CS Project 5 Report

Melody Chen

#705120273

One of the obstacle I faced during this project was when to use the :: operator. Initially, I wasn’t sure when to use it and when not to use it. But eventually, I realize that you use the :: operator when the function, variable, or object belongs to a class. I am still a bit confused, but I think with more practice I will be abe to understand how everything works.

Test Cases:

Ticket t("Pavillion Row 13", 45, "Dodgers vs. Brewers", "Dodger Stadium", 150.00, Ticket::KIND::ATHLETICGAME, "August 1, 7:30PM");

BoxOffice b;

assert(to\_string(t.getPrice()) == "150.000000"); //if getPrice of the ticket gives right price

assert(t.getKind() == Ticket::KIND::ATHLETICGAME); //if getKind gets the right kind

assert(t.getRow() == "Pavillion Row 13"); //if getRow gets the right row

assert(t.getSeat() == 45); //if getSeat gives right seat

assert(t.getEvent() == "Dodgers vs. Brewers"); //if getEvent gives the right event

assert(t.getVenue() == "Dodger Stadium"); //if getVenue gives right venue

assert(t.getDayTime() == "August 1, 7:30PM"); //if getDayTime gives right time

t.setPrice(2000);

assert(to\_string(t.getPrice()) == "2000.000000"); //see if setPrice works

t = Ticket("", 0, "", "", 0, Ticket::KIND::OTHER, ""); //test with empty strings

assert(to\_string(t.getPrice()) == "0.000000"); //tests if 0 works

assert(t.getKind() == Ticket::KIND::OTHER); //tests if kind gives the right kind

assert(t.getRow() == ""); //tests if empty string works for row

assert(t.getSeat() == 0); //tests if getSeat() gives you the right seat number

assert(t.getEvent() == ""); //tests if empty string works for event

assert(t.getVenue() == ""); //tests if empty string works for venue

assert(t.getDayTime() == ""); //test if empty string works for DayTime

t = b.buyTicket("Boelter 22", 345, "CS Lecture", "Boelter Hall", Ticket::KIND::MOVIE, "September 23, 6:35AM");

assert(to\_string(t.getPrice()) == "0.000000"); //tests if buyTicket presets ticket price to zero

assert(t.getKind() == Ticket::KIND::MOVIE); //if getKind gives right kind of movie

assert(t.getRow() == "Boelter 22"); //if getRow works correctly with ticket created by buyTicket

assert(t.getSeat() == 345); //if getSeat works correctly with ticket created by buyTicket

assert(t.getEvent() == "CS Lecture"); //if getEvent works correctly with ticket created by buyTicket

assert(t.getVenue() == "Boelter Hall"); //if getVenue works correctly with ticket created by buyTicket

assert(t.getDayTime() == "September 23, 6:35AM"); //if getDayTime works correctly with ticket created by buyTicket

t.setPrice(30);

assert(to\_string(t.getPrice()) == "30.000000"); //if price is no longer zero

t = b.buyRoyceHallStudentTicket("Orchestra Row Z", 80, "Pop Music Concert", Ticket::KIND::CONCERT, "September 1, 7:30AM");

//fails this one...she put 40

assert(std::to\_string(t.getPrice()) == "70.000000"); //if getPrice gets the correct price for student morning orchestra seats ticket

assert(t.getKind() == Ticket::KIND::CONCERT); //if getKind gives the right kind

assert(t.getRow() == "Orchestra Row Z"); //if getRow gives the right row

assert(t.getSeat() == 80); //if seat number is correct

assert(t.getDayTime() == "September 1, 7:30AM"); //if time is correct

t = b.buyRoyceHallStudentTicket("Balcony Row Z", 80, "Pop Music Concert", Ticket::KIND::CONCERT, "September 1, 7:30PM");

assert(std::to\_string(t.getPrice()) == "35.000000"); //if price is accurate for concert balcony pm student ticket

t = b.buyRoyceHallStudentTicket("Balcony Row Z", 80, "Pop Music Concert", Ticket::KIND::MOVIE, "September 1, 7:30PM");

assert(std::to\_string(t.getPrice()) == "10.000000"); //if price is accurate for movie balcony pm student ticket

t = b.buyRoyceHallStudentTicket("Orchestra Row Z", 80, "Pop Music Concert", Ticket::KIND::MOVIE, "September 1, 7:30PM");

assert(std::to\_string(t.getPrice()) == "20.000000"); //if price is accurate for movie orchestra pm student ticket

t = b.buyRoyceHallStudentTicket("Orchestra Row Z", 80, "Pop Music Concert", Ticket::KIND::ATHLETICGAME, "September 1, 7:30PM");

//should be 120, not 90

assert(std::to\_string(t.getPrice()) == "120.000000"); //if price is accurate for athletic orchestra pm student ticket

t = b.buyRoyceHallStudentTicket("Balcony Row Z", 80, "Pop Music Concert", Ticket::KIND::ATHLETICGAME, "September 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "45.000000"); //if price is accurate for athletic balcony am student ticket

t = b.buyRoyceHallStudentTicket("Orchestra Row Z", 80, "Pop Music Concert", Ticket::KIND::OTHER, "September 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "50.000000"); //if price is accurate for other orchestra am student ticket

t = b.buyRoyceHallStudentTicket("Balcony Row Z", 80, "Pop Music Concert", Ticket::KIND::OTHER, "September 1, 7:30PM");

assert(std::to\_string(t.getPrice()) == "25.000000"); //if price is accurate for other orchestra am student ticket

t = b.buyRoyceHallTicket("Orchestra Row Z", 80, "Pop Music Concert", Ticket::KIND::CONCERT, "September 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "100.000000"); //if getPrice gets the right price for concert orchestra morning reg. tickets

t.setPrice(50);

assert(std::to\_string(t.getPrice()) == "50.000000"); //if setPrice works with ticket created by buyRoyceHallTicket

t = b.buyRoyceHallTicket("Balcony Row Z", 80, "Pop Music Concert", Ticket::KIND::CONCERT, "September 1, 7:30PM");

assert(std::to\_string(t.getPrice()) == "65.000000"); //if getPrice gets the right price for concert orchestra night reg. tickets

t = b.buyRoyceHallTicket("Balcony Row Z", 80, "Tangled", Ticket::KIND::MOVIE, "September 1, 7:30PM");

assert(std::to\_string(t.getPrice()) == "12.500000"); //if price is correctly calcualted for reg. night time movie

t = b.buyRoyceHallTicket("Orchestra Row Z", 80, "Tangled", Ticket::KIND::MOVIE, "September 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "25.000000"); //if price is correctly calcualted for reg. am time movie

t = b.buyRoyceHallTicket("Balcony Row Z", 80, "Gymnastics", Ticket::KIND::ATHLETICGAME, "September 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "75.000000"); //if price is correctly calcualted for reg. am time athletic game

t = b.buyRoyceHallTicket("Orchestra Row Z", 80, "Gymnastics", Ticket::KIND::ATHLETICGAME, "September 1, 7:30PM");

assert(std::to\_string(t.getPrice()) == "150.000000"); //if price is correctly calcualted for reg. am time athletic game

t = b.buyRoyceHallTicket("Orchestra Row Z", 80, "Ling Class", Ticket::KIND::OTHER, "September 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "80.000000"); //if getPrice gets the right price for other orchestra morning reg. tickets

t = b.buyRoyceHallTicket("Balcony Row Z", 80, "Ling Class", Ticket::KIND::OTHER, "September 1, 7:30PM");

assert(std::to\_string(t.getPrice()) == "55.000000"); //if getPrice gets the right price for other orchestra morning reg. tickets

t = b.buyRoyceHallStaffTicket("Balcony Row J", 120, "An Evening With David Sedaris", Ticket::KIND::OTHER, "October 1, 7:30PM");

assert(std::to\_string(t.getPrice()) == "35.000000"); //if price is accurate for staff other night ticket

assert(t.getKind() == Ticket::KIND::OTHER); //if ticket created with buyRoyceHallStaffTicket returns the right kind

assert(t.getRow() == "Balcony Row J");  //if ticket created with buyRoyceHallStaffTicket returns the right row

assert(t.getSeat() == 120);  //if ticket created with buyRoyceHallStaffTicket returns the right seat

t = b.buyRoyceHallStaffTicket("Orchestra Row J", 120, "An Evening With David Sedaris", Ticket::KIND::OTHER, "October 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "60.000000"); //if price is accurate for other orchestra am staff ticket

t = b.buyRoyceHallStaffTicket("Orchestra Row J", 120, "An Evening With David Sedaris", Ticket::KIND::MOVIE, "October 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "22.500000");  //if price is accurate for movie orchestra am staff ticket

t = b.buyRoyceHallStaffTicket("Balcony Row J", 120, "An Evening With David Sedaris", Ticket::KIND::MOVIE, "October 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "11.250000");  //if price is accurate for movie balcony am staff ticket

t = b.buyRoyceHallStaffTicket("Balcony Row J", 120, "An Evening With David Sedaris", Ticket::KIND::ATHLETICGAME, "October 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "55.000000");  //if price is accurate for athletic balcony am staff ticket

t = b.buyRoyceHallStaffTicket("Orchestra Row J", 120, "An Evening With David Sedaris", Ticket::KIND::ATHLETICGAME, "October 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "130.000000");  //if price is accurate for athletic orchestra am staff ticket

t = b.buyRoyceHallStaffTicket("Orchestra Row J", 120, "An Evening With David Sedaris", Ticket::KIND::CONCERT, "October 1, 7:30AM");

assert(std::to\_string(t.getPrice()) == "80.000000");//if price is accurate for concert orchestra am staff ticket

t = b.buyRoyceHallStaffTicket("Balcony Row J", 120, "An Evening With David Sedaris", Ticket::KIND::CONCERT, "October 1, 7:30PM");

assert(std::to\_string(t.getPrice()) == "45.000000");//if price is accurate for concert balcony pm staff ticket