```
main
+n: int
+lambda: float
+mu: float
+M: int
+menu(): void
+AnalyzeModel(): void
+CalculateIdleTime(): float
+CalculateAvgNumPeopleSystem(Po:float): float
+CalculateAvgTimeInSystem(L:float): float
+CaluclateAvgCustomerQueue(L:float): float
+CalculateAvgWait(Lq:float): float
+CalculateUtilizationFactor(): float
+factorial(num:int): int
```

```
Queue
-TotalServers: int
-serverAvailableCnt: int
-numCustomers: int
-mu: float
-CustomerWaitedCnt: int = 0
-TotalWaitTime: float = 0
-TotalServiceTime: float = 0
-IdleTime: float = 0
-heap_size: int
-PQ[10]: Customer * = {nullptr}
-FIFO: std::queue<Customer*>
+Queue(n:int,lambda:float,mu:float,M:int)
+GetNextRandomInterval(avg:float): float
+InsertIntoPQ(C:Customer *): void
+parent(num:int): int
+left(num:int): int
+right(num:int): int
+processStatistics(): void
+processNextEvent(): void
+DeleteFromPQ(): Customer *
+MinHeapify(num:int): void
+PrintResults(): void
```

```
Customer
-arrivalTime: float
-startOfServiceTime: float
-departureTime: float
-waitTime: float = 0
-priority: float
-customerArrival: bool = true
-nextCustomer: Customer * = nullptr
+Customer(arrival:float)
+setStartOfServiceTime(serviceStart:float): void
+setDepartureTime(depature:float): void
+setWaitTime(wait:float): void
+setNextCustomer(next:Customer *): void
+getPriority(): float
+getArrivalTime(): float
+getDepartureTime(): float
+getStartOfServiceTime(): float
+getWaitTime(): float
+customerIsDeparting(): void
+isArrival(): bool
```