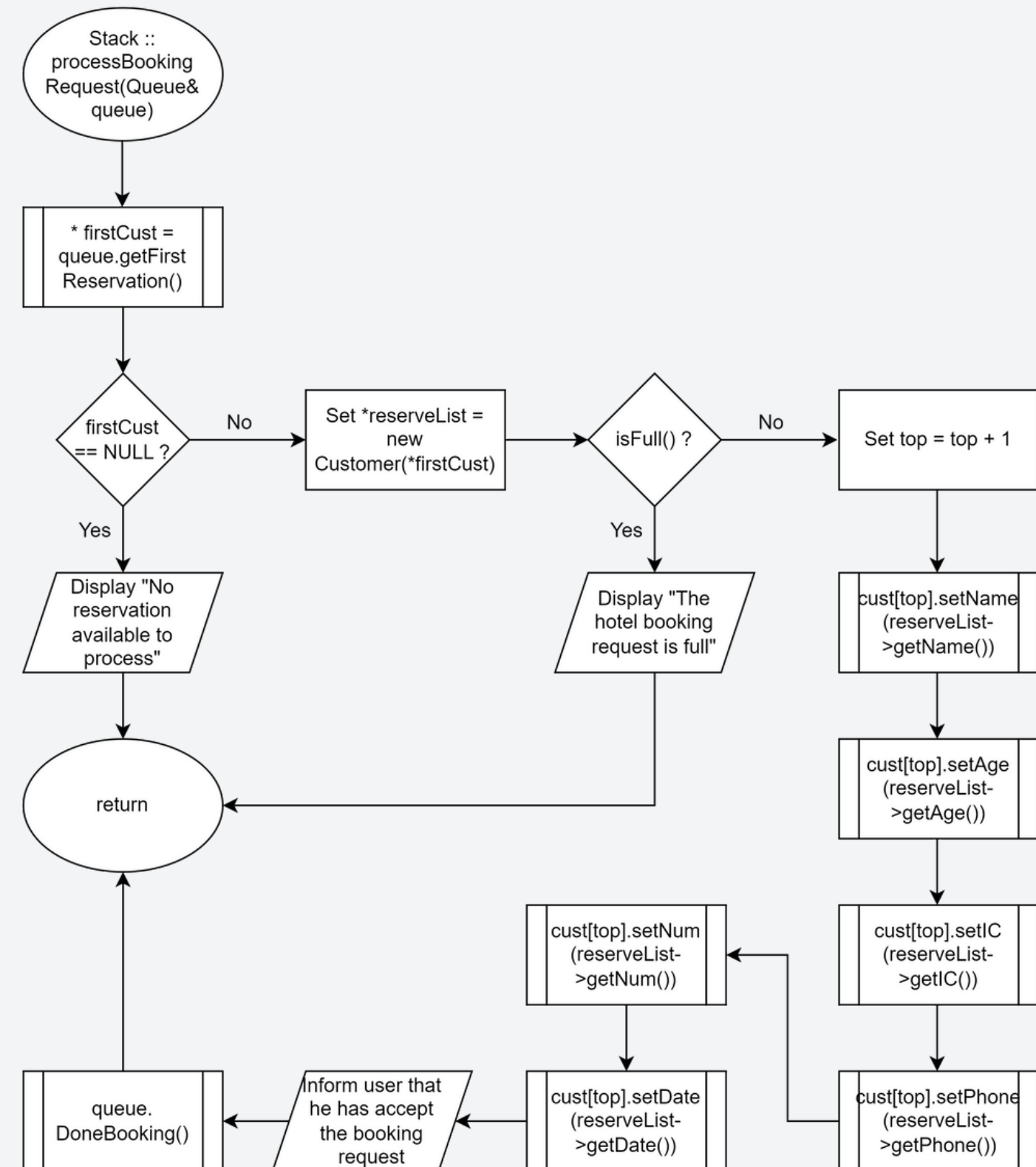
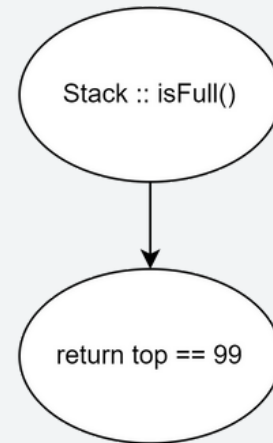
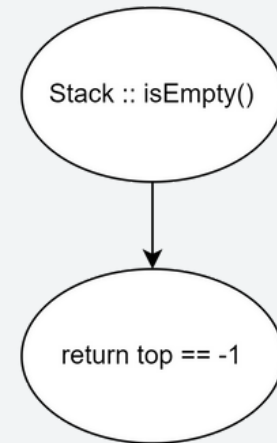
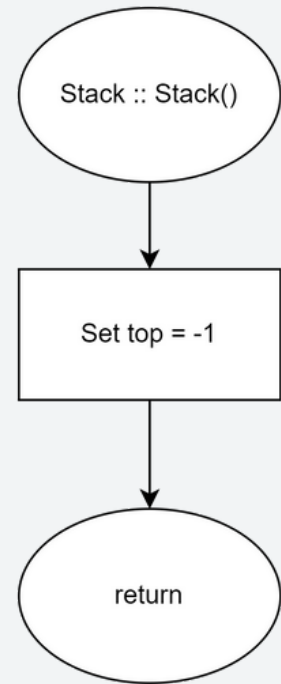
FLOW CHART : CLASS QUEUE



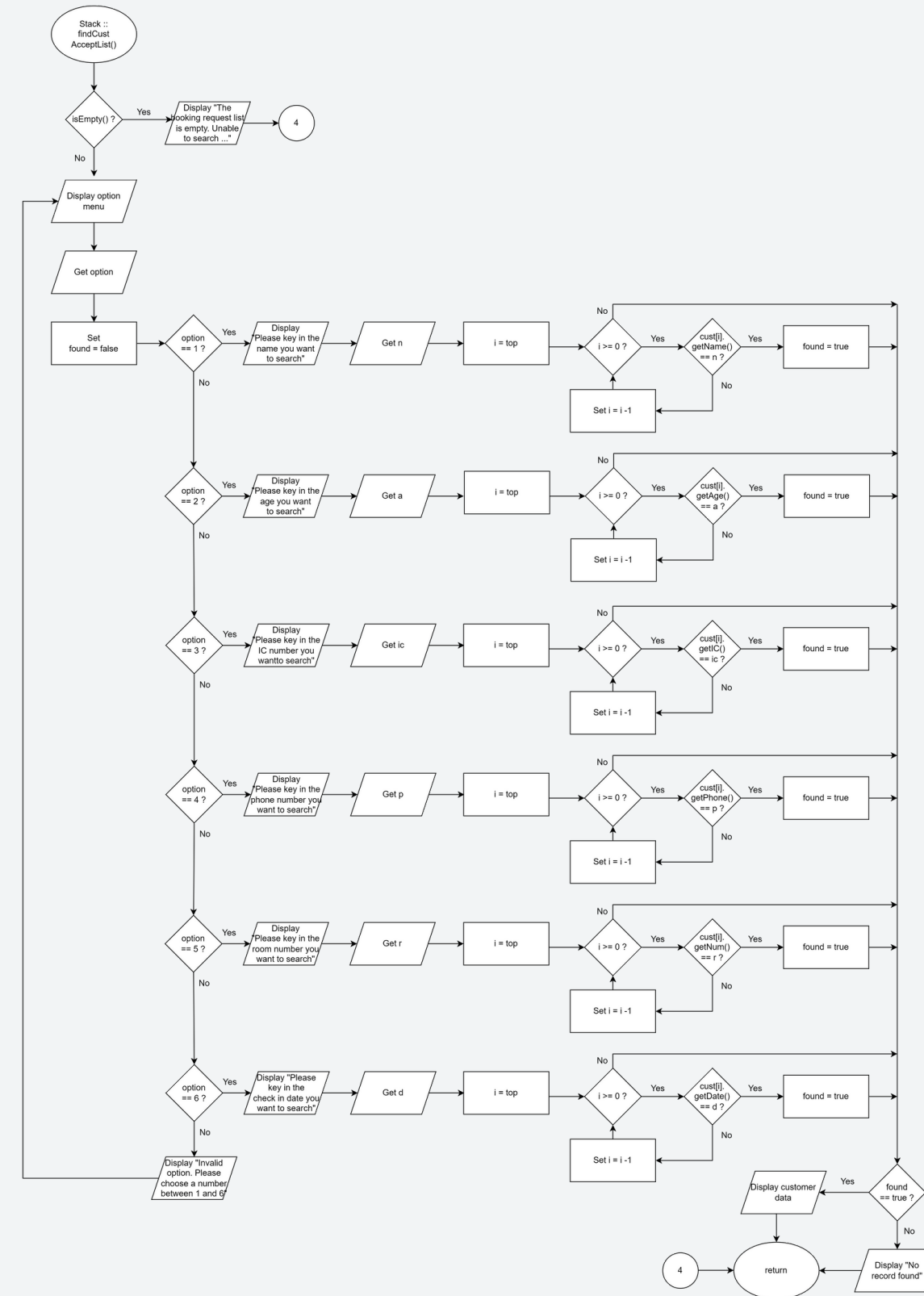
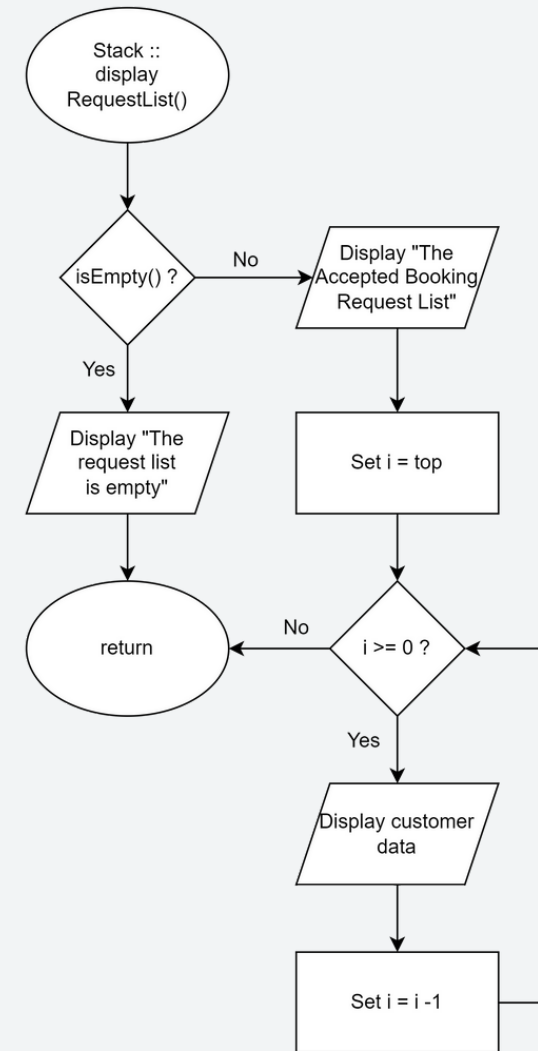
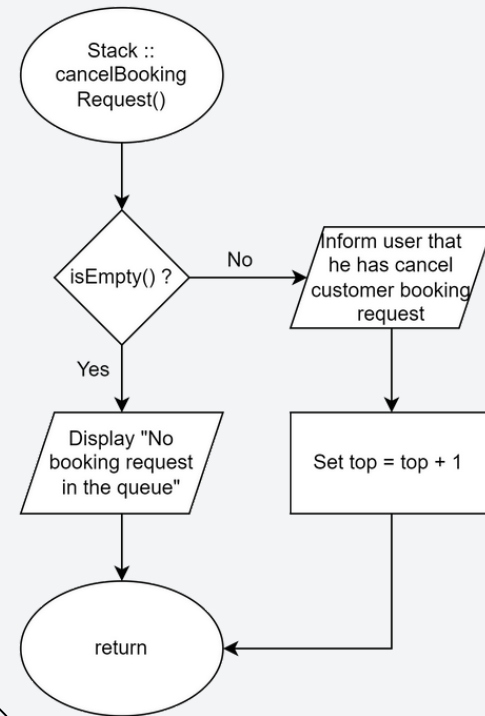
FLOW CHART : CLASS QUEUE



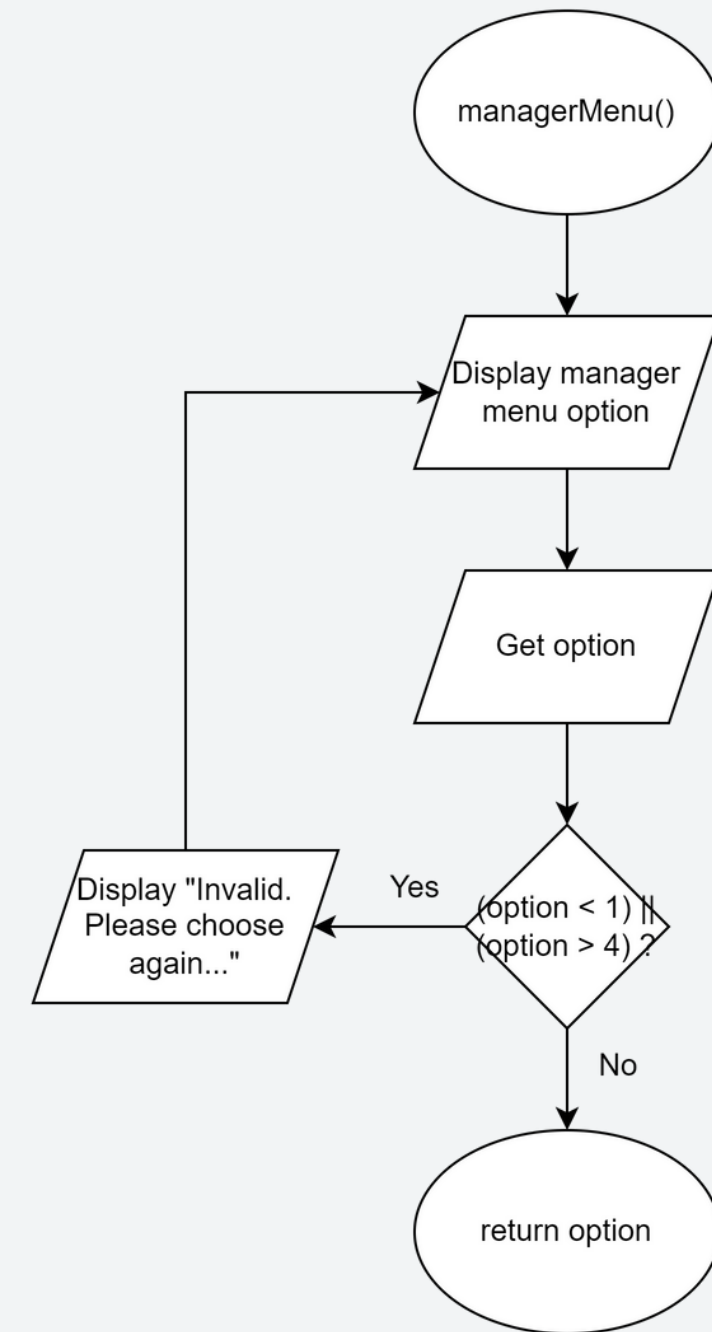
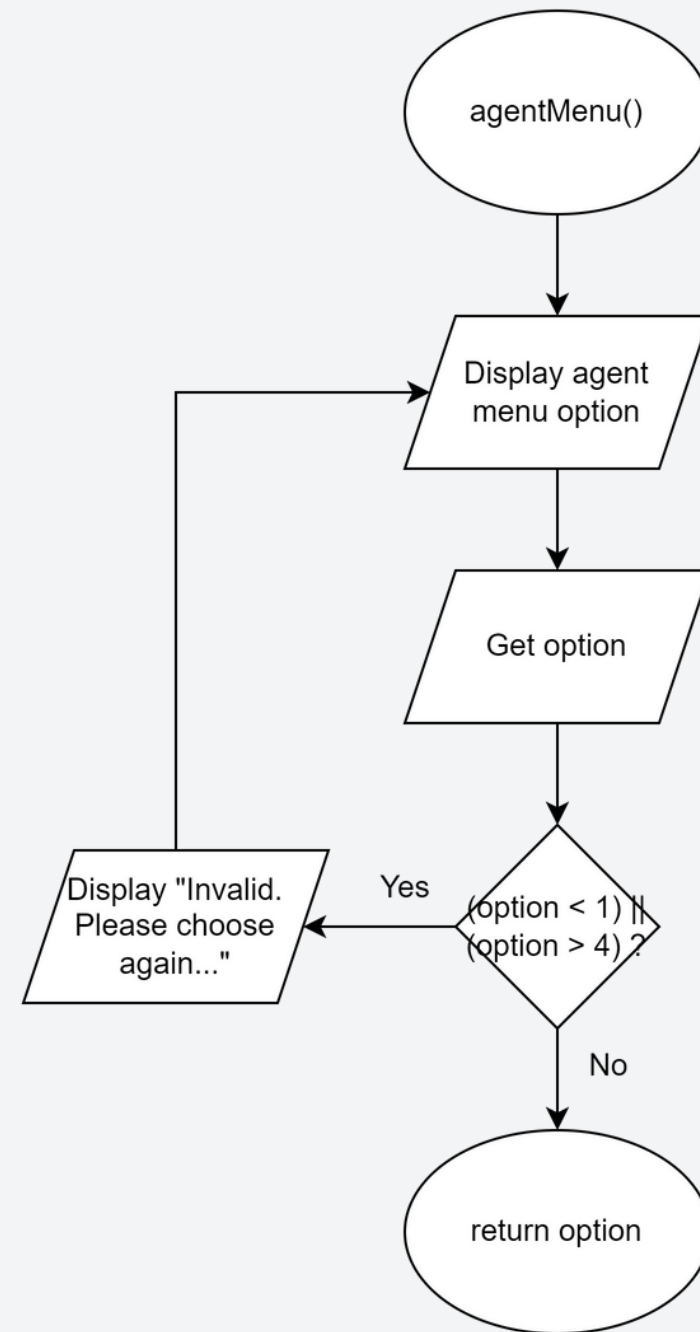
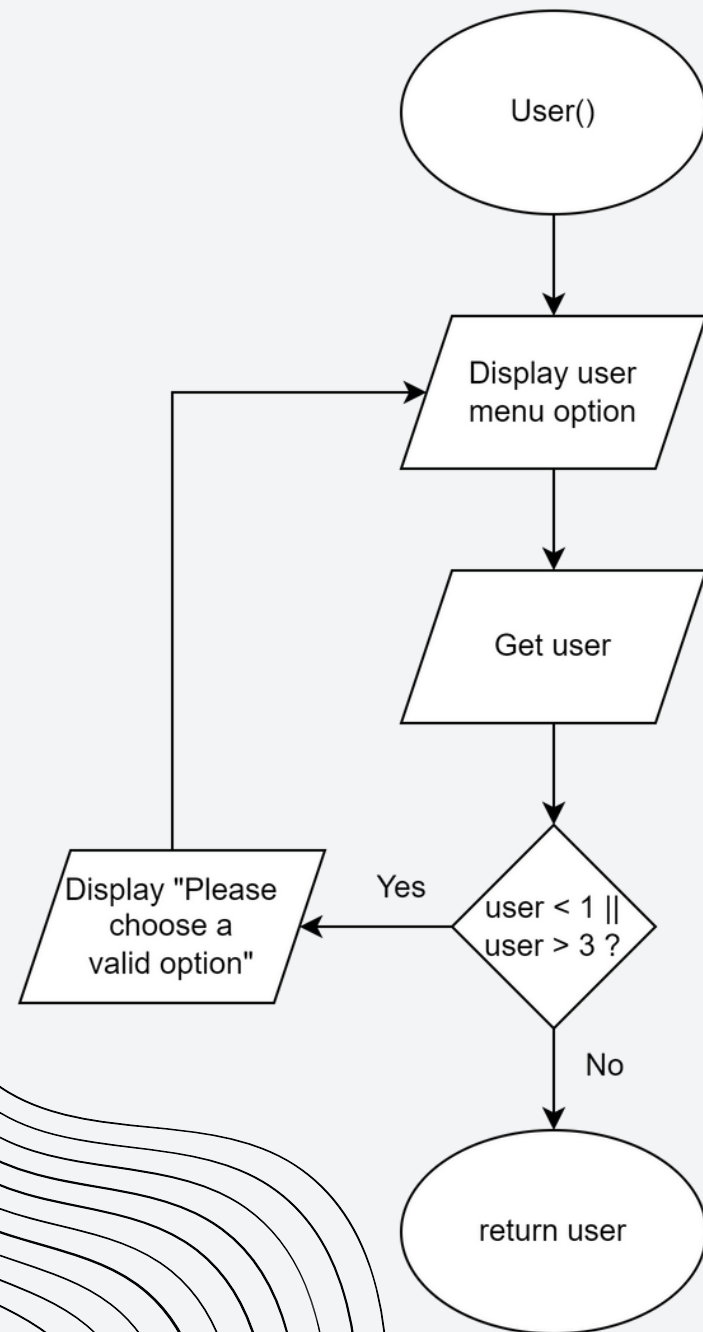
FLOW CHART : CLASS STACK



FLOW CHART : CLASS STACK





FLOW CHART : STAND ALONE FUNCTION





DATA STRUCTURE OF THE CONCEPTS EMPLOYED



QUEUE IMPLEMENTATION

1. QUEUE()

- IT ACT AS A CONSTRUCTOR FOR THE CLASS QUEUE
- WHEN WE DECLARE A CLASS QUEUE IN MAIN BODY, WE INITIALIZE THE *FRONT AND *BACK POINTER EQUAL TO NULL BECAUSE WE HAVEN'T INSERT ANY CUSTOMER DATA INSIDE THE CLASS

2. BOOL ISEMPY()

- THE FUNCTION IS USED TO DETECT WHETHER THERE IS CUSTOMER DATA INSIDE THE CLASS OR NOT
- WE DETECT THE EMPTINESS OF THE CLASS BY RETURNING ((FRONT == NULL) && (BACK == NULL))
- IF EITHER ONE OF THE POINTERS IS NULL, THE FUNCTION WILL RETURN TRUE
- THE FUNCTION WILL RETURN FALSE IF BOTH POINTERS ARE NOT NULL

3. CUSTOMER* GETFIRSTRESERVATION()

- THE FUNCTION IS USED TO GET THE FIRST DATA IN THE CLASS
- WE DETECT THE EMPTINESS OF CLASS USING THE ISEMPTY() FUNCTION
- IF THE CLASS DOES NOT CONTAIN ANY CUSTOMER DATA, IT WILL DISPLAY "THE RESERVATION LIST IS EMPTY"
- IF THE CLASS CONTAINS CUSTOMER DATA, IT WILL RETURN THE FIRST CUSTOMER DATA BY USING POINTER *FRONT

4.VOID MAKERESERVATION(CUSTOMER *NEWCUSTOMER)

- THE FUNCTION ACTS AS ENQUEUE() WHICH MEANS ENTER DATA INTO THE CLASS
- THE PARAMETER OF (CUSTOMER *NEWCUSTOMER) POINTS CUSTOMER DATA FROM THE MAIN BODY
- WE DETECT THE EMPTINESS OF CLASS USING THE ISEMPY() FUNCTION
- IF THE CLASS DOES NOT CONTAIN ANY CUSTOMER DATA, THE NEW CUSTOMER DATA WILL BE THE FIRST DATA INSERT IN THE CLASS BY POINTING *FRONT AND *BACK TO THE DATA
- IF THE CLASS CONTAINS CUSTOMER DATA, THE *NEXT POINTER (DECLARED IN CLASS CUSTOMER) OF THE *BACK POINTER WILL POINT TO THE NEW CUSTOMER WHICH MEANS THE NEW CUSTOMER DATA WILL BE ARRANGED AFTER THE LAST DATA INSIDE THE CLASS. THEN, THE *BACK POINTER WILL POINT TO THE NEW INSERTED DATA TO BECOME IT THE LAST DATA INSIDE THE CLASS

5. VOID FINDCUSTRESERVELIST()

- THE FUNCTION IS USED TO SEARCH THE CUSTOMER DATA OBTAINED IN THE CLASS BY SPECIFIC KEYWORD
- WE DETECT THE EMPTINESS OF CLASS USING THE ISEMPY() FUNCTION
- IF THE CLASS DOES NOT CONTAIN ANY CUSTOMER DATA, IT WILL DISPLAY "THE RESERVATION LIST IS EMPTY. UNABLE TO SEARCH ..." THEN RETURN TO MAIN BODY
- THE FUNCTION WILL DISPLAY OPTION MENU FOR USER (AGENT) AND GET THE OPTION FROM USER
- WE DECLARE POINTER *CURRENT EQUAL TO FRONT (POINT TO FIRST DATA), *PREV EQUAL TO NULL FOR PURPOSE LATER

- WE ALSO DECLARE A "BOOL FOUND = FALSE" TO CHECK THE PRESENCE OF CUSTOMER DATA INSIDE THE CLASS. "FALSE" MEANS THE DATA HAVEN'T FOUND YET
- ENTER THE SWITCH CASE, IF OPTION IS OTHER THAN 1 TO 6, IT WILL LOOPING TO DISPLAY OPTION MENU AGAIN AND GET USER OPTION
- IF USER KEY IN OPTION = 1, THE FUNCTION WILL DISPLAY "PLEASE KEY IN THE NAME YOU WANT TO SEARCH" AND GET THE N (NAME) KEY IN BY USER. THE LINE "WHILE((CURRENT != NULL) && (!FOUND))" IS USED FOR LOOPING IF HAVEN'T GOT THE SAME NAME INSIDE THE CLASS AS THE NAME INPUT BY THE USER. IT WILL CONTINUE LOOPING IF *CURRENT NOT EQUAL TO NULL AND FOUND = FALSE. THE POINTER *CURRENT WILL START FROM FIRST DATA INSIDE THE CLASS, IT WILL DETECT THE CURRENT POINTED DATA IS SAME NAME AS USER KEY IN OR NOT, IF SAME FOUND = TRUE, THEN *PREV WILL POINT TO *CURRENT TO OBTAIN THE DATA POINTED BY *CURRENT AND *CURRENT WILL POINT TO ITS NEXT DATA, AS FOUND = TRUE, THE LOOP WILL BREAK. IF NOT FOUND THE NAME SAME AS USER INPUT, THE FUNCTION WILL CONTINUE LOOPING UNTIL *CURRENT = NULL MEANS THE END OF DATA INSIDE THE CLASS, FOUND WILL NOT EQUAL TO TRUE AS ALL THE DATA NOT SAME AS USER INPUT

- FOR CASE JUST NOW, IF OBTAIN THE SAME NAME (FOUND == TRUE), THE FUNCTION WILL START TO DISPLAY CUSTOMER DATA USING *PREV POINTER. THE REASON OF USING *PREV INSTEAD OF USING *CURRENT IS BECAUSE WHILE *CURRENT POINTING NAME IS SAME AS USER INPUT, FOUND CHANGE TO TRUE BUT THE LOOP IS CONTINUED, *PREV HAVE EQUAL TO *CURRENT AND *CURRENT HAS POINTED TO THE NEXT DATA. THEREFORE, THE *PREV HAS POINTED TO THE NAME SAME AS USER INPUT AND *CURRENT HAS POINTED TO OTHER DATA. IF NOT OBTAIN THE SAME NAME WHICH MEANS FOUND REMAIN FALSE, THE FUNCTION WILL DISPLAY "NO RECORD FOUND"
- THE SAME CONCEPT IS APPLY TO OPTION = 2 TO 6 AND 2 IS FOR SEARCHING AGE, 3 FOR SEARCHING IC NUMBER, 4 FOR SEARCHING PHONE NUMBER, 5 FOR SEARCHING ROOM NUMBER AND 6 FOR SEARCHING CHECK IN DATE

6.VOID DONEBOOKING()

- THE FUNCTION ACTS AS DEQUEUE() WHICH MEANS DELETE THE FIRST DATA IN THE CLASS
- WE DETECT THE EMPTINESS OF CLASS USING THE ISEMPTY() FUNCTION
- IF THE CLASS DOES NOT CONTAIN ANY CUSTOMER DATA, THE FUNCTION WILL DISPLAY “NO RESERVATION IN THE LIST”
- IF THE CLASS CONTAINS CUSTOMER DATA, WE DECLARE A *DELCUST POINTER = *FRONT POINTER FOR PURPOSE LATER
- (DELCUST != BACK) MEANS THERE ARE MANY CUSTOMER DATA IN THE CLASS SINCE AT FIRST WE INSERT THE FIRST DATA BY POINT *FRONT AND *BACK EQUAL TO THE NEW INSERTED DATA AND TO INSERT AGAIN DATA *FRONT REMAIN AND *BACK POINT TO THE NEW INSERTED DATA.
- IF THE CLASS CONTAINS MORE THAN ONE CUSTOMER DATA, WE SET *FRONT POINTER POINT TO THE NEXT DATA INSIDE THE CLASS AND THE *NEXT POINTER OF THE *DELCUST POINT TO NULL TO BREAK THE LINE FOR *DELCUST POINTED CUSTOMER DATA BETWEEN THE QUEUE
- IF THE CLASS CONTAINS ONLY ONE CUSTOMER DATA, WE JUST SET *FRONT AND *BACK POINT TO NULL TO DELETE THE CUSTOMER DATA FROM THE QUEUE
- *DELCUST POINT TO NULL TO PREVENT THE SYSTEM CONFUSE WHICH CUSTOMER DATA IS IN THE QUEUE

7.VOID DISPLAYRESERVATIONLIST()

- WE DECLARE POINTER OF *TEMP = *FRONT AND *PREV = NULL FOR PURPOSE LATER
- WE DETECT THE EMPTINESS OF CLASS USING THE ISEMPTY() FUNCTION
- IF THE CLASS DOES NOT CONTAIN ANY CUSTOMER DATA, THE FUNCTION WILL DISPLAY "THE RESERVATION LIST IS EMPTY"
- IF THE CLASS CONTAINS CUSTOMER DATA, THE *PREV WILL POINT TO *TEMP POINTED DATA. THE CUSTOMER DATA WILL BE DISPLAY BY USING *PREV POINTER. AFTER DISPLAY THE CUSTOMER DATA, *TEMP WILL POINT TO THE NEXT DATA INSIDE THE QUEUE. THE PROCESS WILL CONTINUE UNTIL *TEMP IS POINT TO NULL WHICH MEANS THE END OF DATA INSIDE THE CLASS



STACK IMPLEMENTATION

1. STACK()

- IT ACT AS A CONSTRUCTOR FOR THE CLASS STACK
- WHEN WE DECLARE A CLASS STACK IN MAIN BODY, WE INITIALIZE THE TOP EQUAL TO -1 BECAUSE WE HAVEN'T INSERT ANY CUSTOMER DATA INSIDE THE CLASS

2. BOOL ISEMPY()

- THE FUNCTION IS USED TO DETECT WHETHER THERE IS CUSTOMER DATA INSIDE THE CLASS OR NOT
- WE DETECT THE EMPTINESS OF THE CLASS BY RETURNING (TOP == -1)
- IF TOP EQUAL TO -1, THE FUNCTION WILL RETURN TRUE
- THE FUNCTION WILL RETURN FALSE IF THE TOP IS A POSITIVE NUMBER OR 0

3. BOOL ISFULL()

- THE FUNCTION IS USED TO DETECT WHETHER THE CUSTOMER DATA INSIDE THE CLASS IS FILL OR NOT
- WE DETECT THE FULLNESS OF THE CLASS BY RETURNING (TOP == 99), THE REASON OF TOP == 99 INSTEAD OF TOP == 100 IS BECAUSE THE ARRAY STARTS FROM 0
- IF TOP EQUAL TO 99, THE FUNCTION WILL RETURN TRUE
- THE FUNCTION WILL RETURN FALSE IF THE TOP IS NOT EQUAL TO 99

4. VOID PROCESSBOOKINGREQUEST(Queue& Queue)

- THE FUNCTION ACTS AS PUSH() WHICH MEANS ENTER DATA INTO THE CLASS
- THE PARAMETER (Queue& Queue) IS USED AS WE WANT TO OBTAIN CUSTOMER DATA FROM CLASS Queue. "&" IS USED TO UPDATE THE DATA CHANGE FOR CLASS Queue
- WE DETECT THE FULLNESS OF CLASS USING THE ISFULL() FUNCTION, IF IT IS FULL, WE CANNOT INSERT ANY DATA AND THE SYSTEM WILL DISPLAY "THE HOTEL BOOKING REQUEST IS FULL"
- IF THE SYSTEM IS NOT FULL, THE VALUE OF TOP WILL INCREASE BY ONE FOR PURPOSE STORING THE CUSTOMER DATA INTO THE ARRAY AT ASSIGNED POSITION
- THE CUSTOMER DATA IS BEING INSERTED INSIDE THE ARRAY OF STACK VIA MUTATOR IN CLASS CUSTOMER
- THE Queue.DONEBOOKING() FUNCTION IS CALLED TO DELETE THE INSERTED DATA FROM CLASS Queue. IT MODIFY THE REAL SITUATION OF THE ACCEPTED BOOKING REQUEST WILL BE DELETE FROM THE RESERVATION LIST

5. VOID CANCELBOOKINGREQUEST()

- THE FUNCTION ACTS AS POP() WHICH MEANS POP OUT THE DATA FROM STACK
- THE ISEMPTY() IS USED TO DETECT THE EMPTINESS OF CLASS
- IF THE CLASS IS EMPTY, THE SYSTEM WILL DISPLAY “NO BOOKING REQUEST IN THE QUEUE” AND DO NOTHING
- IF THE CLASS HAS CUSTOMER DATA, THE FUNCTION WILL POP OUT THE LAST DATA IN THE STACK

6. VOID FINDCUSTACCEPTLIST()

- THE FUNCTION IS USED TO SEARCH THE CUSTOMER DATA OBTAINED IN THE CLASS BY SPECIFIC KEYWORD
- THE CONCEPT OF THIS FUNCTION IS SAME AS VOID FINDCUSTRESERVELIST() IN CLASS QUEUE JUST THE POINTER STYLE HAS CHANGE TO ARRAY STYLE

7. VOID DISPLAYREQUESTLIST()

- THE FUNCTION IS USED TO DISPLAY ALL THE CUSTOMER DATA IN THE CLASS STACK
- THE CONCEPT OF THIS FUNCTION IS SAME AS VOID DISPLAYRESERVATIONLIST() IN CLASS QUEUE JUST THE POINTER STYLE HAS CHANGE TO ARRAY STYLE



EXECUTION OF THE PROGRAM