Gedd Johnson

Project 2 Queue Simulation

DSII

Function Decomposition for Project 2

|  |  |  |
| --- | --- | --- |
| File | Functions | Purpose |
| main.c | 1.) unsigned long factorial(double);  2.) double PoEq(double, double, double)  3.) double LEq(double, double, double, double )  4.) double WEq(double , double )  5.) double LqEq(double, double, double)  6.) double WqEq(double, double)  7.) rhoEq(double, double, double)  8.) double getNextRandInterval(double)  9.) void processNextEvent(pqP, queueP, custP, int, int)  10.) void computeStatsPrint(int, int, int, int, double, pqP)  11.) int main(void) | 1.) Computes the factorial used in the math equations.  2.) Analytical Po equation  3.) Analytical L equation  4.) Analytical W equation  5.) Analytical Lq equation  6.) Analytical Wq equation  7.) Analytical rho equation  8.) Computes a random interval according to the negative exp. distribution.  9.) Processes customers that are de-queued form the PQ as arrivals or departures and take appropriate actions including freeing Customers.  10.) Computes both the analytical and simulated stats and prints them to the screen.  11.) Executes program. |
| Queue.h | None | None |
| Queue.c  (FIFO) | 1.) queueP createQueue()  2.) void enQueue(queueP, custP)  3.) custP deQueue(queueP)  4.) void printQueue(queueP)  5.) queueP freeQueue(queueP) | 1.) Creates a FIFO queue and returns a pointer to it.  2.) Places a customer into the queue.  3.) Removes and returns the customer pointed to by head.  4.) Print the contents of the queue.  5.) Free the dynamically allocated space used by the queue. |
| PQ.h | None | None |
| PQ.c  (Priority) | 1.) pqP createPQ()  2.) void InitEnPQ(pqP, custP)  3.) void enPQ(pqP, custP)  4.) custP dePQ(pqP)  5.) void printPQ(pqP pqPtr)  6.) pqP freePQ(pqP) | 1.) Creates the PQ, initializes all counters and theSize, and returns a pointer to the PQ.  2.) Used for placing the first customers into the PQ and “primes the pump.”  3.) Places a customer in the PQ and maintains the heap condition.  4.) Removes the customer at the head of the PQ and maintains the heap condition, returns the customer.  5.) Prints the contents of the PQ.  6.) Free the dynamically allocated space used by the PQ. |
| Customer.h | None | None |
| Customer.c | 1.) custP newCustomer(double); | 1.) Generates a new customer, initializes appropriate member, and returns a pointer to the new customer. |

Data Structures: ADTs are FIFO queue, priority queue (PQ), and Customer

struct Customer{

double PQtime;

double arrivalTime;

double startOfServiceTime;

double departureTime;

struct Customer \*nextCust; // use for FIFO queue list

};

struct Queue{

custP head;

custP tail;

int numFIFO;

};

struct PQ{ // Note that this struct contains all of the simulation stats

custP customerPQ[PQSIZE + 1];

int theSize;

int serverAvailableCount;

double Po\_sim; // the following are the simulation stats to be computed

double L\_sim;

double W\_sim;

double Lq\_sim;

double Wq\_sim;

double rho\_sim;

double numCustomerWaiting\_sim;

double totalServiceTime;

double totalWaitTime;

};