```c
#define _CRT_SECURE_NO_WARNINGS
//---Include File---//
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

//function prototype
double exponential(double x);

//Main Program
int main(void)
{
    double SIM_TIME, ARR_TIME, SERV_TIME;
    printf("Input:\n");
    printf(" Simulation Time : ");
    scanf("%lf", &SIM_TIME);
    printf(" Arrival Time    : ");
    scanf("%lf", &ARR_TIME);
    printf(" Service Time    : ");
    scanf("%lf", &SERV_TIME); //User Input for 模拟时间， 平均到达时间，平均服务时间

    double   time = 0.0;           // Simulation time
  double   t1 = 0.0;            // Time for next event #1 (arrival)
  double   t2 = SIM_TIME;       // Time for next event #2 (departure)
  unsigned int n = 0;           // Number of customers in the system

  unsigned int c = 0;           // Number of service completions
  double   b = 0.0;             // Total busy time
  double   s = 0.0;             // Area of number of customers in system
  double   tn = time;           // Variable for "last event time"
  double   tb;                  // Variable for "last start of busy time"
  double   x;                   // Throughput
  double   u;                   // Utilization
  double   l;                   // Mean number in the system
  double   w;                   // Mean residence time

    // Main simulation loop
  while (time < SIM_TIME)
  {
    if (t1 < t2)                // *** Event #1 (arrival) ***
    {
      time = t1;
      s = s + n * (time - tn);  // Update area under "s" curve
      n++;
      tn = time;                // tn = "last event time" for next event
      t1 = time + exponential(ARR_TIME);
      if (n == 1)
      {
        tb = time;              // Set "last start of busy time"
        t2 = time + exponential(SERV_TIME);
      }
    }
    else                        // *** Event #2 (departure) ***
    {
      time = t2;
      s = s + n * (time - tn);  // Update area under "s" curve
      n--;
      tn = time;                // tn = "last event time" for next event
      c++;                      // Increment number of completions
      if (n > 0)
        t2 = time + exponential(SERV_TIME);
      else
      {
        t2 = SIM_TIME;
        b = b + time - tb;      // Update busy time sum if empty
      }
    }
  }

  // Compute outputs
  x = c / time;   // Compute throughput rate
  u = b / time;   // Compute server utilization
  l = s / time;   // Compute mean number in system
  w = l / x;      // Compute mean residence or system time

    // Output results
  printf("=============================================================== \n");
  printf("=            *** Results from M/M/1 simulation ***            = \n");
  printf("=============================================================== \n");
  printf("=  Total simulated time         = %3.4f sec   \n", SIM_TIME);
  printf("=============================================================== \n");
  printf("=  INPUTS:                                    \n");
  printf("=    Mean time between arrivals = %f sec      \n", ARR_TIME);
  printf("=    Mean service time          = %f sec      \n", SERV_TIME);
  printf("=============================================================== \n");
  printf("=  OUTPUTS:                                   \n");
  printf("=    Number of completions      = %ld cust    \n", c);    //顾客数目
  printf("=    Throughput rate            = %f cust/sec \n", x);
  printf("=    Server utilization         = %f %%       \n", 100.0 * u); //服务器利用率
  printf("=    Mean number in system      = %f cust     \n", l);
  printf("=    Mean residence time        = %f sec      \n", w);    //平均等待时间
  printf("=============================================================== \n");
}


double exponential(double x)
{
  double z;                     // Uniform random number from 0 to 1

  // Pull a uniform RV (0 < z < 1)
  do
  {
    z = ((double) rand() / RAND_MAX);
  }
  while ((z == 0) || (z == 1));

  return(-x * log(z));
}

```