Ethan Duong

UID: 805124044

Notable Obstacles:

I had difficulties managing multiple compilation. Understanding the concepts behind the header file guards was also difficult. I have had experience with object oriented methods before so the concept of classes and structs came easily to me.

Example Test Data:

Auction auction("4th Generation iPod", 50.00);

EBayBucks bucks;

//This first set of tests test if the auction class members are initialized correctly

assert(auction.numberOfBids() == 0);

assert(auction.wasSuccessful() == false);

assert(auction.getDescription() == "4th Generation iPod");

assert(std::to\_string(auction.winningBid()) == "-1.000000");

//This next set tests that bids don't count if the auction isn't open...

assert(auction.bid(100.00) == false);

assert(auction.bid(200.00) == false);

assert(auction.bid(300.00) == false);

assert(auction.numberOfBids() == 0);

assert(auction.wasSuccessful() == false);

assert(std::to\_string(auction.winningBid()) == "-1.000000");

auction.openAuction();

//This next set tests that auctions are not successful if the bids are too low...

assert(auction.bid(-50.00) == false); // negative bids are ignored

assert(auction.bid(1.00) == true);

assert(auction.bid(2.00) == true);

assert(auction.bid(3.00) == true);

assert(auction.bid(1.50) == false);//bids lower than the currentBid value are ignored

assert(auction.numberOfBids() == 3);

assert(auction.wasSuccessful() == false);

assert(std::to\_string(auction.winningBid()) == "-1.000000");

//Tests that auctions must be closed to be deemed successful

assert(auction.bid(100.00) == true);

assert(auction.numberOfBids() == 4);

assert(auction.wasSuccessful() == false);

auction.closeAuction();

assert(auction.wasSuccessful() == true);

assert(std::to\_string(auction.winningBid()) == "100.000000");

//Tests that bids don't count if the auction is closed...

assert(auction.bid(500.00) == false);

assert(auction.bid(600.00) == false);

assert(auction.numberOfBids() == 4);

//Tests that earnings class members are initialized correctly and that earnings are // distributed appropriately

assert(std::to\_string(bucks.earnings()) == "0.000000");

assert(std::to\_string(bucks.issueCertificate()) == "0.000000");

bucks.addAuction(auction);

assert(std::to\_string(bucks.earnings()) == "1.000000");

assert(std::to\_string(bucks.issueCertificate()) == "1.000000");

// once a certificate is issued, the earned award gets reset to zero

assert(std::to\_string(bucks.earnings()) == "0.000000");

Auction a("unsuccessful", 10000.00);

a.openAuction();

assert(a.bid(100.00) == true);

a.closeAuction();

assert(a.wasSuccessful() == false);

// unsuccessful auctions don't raise the earned award

bucks.addAuction(a);

assert(std::to\_string(bucks.earnings()) == "0.000000");