Name : Avnish Sengupta

UCLA ID: 705299746

Notable Obstacles Overcome:

* Since the quickPick function assigned the random values and returned the ticket behind-the-scenes, it was hard to see if it was doing what I wanted it to do. So, I created a long assert statement that would check if the values assigned to the first five balls were unequal and ran it under a loop about 20 times. Following which, I added diagnostic print statements in the quickpick functions to see what ball values were being assigned for it.
* To ensure that the first five ball values are unequal, I made a lot of while statements that would keep running if the values were equal and would call the random function again until a dissimilar value was attained.
* Keeping a track of the extent of tallying between the lottery and the ticket in the checkTicket function. I did it by using two nested for loops that checked each array and used a guarded action to see if the values were equal, then a counter variable would increment by 1. I used the value returned by the counter variable to assign the appropriate enum statements.
* I came across a warning that stated that the MegaMillionsLottery::WinningPossibility class was unscoped and so it would prefer enum class over enum when I tried to initialize a variable to store my appropriate enum value in based on the extent of tally. So, while debugging it would always skip over the line but it would working fine so I didn’t change it really.

Data Testing : -

Here is a list of data that could be used to thoroughly test my program: -

Initializations: -

MegaMillionsTicket t (1, 10, 9, 8, 7, 6); //to check the accessor functions for the MegaMillionsTicket class

MegaMillionsTicket s(1, 2, 3, 4, 5, 6); //to use in the upcoming tests.

MegaMillionsLottery l (1, 2, 3, 4, 5, 6); //to check the accessor functions for the MegaMillionsLottery class

MegaMillionsTicket q(44, 14, 28, 65, 72, 16); //to check the accessor functions for the MegaMillionsTicket class

MegaMillionsLottery w(75, 74, 11, 42, 67, 21); // to check the accessor functions for the MegaMillionsLottery class

assert(q.getBall1() == 44); //checking the MegaMillionsTicket accessor functions for random values.

assert(q.getBall2() == 14); //checking the MegaMillionsTicket accessor functions for random values.

assert(q.getBall3() == 28); //checking the MegaMillionsTicket accessor functions for random values.

assert(q.getBall4() == 65); //checking the MegaMillionsTicket accessor functions for random values.

assert(q.getBall5() == 72); //checking the MegaMillionsTicket accessor functions for random values.

assert(q.getMegaBall() == 16); //checking the MegaMillionsTicket accessor functions for random values.

assert(w.getBall1() == 75); //checking the MegaMillionsTicket accessor functions for random values.

assert(w.getBall2() == 74); //checking the MegaMillionsTicket accessor functions for random values.

assert(w.getBall3() == 1); //checking the MegaMillionsTicket accessor functions for random values.

assert(w.getBall4() == 2); //checking the MegaMillionsTicket accessor functions for random values.

assert(w.getBall5() == 37); //checking the MegaMillionsTicket accessor functions for random values.

assert(w.getMegaBall() == 21); //checking the MegaMillionsTicket accessor functions for random values.

assert(l.getBall1() == 1); //checking the MegaMillionsTicket accessor functions for random values.

assert(l.getBall2() == 2); //checking the MegaMillionsTicket accessor functions for random values.

assert(l.getBall3() == 3); //checking the MegaMillionsTicket accessor functions for random values.

assert(l.getBall4() == 4); //checking the MegaMillionsTicket accessor functions for random values.

assert(l.getBall5() == 5); //checking the MegaMillionsTicket accessor functions for random values.

assert(l.getMegaBall() == 6); //checking the MegaMillionsTicket accessor functions for random values.

assert(t.getBall1() == 1); //checking the MegaMillionsTicket accessor functions for random values.

assert(t.getBall2() == 10); //checking the MegaMillionsTicket accessor functions for random values.

assert(t.getBall3() == 9); //checking the MegaMillionsTicket accessor functions for random values.

assert(t.getBall4() == 8); //checking the MegaMillionsTicket accessor functions for random values.

assert(t.getBall5() == 7); //checking the MegaMillionsTicket accessor functions for random values.

assert(t.getMegaBall() == 6); //checking the MegaMillionsTicket accessor functions for random values.

//to check if the enum outcomes return the correct value when CheckTicket and whatHappened returns

the corresponding correct value when a ticket when 5 balls and the megaball tally.

assert(l.checkTicket(s) == MegaMillionsLottery::WinningPossibility:: FIVEPLUSMEGABALL);

assert(l.whatHappened(s) == "You matched 5 balls plus the megaball!");

//to check if the enum outcomes return the correct value when CheckTicket and whatHappened returns

the corresponding correct value when a ticket when 4 balls and the megaball tally.

MegaMillionsLottery l1(63, 71, 44, 25, 60, 37);

ticket = MegaMillionsTicket(63, 2, 44, 25, 60, 37);

assert(l1.checkTicket(ticket) == MegaMillionsLottery::WinningPossibility::FOURPLUSMEGABALL);

assert(l1.whatHappened(ticket) == "You matched 4 balls plus the megaball!");

//to check if the enum outcomes return the correct value when CheckTicket and whatHappened returns

the corresponding correct value when a ticket when 4 balls tally.

ticket = MegaMillionsTicket(1, 2, 3, 4, 26, 33);

assert(l.checkTicket(ticket) == MegaMillionsLottery::WinningPossibility::FOUR);

assert(l.whatHappened(ticket) == "You matched 4 balls!");

//to check if the enum outcomes return the correct value when CheckTicket and whatHappened returns

the corresponding correct value when a ticket when 3 balls and the megaball tally.

MegaMillionsLottery l3(63, 71, 44, 25, 49, 37);

ticket = MegaMillionsTicket(63, 2, 44, 25, 15, 37);

assert(l3.checkTicket(ticket) == MegaMillionsLottery::WinningPossibility::THREEPLUSMEGABALL);

assert(l3.whatHappened(ticket) == "You matched 3 balls plus the megaball!");

//to check if the enum outcomes return the correct value when CheckTicket and whatHappened returns

the corresponding correct value when a ticket when 3 balls tally.

MegaMillionsLottery l4(63, 71, 44, 25, 49, 26);

ticket = MegaMillionsTicket(63, 2, 44, 25, 15, 4);

assert(l4.checkTicket(ticket) == MegaMillionsLottery::WinningPossibility::THREE);

assert(l4.whatHappened(ticket) == "You matched 3 balls!");

//to check if the enum outcomes return the correct value when CheckTicket and whatHappened returns

the corresponding correct value when a ticket when 2 balls and the megaball tally.

MegaMillionsLottery lottery5(15, 44, 3, 65, 49, 7);

ticket = MegaMillionsTicket(74, 2, 3, 24, 49, 7);

assert(l5.checkTicket(ticket) == MegaMillionsLottery::WinningPossibility::TWOPLUSMEGABALL);

assert(l5.whatHappened(ticket) == "You matched 2 balls plus the megaball!");

//to check if the enum outcomes return the correct value when CheckTicket and whatHappened returns

the corresponding correct value when a ticket when 1 ball and the megaball tally.

MegaMillionsLottery l6(25, 34, 45, 56, 67, 5);

ticket = MegaMillionsTicket(74, 2, 33, 14, 67, 5);

assert(l6.checkTicket(ticket) == MegaMillionsLottery::WinningPossibility::ONEPLUSMEGABALL);

assert(l6.whatHappened(ticket) == "You matched 1 ball plus the megaball!");

//to check if the enum outcomes return the correct value when CheckTicket and whatHappened returns

the corresponding correct value when a ticket when the megaball tallies.

MegaMillionsLottery l7(12, 23, 34, 45, 56, 24);

ticket = MegaMillionsTicket(47, 2, 66, 32, 15, 24);

assert(l7.checkTicket(ticket) == MegaMillionsLottery::WinningPossibility::MEGABALL);

assert(l7.whatHappened(ticket) == "You matched the megaball!");

//to check if the enum outcomes return the correct value when CheckTicket and whatHappened returns

the corresponding correct value when a ticket when 3 balls and the megaball tally and when the

corresponding tallying values are not in order.

MegaMillionsLottery l8(12, 54, 66, 75, 49, 3);

ticket = MegaMillionsTicket(66, 75, 66, 54, 34, 3);

assert(l8.checkTicket(ticket) == MegaMillionsLottery::WinningPossibility::THREEPLUSMEGABALL);

assert(l8.whatHappened(ticket) == "You matched 3 balls plus the megaball!");

//to check if the enum outcomes return the correct value when CheckTicket and whatHappened returns

the corresponding correct value when a ticket when nothing tallies.

MegaMillionsLottery l2(12, 23, 34, 45, 56, 22);

ticket = MegaMillionsTicket(1, 2, 3, 4, 15, 10);

assert(l2.checkTicket(ticket) == MegaMillionsLottery::WinningPossibility::NOTWINNING);

assert(l2.whatHappened(ticket) == "You didn't win anything at all!");

//to check that quickPick always returns dissimilar ball values in the ticket for multiple trials. (This code

would intermittently fail after 10 or so tries when 100 trials were run.)

MegaMillionsTicket quickPickTicket(1, 2, 3, 4, 5, 6);

for (int i = 0; i < 20; i++)

{

quickPickTicket = l.quickPick();

assert(quickPickTicket.getBall1() != quickPickTicket.getBall2() &&

quickPickTicket.getBall1() != quickPickTicket.getBall3() &&

quickPickTicket.getBall1() != quickPickTicket.getBall4() &&

quickPickTicket.getBall1() != quickPickTicket.getBall5() &&

quickPickTicket.getBall2() != quickPickTicket.getBall3() &&

quickPickTicket.getBall2() != quickPickTicket.getBall4() &&

quickPickTicket.getBall2() != quickPickTicket.getBall5() &&

quickPickTicket.getBall3() != quickPickTicket.getBall4() &&

quickPickTicket.getBall3() != quickPickTicket.getBall5() &&

quickPickTicket.getBall4() != quickPickTicket.getBall5());

}

}