**CS 3360 Programming Assignment 1 Report**

**Overview**

This report briefly describes how I implemented the two programming problems and the results of them, including the methodology for generating custom random numbers and the simulation of system failures. Some of this description is also included with the code in comment blocks that I typed out while thinking through each problem and part.

**Problem 1:**

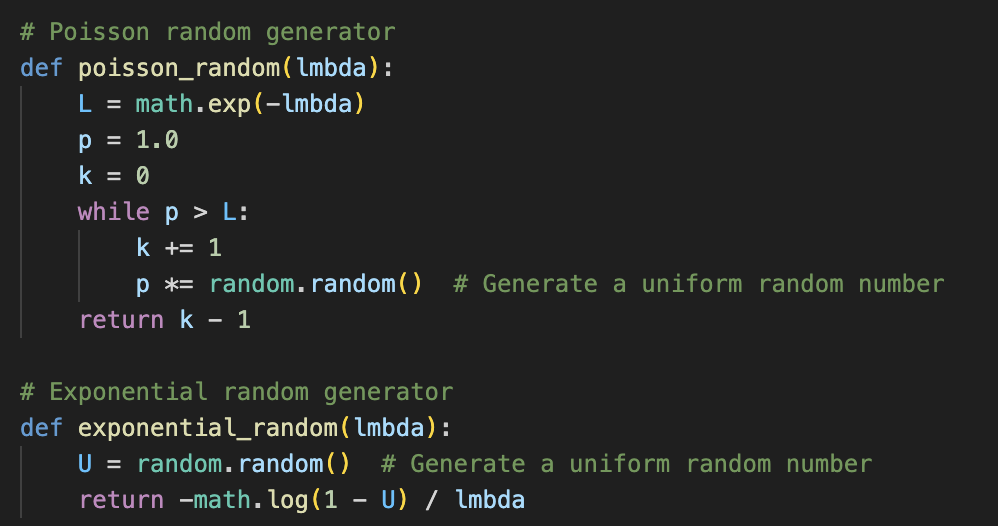
Description

In this problem, we generated a workload for 1000 processes. The processes arrive with an expected average arrival rate of 2 processes per second, and each process has a requested service time that follows an Exponential distribution with a mean of 1 second.

The arrival times follow a Poisson process, which was simulated using the inverse transform sampling method. The inter-arrival times between processes were generated using an Exponential distribution. The service times were also generated using the Exponential distribution with an average service time of 1 second.

Key elements of the program:

* Custom Poisson and Exponential Random Generators: The program uses custom functions to generate random numbers that follow Poisson and Exponential distributions, which is a key requirement for this assignment.



* Output: A list of processes is printed, where each line contains the Process ID, Arrival Time, and Service Time.A screen shot of a computer

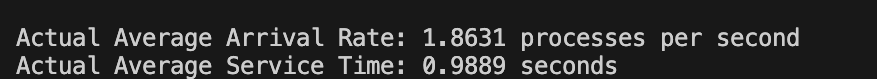
  Description automatically generated

Results

The program prints out the workload for 1000 processes, including the arrival time and service time for each process. Additionally, the actual averages were computed from the generated data:

* Actual Average Arrival Rate: The actual average arrival rate was calculated based on the total number of processes and the time taken for the last process to arrive.
* Actual Average Service Time: The actual average service time was computed as the mean of the generated service times.

Example Output:



Answers to Questions:

* The actual average arrival rate and service time closely match the expected values of 2 processes per second and 1 second, respectively, with minor deviations due to randomness every time it was run.

**Problem 2:**

Part A: Simulating Failures and Restoration Times for Two Servers

Description: We simulated the failure and restoration of two mirrored servers over a 20-year period. The Mean Time Between Failures (MTBF) for each server was set to 500 hours, and the restoration time after each failure was fixed at 10 hours (per assignment requirements).

* Custom Exponential Random Generator: The failure times for each server were generated using a custom Exponential random number generator based on uniform random numbers. A screen shot of a computer code

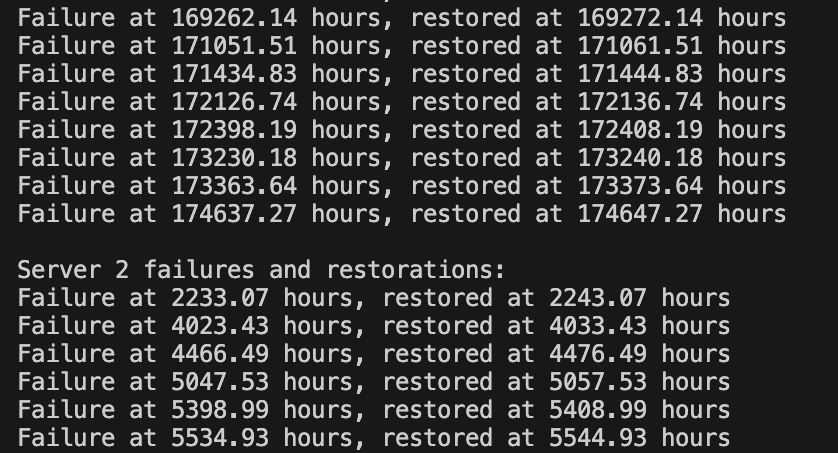
  Description automatically generated
* Restoration Times: After each failure, the restoration period was calculated by adding 10 hours to the failure time. Earlier in the code (not shown below): restore\_time = 10A computer screen shot of text

  Description automatically generated

Results

The program generated the failure and restoration times for both servers over the 20-year period. The output shows the failure time and the corresponding restoration time for each server.

Example Output:



Part B: Determining Time Until System Failure

Description: The system was simulated multiple times to determine when the entire system (both servers) would fail. A system failure occurs when both servers experience a failure within the same 10-hour restoration window. The program simulated this scenario across multiple runs (1000 runs) to compute the average time until system failure.

* System Failure Check: The program checks if any failure from one server occurs during the restoration period of the other server.

A computer screen with text on it

Description automatically generated

* Multiple Simulations: The program runs the simulation multiple times and calculates the average time until system failure.

A computer screen shot of a program

Description automatically generated

Results: After running the simulation 1000 times, the average time until system failure was computed. The result represents the average number of hours it takes for both servers to fail within the same 10-hour window. I added some if statements to print the results in the most readable time format (years, months, days, hours).

Example Output:A black background with white text

Description automatically generated

Answers to Questions:

* The average time until system failure is approximately 1.5 years, based on 1000 simulations. This indicates that under normal conditions, it would take around this many hours before the mirrored server system experiences overlapping failures and the entire system goes down.

Conclusion

This report outlines the implementation and results for generating a process workload and simulating server failures using custom random number generators. The programs adhered to the requirements by avoiding the use of direct random distribution functions from libraries (which would have been useful). The results generated include the actual averages for arrival rates and service times in Problem 1, and the average time until system failure in Problem 2. Both problems were successfully simulated, yielding results that align with the theoretical expectations for Poisson and Exponential distributions.