# COIT20256 Assignment 1 Report

|  |  |
| --- | --- |
| Student ID |  |
| Student Name |  |

This coversheet **must** be completed with your submission.

Please check (X) to indicate that you have satisfied these requirements

This work and my programming submission are my original work and no part of it has been copied from any other student’s work or any other source.

No part of this work or my code has been written for me by any other person/student.

I have taken proper and reasonable care to prevent this work and my code from being copied by another student.

I acknowledge that it is my responsibility to check that the file submitted is readable and complete and that the code submitted can be uploaded in NetBeans and will compile and run correctly.

I understand that plagiarism also includes the act of assisting or allowing another person to plagiarise or to copy my own work.

# Description of the Program and Phase Completed

**Final Phase Completed:** phase 5 and 6 both were completed and submitted

The application creates multiple events like the ArrivalEvent, OrderEvent and LeaveEvent, runs simulation and then creates a statistics.txt file and saves it in the working directory.

# Testing

Following checks have been performed:

1. Check that the simulation runs for the correct number of time units – PASSED.
2. Check the id is unique for every customer group – PASSED.
3. Check the simulation trace to verify the format of the output - PASSED.
4. Check the simulation trace to make sure that the groups arrive – PASSED.
5. Check the simulation trace to make sure that they enter if seats are available – PASSED.
6. Check the simulation trace to make sure that the groups are denied entry if seats are not available – PASSED.
7. Check the simulation trace to get their order – PASSED.
8. Check if the group left and the seats become unoccupied – PASSED.
9. Check how many groups still in the shop – PASSED.
10. Verify if the statistics.txt file is created – PASSED.
11. Verify if the statistics.txt file contents are correct – PASSED.
12. Check if the upper bounds and the lower bound of the random generator is working properly and accurately – PASSED.

# Expected Output for Phase 5

Number of seats = 8

Stop time = 5

At t=0 , group 0 arrives with 2 people, so now the total number of seats = 6 (8-2). Once group 0 is seated group 1 arrives , so now the total number of seats become 6-2 = 4.

Next group 2 arrives with 2 people , so now the total number of seats become 4-2 = 2.

Next group 3 arrives with 2 people , so the total number of seats become 2-2 =0

# Actual Output for Phase 5

## Simulation trace to the standard output:

t= 0 group: 0 < 2 people arrived >

t= 0: Group 0 ( 2 people ) seated

t= 2 group: 1 < 2 people arrived >

t= 2: Group 1 ( 2 people ) seated

t= 1: Order served for Group 0

t= 4 group: 2 < 2 people arrived >

t= 4: Group 2 ( 2 people ) seated

t= 3: Order served for Group 1

## Statistics written to the statistics.txt file

Statistics  
==========  
The number of people served = 6  
The lost business = 0 people  
  
The following groups are in the shop:  
=====================================  
Group 0 (2 people) arrived at t = 0  
Group 1 (2 people) arrived at t = 2  
Group 2 (2 people) arrived at t = 4  
  
The following groups are in the history/log:  
============================================  
Group 0 (2 people) arrived at t = 0  
Group 1 (2 people) arrived at t = 2  
Group 2 (2 people) arrived at t = 4

# Actual Output for Phase 6

Group generator bound between 1 to 4

Order time after arrival bound between 1 to 5

Leave event bound between 5 to 12

Console output:

t= 0 group: 0 < 3 people arrived >

t= 0: Group 0 ( 3 people ) seated

t= 2 group: 1 < 2 people arrived >

t= 2: Group 1 ( 2 people ) seated

t= 4: Order served for Group 0

t= 4 group: 2 < 1 people arrived >

t= 4: Group 2 leaves as there are insufficient seats for the group

t= 6: Order served for Group 1

t= 11: Group 0 leaves

t= 6 group: 3 < 4 people arrived >

t= 6: Group 3 leaves as there are insufficient seats for the group

t= 21: Group 1 leaves

t= 8 group: 4 < 1 people arrived >

t= 8: Group 4 ( 1 people ) seated

t= 10 group: 5 < 4 people arrived >

t= 10: Group 5 ( 4 people ) seated

t= 12: Order served for Group 4

t= 12 group: 6 < 3 people arrived >

t= 12: Group 6 leaves as there are insufficient seats for the group

t= 14: Order served for Group 5

t= 23: Group 4 leaves

t= 14 group: 7 < 4 people arrived >

t= 14: Group 7 leaves as there are insufficient seats for the group

t= 25: Group 5 leaves

t= 16 group: 8 < 2 people arrived >

t= 16: Group 8 ( 2 people ) seated

t= 18 group: 9 < 2 people arrived >

t= 18: Group 9 ( 2 people ) seated

t= 19: Order served for Group 8

t= 20 group: 10 < 2 people arrived >

t= 20: Group 10 leaves as there are insufficient seats for the group

t= 23: Order served for Group 9

t= 26: Group 8 leaves

t= 22 group: 11 < 1 people arrived >

t= 22: Group 11 ( 1 people ) seated

Statistics file:  
  
Statistics  
==========  
The number of people served = 15  
The lost business = 14 people  
  
The following groups are in the shop:  
=====================================  
Group 9 (2 people) arrived at t = 18  
Group 11 (1 people) arrived at t = 22  
  
The following groups are in the history/log:  
============================================  
Group 0 (3 people) arrived at t = 0  
Group 1 (2 people) arrived at t = 2  
Group 2 (1 people) arrived at t = 4  
Group 3 (4 people) arrived at t = 6  
Group 4 (1 people) arrived at t = 8  
Group 5 (4 people) arrived at t = 10  
Group 6 (3 people) arrived at t = 12  
Group 7 (4 people) arrived at t = 14  
Group 8 (2 people) arrived at t = 16  
Group 9 (2 people) arrived at t = 18  
Group 10 (2 people) arrived at t = 20  
Group 11 (1 people) arrived at t = 22

# Bugs and Limitations

## Bugs

Sorting not working properly

## Limitations

1. Time interval increases by 2 units always, cannot be modified
2. Seed value is 1

# Additional Future Work

1. Sorting needs to be corrected.
2. Design patterns can be included.
3. Unit tests needs to be written.

# Answers to Questions

**Question 1**

To repeatably have the same random sequence always

**Question 2**

The name of the access specifier is Default access specifier, meaning the variable can only be accessed within the same package.

Classes in the same package could access the variable. this is also called package private.

**Question 3**

By using the protected access specifier