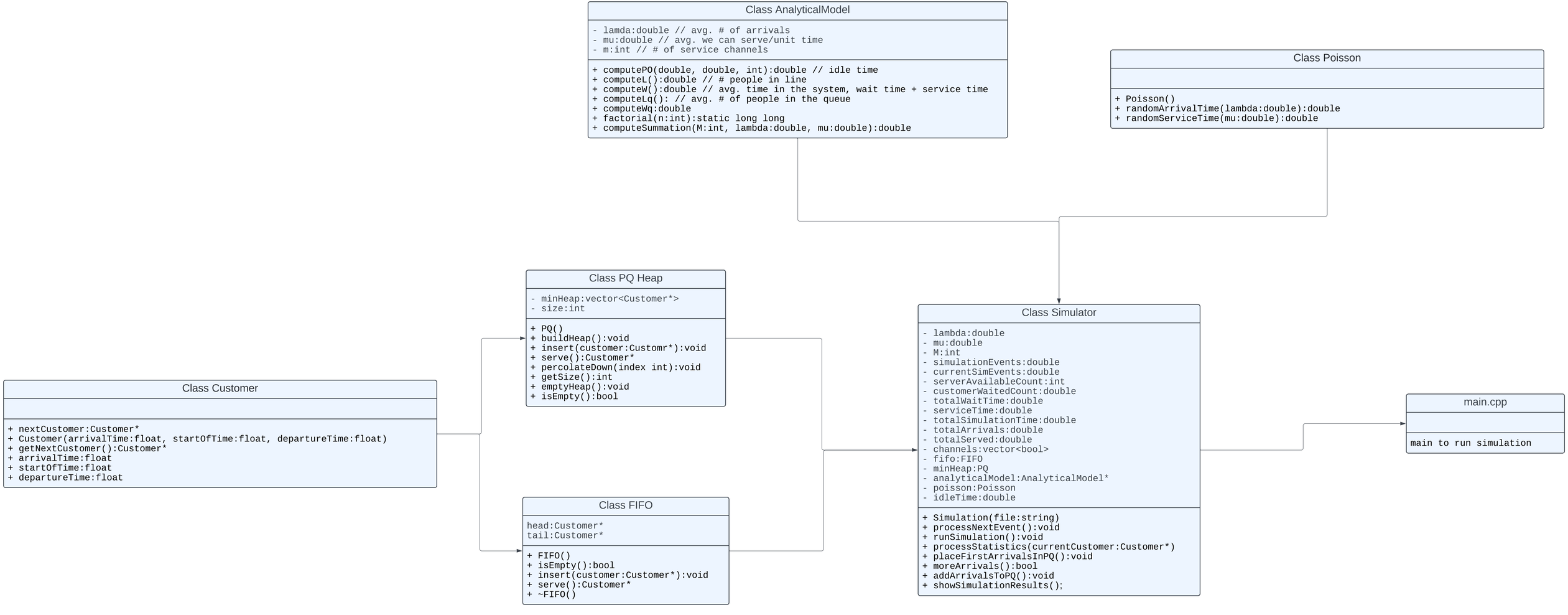
Class AnalyticalModel
- lamda:double // avg. # of arrivals
- mu:double // avg. we can serve/unit time
- m:int // # of service channels
+ computePO(double, double, int):double // idle time
+ computeL():double // # people in line
+ computeW():double // avg. time in the system, wait time + service time
+ computeLq(): // avg. # of people in the queue
+ computeWq:double
+ factorial(n:int):static long long
+ computeSummation(M:int, lambda:double, mu:double):double
Class Poisson
+ Poisson()
+ randomArrivalTime(lambda:double):double
+ randomServiceTime(mu:double):double
Class PQ Heap
- minHeap:vector<Customer*>
- size:int
+ PQ()
+ buildHeap():void
+ insert(customer:Customr*):void
+ serve():Customer*
+ percolateDown(index int):void
+ getSize():int
+ emptyHeap():void
+ isEmpty():bool
Class Customer
+ nextCustomer:Customer*
+ Customer(arrivalTime:float, startOfTime:float, departureTime:float)
+ getNextCustomer():Customer*
+ arrivalTime:float
+ startOfTime:float
+ departureTime:float
Class FIFO
head:Customer*
tail:Customer*
+ FIFO()
+ isEmpty():bool
+ insert(customer:Customer*):void
+ serve():Customer*
+ ~FIFO()
Class Simulator
- lambda:double
- mu:double
- M:int
- simulationEvents:double
- currentSimEvents:double
- serverAvailableCount:int
- customerWaitedCount:double
- totalWaitTime:double
- serviceTime:double
- totalSimulationTime:double
- totalArrivals:double
- totalServed:double
- channels:vector<bool>
- fifo:FIFO
- minHeap:PQ
- analyticalModel:AnalyticalModel*
- poisson:Poisson
- idleTime:double
+ Simulation(file:string)
+ processNextEvent():void
+ runSimulation():void
+ processStatistics(currentCustomer:Customer*)
+ placeFirstArrivalsInPQ():void
+ moreArrivals():bool
+ addArrivalsToPQ():void
+ showSimulationResults();
main.cpp
main to run simulation