1. Declare member data in class *Sim.*
   1.1. Arrival related data
      1.1.1. **double** *arvt*; //Mean time between arrivals: 1st parameter read $\left(\frac{1}{\lambda}\right)$
      1.1.2. **double** *arvr*; // Arrival rate lambda (λ):
   1.2. Service related data
      1.2.1. **double** *srvt*; //Mean time between departures: 2nd parameter $\left(\frac{1}{\mu}\right)$
      1.2.2. **double** *srvr*; //Departure rate mu (μ):
   1.3. Response time related data
      1.3.1. **double** *E_R*; //Expected response time $E[R] = \frac{E[N]}{\lambda}$.
      1.3.2. **double** *Var_R*; Response time variance: $Var[R] = \frac{Var[N]}{\lambda}$
      1.3.3. **double** *Sim_R*; //Average response time for this simulation: $\frac{Tot\_R}{N}$
      1.3.4. **double** *Tot_R*; //Accumulated wait of the customers that were served
      1.3.5. **double** *Max_R*; //Longest wait of any customer that was served
   1.4. Line length related data
      1.4.1. **int** *N*; //Number of customers served
      1.4.2. **double** *E_N*; //Expected length of the line $E[N] = \frac{\rho}{1-\rho}$, Calculated mean line length
      1.4.3. **double** *Var_N*; //Line length variance. $Var[N] = \frac{\rho}{(1-\rho)^2}$
      1.4.4. **double** *Sim_N*; //Average length of the line totallength/duration
      1.4.5. **int** *Max_N*; //Longest line length in any interval
      1.4.6. **double** *Tot_N*; //Sum of the products of line lengths multiplied by the time in line.
   1.5. Simulation interval time related data
      1.5.1. **int** *Sim_L*; //Simulation time limit: 3rd parameter. The duration of the simulation can be no more than the value assigned to *Sim_L.* As a consequence, the actual duration of the simulation, recorded in variable *Sim_D*, will always be less than *Sim_L.*
      1.5.2. **double** *Sim_D; //*Duration of the simulation. The simulation terminates on the event that is closest to simulation time limit, *Sim_L*. The value of *Sim*_D is computed by assigning the time of the most recent event.
   1.6. Traffic intensity
      1.6.1. **double** *rho*; // $\rho = \frac{\lambda}{\mu}$ arvr/srvr, traffic intensity

2. Code function prototypes and implementations of member functions.
   - 2.1. Create the constructor *Sim*::*Sim*(); Initialize all member data to zero.
   - 2.2. Create member function **void** *Sim*::*Print*(**char*** *t*,**int** *v*,**char*** *u*);
   - 2.3. Create member function **void** *Sim*::*Print*(**char*** *t*,**double** *v*,**char*** *u*);
   - 2.4. Create member function **void** *Sim*::*Print*(*ostream*& o);
     - 2.4.1. Print the mean time between arrivals, *arvt*.
     - 2.4.2. Print the arrival rate, 60**arvr*, customers/hour.
     - 2.4.3. Print the mean time between departures, *srvt*.
     - 2.4.4. Print the departure rate, 60**srvr*, customers/hour.
     - 2.4.5. Print the expected response time, *E_R*.
     - 2.4.6. Print the response time variance, *Var_R*.
     - 2.4.7. Print the simulation response time, *Sim_R*.
     - 2.4.8. Print the expected line length, *E_N*.
     - 2.4.9. Print the line length variance, *Var_N*.
     - 2.4.10. Print the simulation mean line length, *Sim_N*.
     - 2.4.11. Print the number of "customers served," *N*.
     - 2.4.12. Print the simulation maximum response time, *Max_R*.
     - 2.4.13. Print the simulation maximum line length, *Max_N*.
   - 2.5. Create member function **void** *Sim*::*Run*(*istream*& i);
     - 2.5.1. Read input parameters.
       - 2.5.1.1. Read the mean time between arrivals, *arvt* $\left(\frac{1}{l}\right)$
       - 2.5.1.2. Read the mean time between departures, *srvt* $\left(\frac{1}{m}\right)$
       - 2.5.1.3. Read the simulation time limit *Sim_L*.
     - 2.5.2. Compute data that are independent of the simulation.
       - 2.5.2.1. Compute the arrival rate, *arvr*, l
       - 2.5.2.2. Compute the departure rate, *srvr*, m
       - 2.5.2.3. Compute the traffic intensity, *rho*, $r = \frac{l}{m}$
       - 2.5.2.4. Compute the expected length of the line *E_N*, $E[N] = \frac{r}{1-r}$
       - 2.5.2.5. Compute the expected response time, *E_R*, $E[R] = \frac{E[N]}{l}$
       - 2.5.2.6. Compute the line length variance, *Var_N*, $Var[N] = \frac{r}{(1-r)^2}$
       - 2.5.2.7. Compute the response time variance, *Var_R*, $Var[R] = \frac{Var[N]}{l}$
     - 2.5.3. Initialize the Event manager assigning l and m.
     - 2.5.4. Assign zero to member *Sim_D*.
     - 2.5.5. Loop forever
       - 2.5.5.1. Obtain an occurrence of an event.
       - 2.5.5.2. If the time of the occurrence exceeds the simulation limit, terminate the simulation
       - 2.5.5.3. Compute the interval between the last event and this event.
       - 2.5.5.4. Obtain the number of customers waiting on the queue.
       - 2.5.5.5. Add the product of the number of customers waiting on the queue and the interval between this event and the previous event to the running total of these products, *Tot_N*. Member Tot_N is the weighted sum of the number of customers times the amount time they waited in line. The sum of products recorded in Tot_N can be

divided by the total time of the simulation, Sim_D, to find the weighted average of how many people were waiting throughout the simulation. Consider the following example.

Customers in line	Time waited	Product
1	**5**	**5**
2	**3**	**6**
4	**1**	**4**
1	**1**	**1**
Total	**10**	**16**

The average number of customers in line is $\overline{N} = \frac{\sum_{i=1}^{n} c_i t_i}{\sum_{i=1}^{n} t_i} = \frac{16}{10} = 1.6$

2.5.5.6. Update the computation for the maximum number of customers in line.
2.5.5.7. Record the time of this event in *Sim_D*.
2.5.5.8. Determine if the occurrence is an arrival or a departure and process accordingly.
2.5.6. Print the simulation duration, *Sim_D*.
2.6. Create member function **void** *Sim*::*Arrival*(int *time*);
2.6.1. **if** the queue is full **return**.
2.6.2. Put the current time, *time*, on the queue.

2.7. Create member function **void** *Sim*::*Departure*(**int** *time*);

2.7.1. if the queue is empty return.

2.7.2. Find the response time, r, for this customer. $r = time - Q.Deq()$ where *Q.Deq()* is the time removed from the queue.

2.7.3. Add the response time, *r*, to the running total of all response times, *Tot_R*.

2.7.4. Compute the maximum response time, *Max_R*, by finding the maximum of *Max_R* and *r*.

2.7.5. Increment the total number of customers served, *N*.

3. Compute results obtainable by running the simulation.

3.1. Compute the simulation response time, Sim_R, $Sim\_R = \frac{Tot\_R}{N}$

3.2. Compute the simulation mean line length, *Sim_N*, $Sim\_N = \frac{Tot\_N}{Sim\_D}$