

**Preparatory data Structure (CSCI 591)**



**Comparing three Sorting Algorthms**

**Project - VIII**

**April 9, 2020**

**St. Cloud state university**

**Department of Computer Science**

Submitted By: Taddese Erba

# **Design Document**

## **Introduction**

A sorting algorithm is an algorithm that puts elements in a certain order. The order of the elements is often governed by certain comparison rules that a programmer is interested in. Sorting is often employed to optimize the operstion of other algorithms that operates on the elements.

There are a number of sorting algorithms in computer science. The most common ones are: Insertion Sort, Merge Sort, and Quicksort. Sorting algorithms use numerical values to sort elements of a data stracture. This project explores the use of sorting algorithms and their data structure and compares the time complexity of these sorting algorithms.

## **Data Structure**

The program has three distinct files. The sort.h file contains all the declaration of the required member functions and variable. It is the framework for sort class implementation. It consists of three private variables that are used for counting the number of program executions. Furthermore, the sort class contains eleven functions each with their operations as discussed in the following section of this document.

## **Functions**

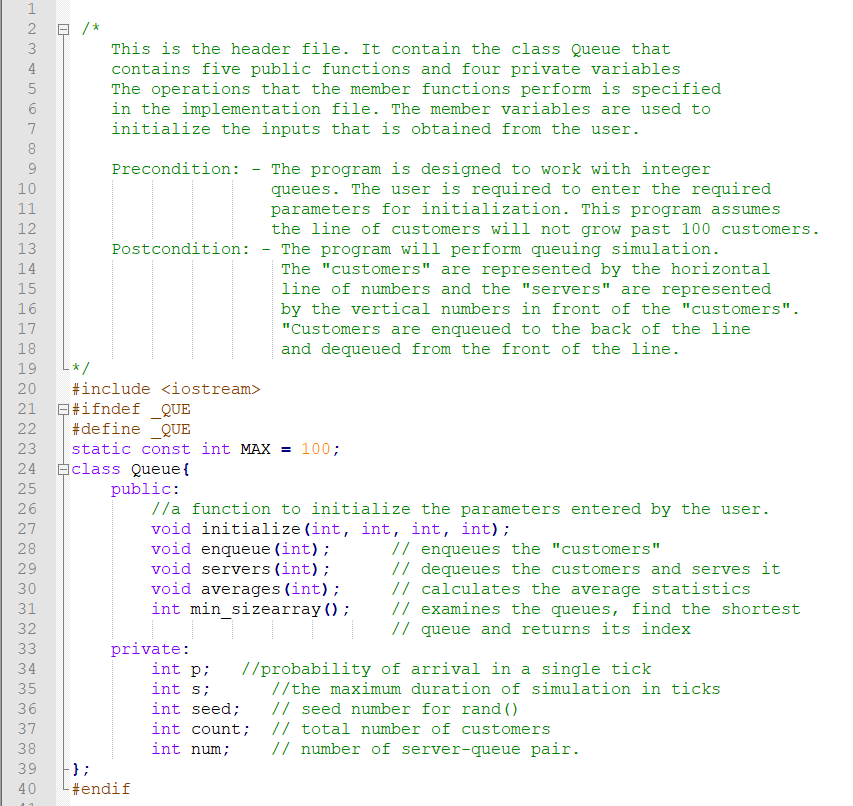
As described in the Data Structure section of this document, there are eleven functions in this project. The functions int counter1() is used to return the number of program execution of the insertion sort algorithm. The functions int counter2() is used to return the number of program execution of the merge sort algorithm. The functions int counter3() is used to return the number of program execution of the quicksort algorithm. The three counting functions takes no argument and returns the number of cycles the program executes. The void resetCounter() is used to initialize the counters. It takes no argument and returns none. The void printArray(int [], int) function is used to print the unsorted and sorted arrays. It takes the array and its size as an argument but does not return anything. The void insertion\_sort(int [], int) function takes the the array and its size as its argument, performs sorting operation on the array. The functions, void merge(int [], int, int, int) is used to merge the two halves of the array. The functions, void merge\_sort(int [], int, int) is used to recursively sort the two halves of the array. The functions, int partition(int [], int, int) is used rearrange the elements of the array in the two halves of the array based on their value compared to the element of the array called the pivot. It takes the array, the starting index of the array, and the ending index of the array as its arguments and returns an integer value, the index of the new pivot position. The functions, void quick\_sort(int [], int, int) is used to recursively sort the two halves of the array. The functions, void swap(int\*, int\*) is used swap the array ellements to rearrange them and put them in their correct position.

## **The Main Program**

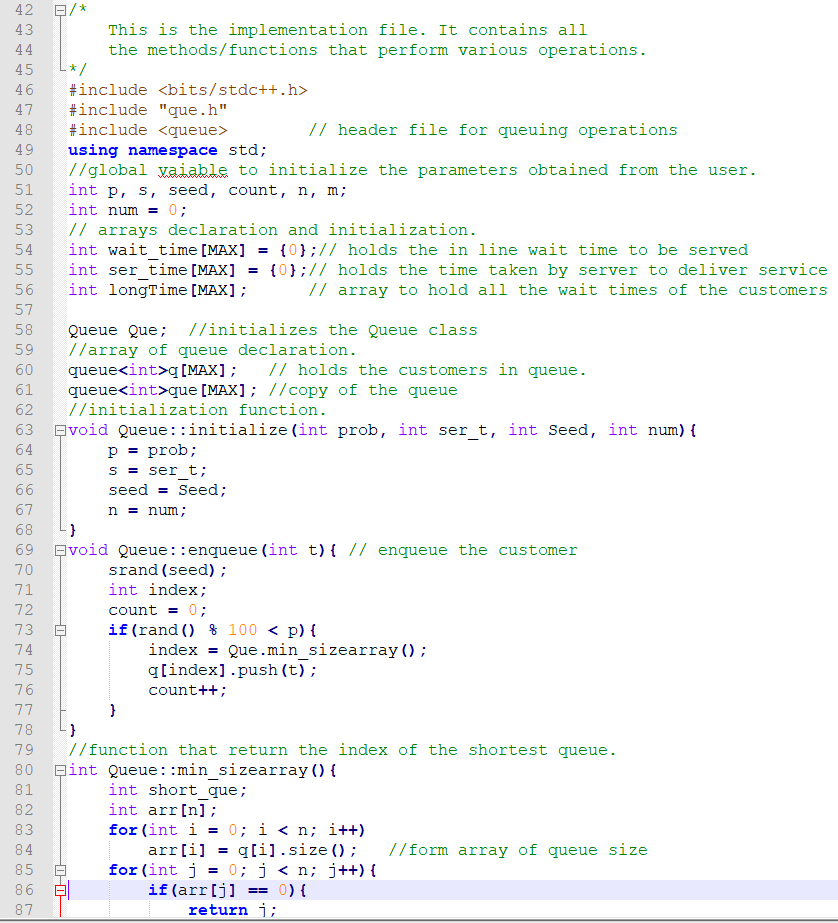
The main() function is used to promote the user to enter the size of the array and the initial smallest value of the array. It hosts the variables, the testing functions and display the output on the terminal. A series of the statement is printed on the terminal requesting inputs, and functions are called from the class to do the job. It also reports what the user wants to see in the output.

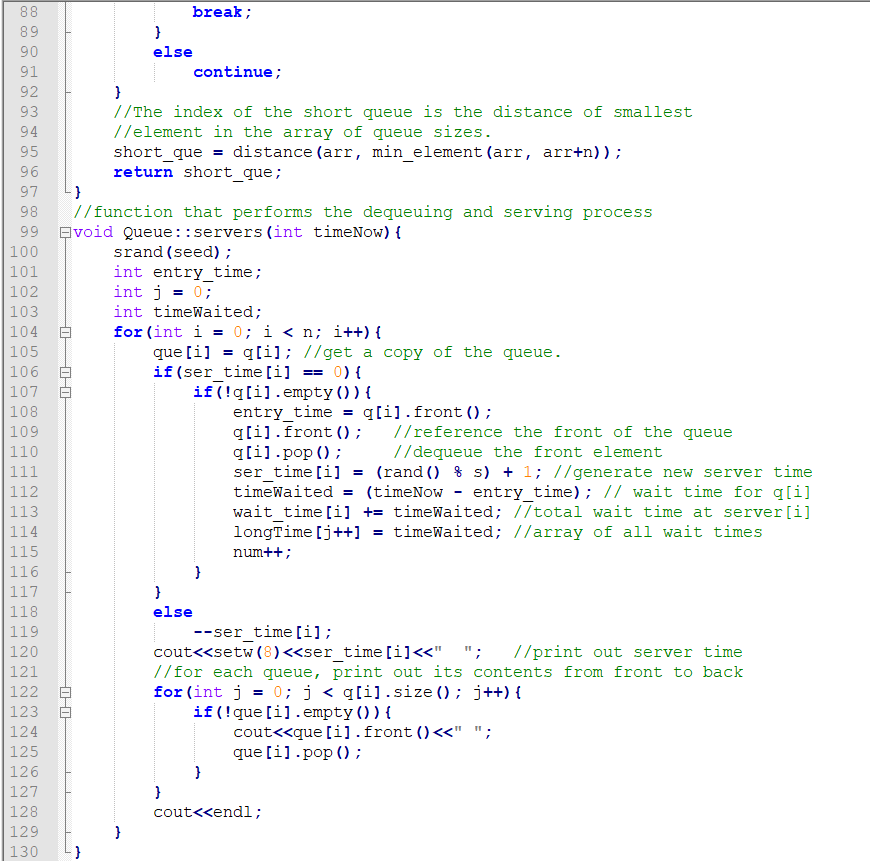
# **Code listing**

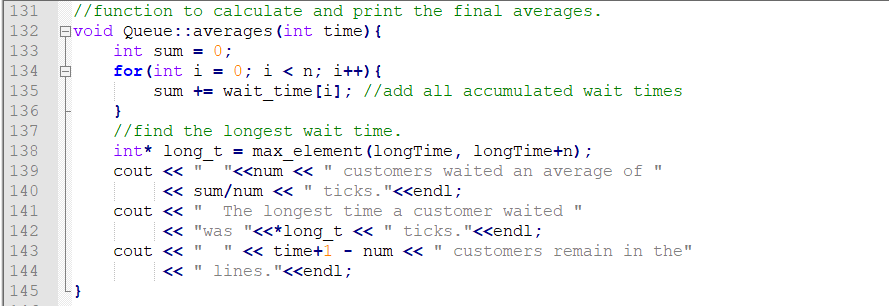
## **The header file (List.h)**



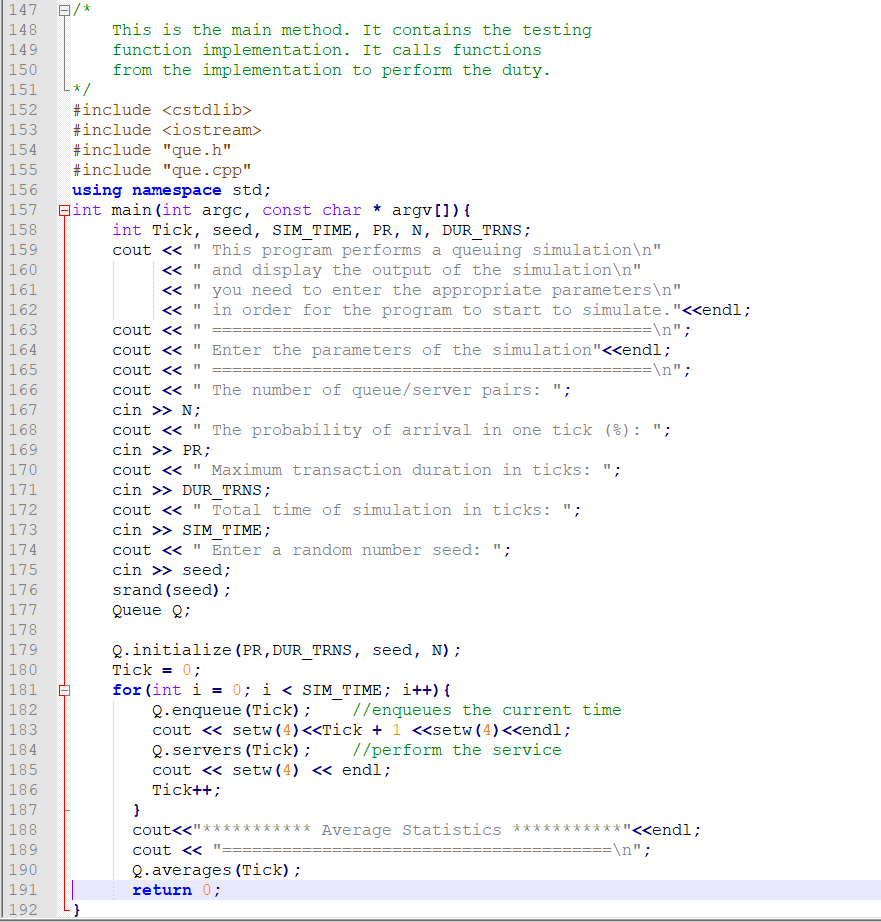
## **The implementation file (List.cpp)**





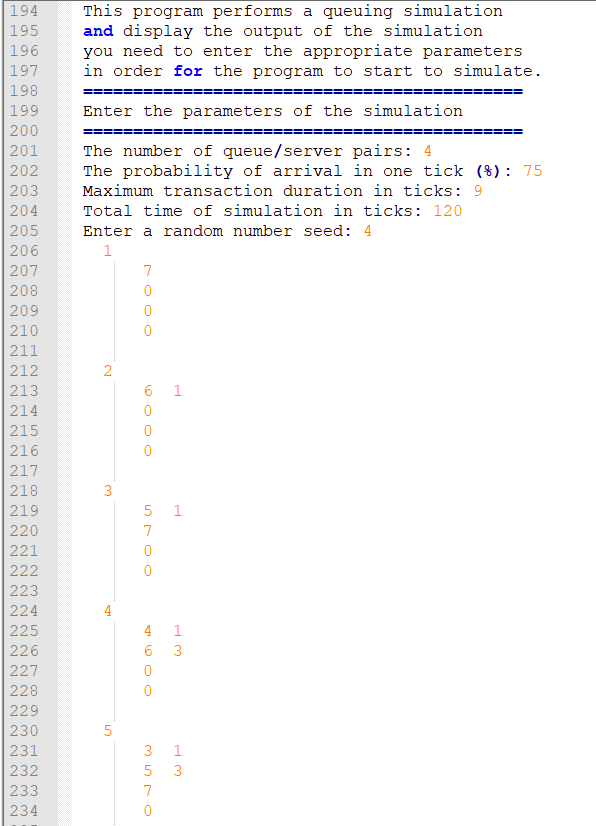


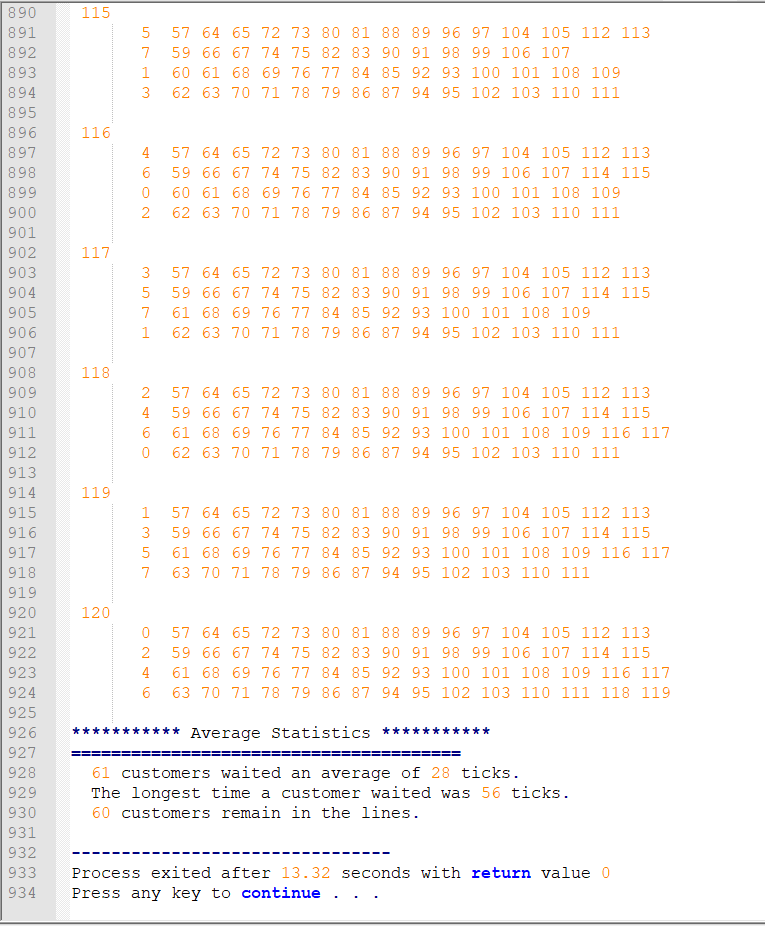
## **The testing file (main.cpp)**



# **Test Results**

I will be providing the partial run here. However, I am attaching the whole run at the end of this document for reference.





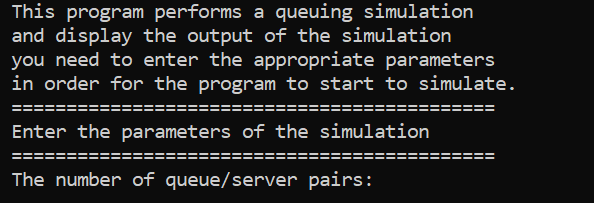
# **User document**

This program can perform queuing simulation operations. To run the program, you must perform the following steps.

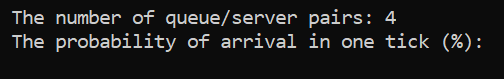
* Run the main.cpp. To compile and run the program, enter the following command to on the terminal window.

g++ -o main main.cpp

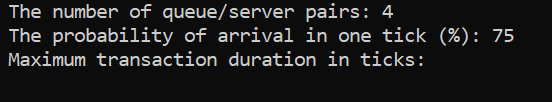
* The program will compile and open the following window:



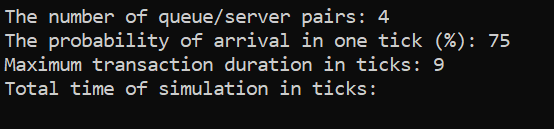
* Once the window opens, enter an integer number for server-queue pair.



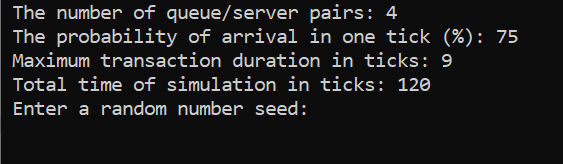
* Next, enter the probability (%) that a customer will arrive in the next clock time (tick).



* Next, enter the maximum time a server might need to perform a service ( integer number only).

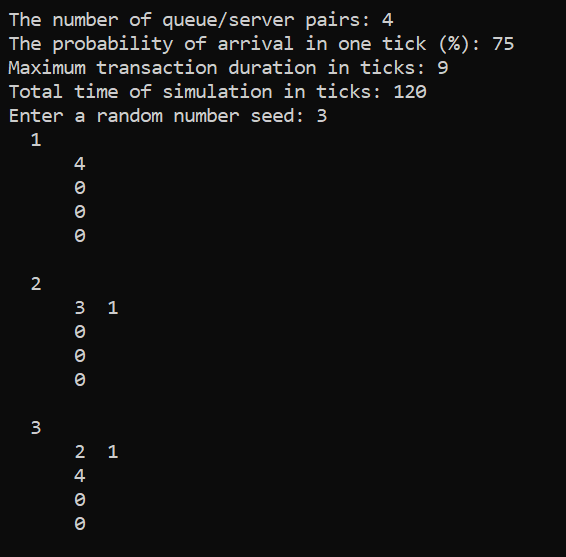


* Next, enter for how long you want to simulate (integer number greater than or equal to 100 is preferable)

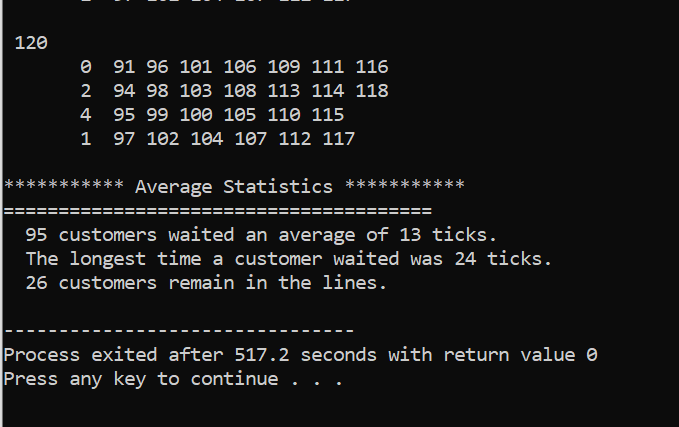


* Finally inter a pseudo number generation initial number.

The result is a long line of run that looks like this:



All the way to down it looks like this:



# **Summery**

The simulation of a queuing operation using the queue abstract data time is the main goal of this project. The implementation of the simulation used various functions that perform operations on a random number generated by the rand() function. The servers use an array that holds the server time and the queues (equal to the number of servers) use an array index to indicate to which server they are assigned and can contain a line of customers. A line will be dequeued and served only if its server is free

In terms of difficulty, this project is the most difficult so far. It requires the use of an array of queues. The operation on each member of the queue requires the identification of the queue index. At the same time, the server-queue pair operation should go and in hand which otherwise will not produce the intended output. This project can further be improved to handle a problem in which a customer from the longest line (waited for a long time) can proceed to the immediate free server (different from to which he is already tied to). However, this will create more complications than this project already has and might be difficult to achieve at this level.

By completing this project, I have gained a significant level of confidence and the necessary knowledge to work with queues. Although queues can handle a lot of problems by themselves, I think, using them with stacks will probably make it easier.

# **Complete run file.**

========================================================================

This program performs a queuing simulation

and display the output of the simulation

you need to enter the appropriate parameters

in order for the program to start to simulate.

============================================

Enter the parameters of the simulation

============================================

The number of queue/server pairs: 4

The probability of arrival in one tick (%): 75

Maximum transaction duration in ticks: 9

Total time of simulation in ticks: 120

Enter a random number seed: 4

1

7

0

0

0

2

6 1

0

0

0

3

5 1

7

0

0

4

4 1

6 3

0

0

5

3 1

5 3

7

0

6

2 1

4 3

6 5

0

7

1 1

3 3

5 5

7

8

0 1

2 3

4 5

6 7

9

7

1 3

3 5

5 7

10

6 8 9

0 3

2 5

4 7

11

5 8 9

7 10

1 5

3 7

12

4 8 9

6 10 11

0 5

2 7

13

3 8 9

5 10 11

7 12

1 7

14

2 8 9

4 10 11

6 12 13

0 7

15

1 8 9

3 10 11

5 12 13

7

16

0 8 9

2 10 11

4 12 13

6 14 15

17

7 9 16

1 10 11

3 12 13

5 14 15

18

6 9 16 17

0 10 11

2 12 13

4 14 15

19

5 9 16 17

7 11

1 12 13

3 14 15

20

4 9 16 17

6 11 18 19

0 12 13

2 14 15

21

3 9 16 17

5 11 18 19

7 13

1 14 15

22

2 9 16 17

4 11 18 19

6 13 20 21

0 14 15

23

1 9 16 17

3 11 18 19

5 13 20 21

7 15

24

0 9 16 17

2 11 18 19

4 13 20 21

6 15 22 23

25

7 9 16 17

1 11 18 19

3 13 20 21

5 15 22 23

26

6 16 17 24 25

0 11 18 19

2 13 20 21

4 15 22 23

27

5 16 17 24 25

7 18 19

1 13 20 21

3 15 22 23

28

4 16 17 24 25

6 18 19 26 27

0 13 20 21

2 15 22 23

29

3 16 17 24 25

5 18 19 26 27

7 20 21

1 15 22 23

30

2 16 17 24 25

4 18 19 26 27

6 20 21 28 29

0 15 22 23

31

1 16 17 24 25

3 18 19 26 27

5 20 21 28 29

7 22 23

32

0 16 17 24 25

2 18 19 26 27

4 20 21 28 29

6 22 23 30 31

33

7 17 24 25

1 18 19 26 27

3 20 21 28 29

5 22 23 30 31

34

6 17 24 25 32 33

0 18 19 26 27

2 20 21 28 29

4 22 23 30 31

35

5 17 24 25 32 33

7 19 26 27

1 20 21 28 29

3 22 23 30 31

36

4 17 24 25 32 33

6 19 26 27 34 35

0 20 21 28 29

2 22 23 30 31

37

3 17 24 25 32 33

5 19 26 27 34 35

7 21 28 29

1 22 23 30 31

38

2 17 24 25 32 33

4 19 26 27 34 35

6 21 28 29 36 37

0 22 23 30 31

39

1 17 24 25 32 33

3 19 26 27 34 35

5 21 28 29 36 37

7 23 30 31

40

0 17 24 25 32 33

2 19 26 27 34 35

4 21 28 29 36 37

6 23 30 31 38 39

41

7 24 25 32 33

1 19 26 27 34 35

3 21 28 29 36 37

5 23 30 31 38 39

42

6 24 25 32 33 40 41

0 19 26 27 34 35

2 21 28 29 36 37

4 23 30 31 38 39

43

5 24 25 32 33 40 41

7 26 27 34 35

1 21 28 29 36 37

3 23 30 31 38 39

44

4 24 25 32 33 40 41

6 26 27 34 35 42 43

0 21 28 29 36 37

2 23 30 31 38 39

45

3 24 25 32 33 40 41

5 26 27 34 35 42 43

7 28 29 36 37

1 23 30 31 38 39

46

2 24 25 32 33 40 41

4 26 27 34 35 42 43

6 28 29 36 37 44 45

0 23 30 31 38 39

47

1 24 25 32 33 40 41

3 26 27 34 35 42 43

5 28 29 36 37 44 45

7 30 31 38 39

48

0 24 25 32 33 40 41

2 26 27 34 35 42 43

4 28 29 36 37 44 45

6 30 31 38 39 46 47

49

7 25 32 33 40 41

1 26 27 34 35 42 43

3 28 29 36 37 44 45

5 30 31 38 39 46 47

50

6 25 32 33 40 41 48 49

0 26 27 34 35 42 43

2 28 29 36 37 44 45

4 30 31 38 39 46 47

51

5 25 32 33 40 41 48 49

7 27 34 35 42 43

1 28 29 36 37 44 45

3 30 31 38 39 46 47

52

4 25 32 33 40 41 48 49

6 27 34 35 42 43 50 51

0 28 29 36 37 44 45

2 30 31 38 39 46 47

53

3 25 32 33 40 41 48 49

5 27 34 35 42 43 50 51

7 29 36 37 44 45

1 30 31 38 39 46 47

54

2 25 32 33 40 41 48 49

4 27 34 35 42 43 50 51

6 29 36 37 44 45 52 53

0 30 31 38 39 46 47

55

1 25 32 33 40 41 48 49

3 27 34 35 42 43 50 51

5 29 36 37 44 45 52 53

7 31 38 39 46 47

56

0 25 32 33 40 41 48 49

2 27 34 35 42 43 50 51

4 29 36 37 44 45 52 53

6 31 38 39 46 47 54 55

57

7 32 33 40 41 48 49

1 27 34 35 42 43 50 51

3 29 36 37 44 45 52 53

5 31 38 39 46 47 54 55

58

6 32 33 40 41 48 49 56 57

0 27 34 35 42 43 50 51

2 29 36 37 44 45 52 53

4 31 38 39 46 47 54 55

59

5 32 33 40 41 48 49 56 57

7 34 35 42 43 50 51

1 29 36 37 44 45 52 53

3 31 38 39 46 47 54 55

60

4 32 33 40 41 48 49 56 57

6 34 35 42 43 50 51 58 59

0 29 36 37 44 45 52 53

2 31 38 39 46 47 54 55

61

3 32 33 40 41 48 49 56 57

5 34 35 42 43 50 51 58 59

7 36 37 44 45 52 53

1 31 38 39 46 47 54 55

62

2 32 33 40 41 48 49 56 57

4 34 35 42 43 50 51 58 59

6 36 37 44 45 52 53 60 61

0 31 38 39 46 47 54 55

63

1 32 33 40 41 48 49 56 57

3 34 35 42 43 50 51 58 59

5 36 37 44 45 52 53 60 61

7 38 39 46 47 54 55

64

0 32 33 40 41 48 49 56 57

2 34 35 42 43 50 51 58 59

4 36 37 44 45 52 53 60 61

6 38 39 46 47 54 55 62 63

65

7 33 40 41 48 49 56 57

1 34 35 42 43 50 51 58 59

3 36 37 44 45 52 53 60 61

5 38 39 46 47 54 55 62 63

66

6 33 40 41 48 49 56 57 64 65

0 34 35 42 43 50 51 58 59

2 36 37 44 45 52 53 60 61

4 38 39 46 47 54 55 62 63

67

5 33 40 41 48 49 56 57 64 65

7 35 42 43 50 51 58 59

1 36 37 44 45 52 53 60 61

3 38 39 46 47 54 55 62 63

68

4 33 40 41 48 49 56 57 64 65

6 35 42 43 50 51 58 59 66 67

0 36 37 44 45 52 53 60 61

2 38 39 46 47 54 55 62 63

69

3 33 40 41 48 49 56 57 64 65

5 35 42 43 50 51 58 59 66 67

7 37 44 45 52 53 60 61

1 38 39 46 47 54 55 62 63

70

2 33 40 41 48 49 56 57 64 65

4 35 42 43 50 51 58 59 66 67

6 37 44 45 52 53 60 61 68 69

0 38 39 46 47 54 55 62 63

71

1 33 40 41 48 49 56 57 64 65

3 35 42 43 50 51 58 59 66 67

5 37 44 45 52 53 60 61 68 69

7 39 46 47 54 55 62 63

72

0 33 40 41 48 49 56 57 64 65

2 35 42 43 50 51 58 59 66 67

4 37 44 45 52 53 60 61 68 69

6 39 46 47 54 55 62 63 70 71

73

7 40 41 48 49 56 57 64 65

1 35 42 43 50 51 58 59 66 67

3 37 44 45 52 53 60 61 68 69

5 39 46 47 54 55 62 63 70 71

74

6 40 41 48 49 56 57 64 65 72 73

0 35 42 43 50 51 58 59 66 67

2 37 44 45 52 53 60 61 68 69

4 39 46 47 54 55 62 63 70 71

75

5 40 41 48 49 56 57 64 65 72 73

7 42 43 50 51 58 59 66 67

1 37 44 45 52 53 60 61 68 69

3 39 46 47 54 55 62 63 70 71

76

4 40 41 48 49 56 57 64 65 72 73

6 42 43 50 51 58 59 66 67 74 75

0 37 44 45 52 53 60 61 68 69

2 39 46 47 54 55 62 63 70 71

77

3 40 41 48 49 56 57 64 65 72 73

5 42 43 50 51 58 59 66 67 74 75

7 44 45 52 53 60 61 68 69

1 39 46 47 54 55 62 63 70 71

78

2 40 41 48 49 56 57 64 65 72 73

4 42 43 50 51 58 59 66 67 74 75

6 44 45 52 53 60 61 68 69 76 77

0 39 46 47 54 55 62 63 70 71

79

1 40 41 48 49 56 57 64 65 72 73

3 42 43 50 51 58 59 66 67 74 75

5 44 45 52 53 60 61 68 69 76 77

7 46 47 54 55 62 63 70 71

80

0 40 41 48 49 56 57 64 65 72 73

2 42 43 50 51 58 59 66 67 74 75

4 44 45 52 53 60 61 68 69 76 77

6 46 47 54 55 62 63 70 71 78 79

81

7 41 48 49 56 57 64 65 72 73

1 42 43 50 51 58 59 66 67 74 75

3 44 45 52 53 60 61 68 69 76 77

5 46 47 54 55 62 63 70 71 78 79

82

6 41 48 49 56 57 64 65 72 73 80 81

0 42 43 50 51 58 59 66 67 74 75

2 44 45 52 53 60 61 68 69 76 77

4 46 47 54 55 62 63 70 71 78 79

83

5 41 48 49 56 57 64 65 72 73 80 81

7 43 50 51 58 59 66 67 74 75

1 44 45 52 53 60 61 68 69 76 77

3 46 47 54 55 62 63 70 71 78 79

84

4 41 48 49 56 57 64 65 72 73 80 81

6 43 50 51 58 59 66 67 74 75 82 83

0 44 45 52 53 60 61 68 69 76 77

2 46 47 54 55 62 63 70 71 78 79

85

3 41 48 49 56 57 64 65 72 73 80 81

5 43 50 51 58 59 66 67 74 75 82 83

7 45 52 53 60 61 68 69 76 77

1 46 47 54 55 62 63 70 71 78 79

86

2 41 48 49 56 57 64 65 72 73 80 81

4 43 50 51 58 59 66 67 74 75 82 83

6 45 52 53 60 61 68 69 76 77 84 85

0 46 47 54 55 62 63 70 71 78 79

87

1 41 48 49 56 57 64 65 72 73 80 81

3 43 50 51 58 59 66 67 74 75 82 83

5 45 52 53 60 61 68 69 76 77 84 85

7 47 54 55 62 63 70 71 78 79

88

0 41 48 49 56 57 64 65 72 73 80 81

2 43 50 51 58 59 66 67 74 75 82 83

4 45 52 53 60 61 68 69 76 77 84 85

6 47 54 55 62 63 70 71 78 79 86 87

89

7 48 49 56 57 64 65 72 73 80 81

1 43 50 51 58 59 66 67 74 75 82 83

3 45 52 53 60 61 68 69 76 77 84 85

5 47 54 55 62 63 70 71 78 79 86 87

90

6 48 49 56 57 64 65 72 73 80 81 88 89

0 43 50 51 58 59 66 67 74 75 82 83

2 45 52 53 60 61 68 69 76 77 84 85

4 47 54 55 62 63 70 71 78 79 86 87

91

5 48 49 56 57 64 65 72 73 80 81 88 89

7 50 51 58 59 66 67 74 75 82 83

1 45 52 53 60 61 68 69 76 77 84 85

3 47 54 55 62 63 70 71 78 79 86 87

92

4 48 49 56 57 64 65 72 73 80 81 88 89

6 50 51 58 59 66 67 74 75 82 83 90 91

0 45 52 53 60 61 68 69 76 77 84 85

2 47 54 55 62 63 70 71 78 79 86 87

93

3 48 49 56 57 64 65 72 73 80 81 88 89

5 50 51 58 59 66 67 74 75 82 83 90 91

7 52 53 60 61 68 69 76 77 84 85

1 47 54 55 62 63 70 71 78 79 86 87

94

2 48 49 56 57 64 65 72 73 80 81 88 89

4 50 51 58 59 66 67 74 75 82 83 90 91

6 52 53 60 61 68 69 76 77 84 85 92 93

0 47 54 55 62 63 70 71 78 79 86 87

95

1 48 49 56 57 64 65 72 73 80 81 88 89

3 50 51 58 59 66 67 74 75 82 83 90 91

5 52 53 60 61 68 69 76 77 84 85 92 93

7 54 55 62 63 70 71 78 79 86 87

96

0 48 49 56 57 64 65 72 73 80 81 88 89

2 50 51 58 59 66 67 74 75 82 83 90 91

4 52 53 60 61 68 69 76 77 84 85 92 93

6 54 55 62 63 70 71 78 79 86 87 94 95

97

7 49 56 57 64 65 72 73 80 81 88 89

1 50 51 58 59 66 67 74 75 82 83 90 91

3 52 53 60 61 68 69 76 77 84 85 92 93

5 54 55 62 63 70 71 78 79 86 87 94 95

98

6 49 56 57 64 65 72 73 80 81 88 89 96 97

0 50 51 58 59 66 67 74 75 82 83 90 91

2 52 53 60 61 68 69 76 77 84 85 92 93

4 54 55 62 63 70 71 78 79 86 87 94 95

99

5 49 56 57 64 65 72 73 80 81 88 89 96 97

7 51 58 59 66 67 74 75 82 83 90 91

1 52 53 60 61 68 69 76 77 84 85 92 93

3 54 55 62 63 70 71 78 79 86 87 94 95

100

4 49 56 57 64 65 72 73 80 81 88 89 96 97

6 51 58 59 66 67 74 75 82 83 90 91 98 99

0 52 53 60 61 68 69 76 77 84 85 92 93

2 54 55 62 63 70 71 78 79 86 87 94 95

101

3 49 56 57 64 65 72 73 80 81 88 89 96 97

5 51 58 59 66 67 74 75 82 83 90 91 98 99

7 53 60 61 68 69 76 77 84 85 92 93

1 54 55 62 63 70 71 78 79 86 87 94 95

102

2 49 56 57 64 65 72 73 80 81 88 89 96 97

4 51 58 59 66 67 74 75 82 83 90 91 98 99

6 53 60 61 68 69 76 77 84 85 92 93 100 101

0 54 55 62 63 70 71 78 79 86 87 94 95

103

1 49 56 57 64 65 72 73 80 81 88 89 96 97

3 51 58 59 66 67 74 75 82 83 90 91 98 99

5 53 60 61 68 69 76 77 84 85 92 93 100 101

7 55 62 63 70 71 78 79 86 87 94 95

104

0 49 56 57 64 65 72 73 80 81 88 89 96 97

2 51 58 59 66 67 74 75 82 83 90 91 98 99

4 53 60 61 68 69 76 77 84 85 92 93 100 101

6 55 62 63 70 71 78 79 86 87 94 95 102 103

105

7 56 57 64 65 72 73 80 81 88 89 96 97

1 51 58 59 66 67 74 75 82 83 90 91 98 99

3 53 60 61 68 69 76 77 84 85 92 93 100 101

5 55 62 63 70 71 78 79 86 87 94 95 102 103

106

6 56 57 64 65 72 73 80 81 88 89 96 97 104 105

0 51 58 59 66 67 74 75 82 83 90 91 98 99

2 53 60 61 68 69 76 77 84 85 92 93 100 101

4 55 62 63 70 71 78 79 86 87 94 95 102 103

107

5 56 57 64 65 72 73 80 81 88 89 96 97 104 105

7 58 59 66 67 74 75 82 83 90 91 98 99

1 53 60 61 68 69 76 77 84 85 92 93 100 101

3 55 62 63 70 71 78 79 86 87 94 95 102 103

108

4 56 57 64 65 72 73 80 81 88 89 96 97 104 105

6 58 59 66 67 74 75 82 83 90 91 98 99 106 107

0 53 60 61 68 69 76 77 84 85 92 93 100 101

2 55 62 63 70 71 78 79 86 87 94 95 102 103

109

3 56 57 64 65 72 73 80 81 88 89 96 97 104 105

5 58 59 66 67 74 75 82 83 90 91 98 99 106 107

7 60 61 68 69 76 77 84 85 92 93 100 101

1 55 62 63 70 71 78 79 86 87 94 95 102 103

110

2 56 57 64 65 72 73 80 81 88 89 96 97 104 105

4 58 59 66 67 74 75 82 83 90 91 98 99 106 107

6 60 61 68 69 76 77 84 85 92 93 100 101 108 109

0 55 62 63 70 71 78 79 86 87 94 95 102 103

111

1 56 57 64 65 72 73 80 81 88 89 96 97 104 105

3 58 59 66 67 74 75 82 83 90 91 98 99 106 107

5 60 61 68 69 76 77 84 85 92 93 100 101 108 109

7 62 63 70 71 78 79 86 87 94 95 102 103

112

0 56 57 64 65 72 73 80 81 88 89 96 97 104 105

2 58 59 66 67 74 75 82 83 90 91 98 99 106 107

4 60 61 68 69 76 77 84 85 92 93 100 101 108 109

6 62 63 70 71 78 79 86 87 94 95 102 103 110 111

113

7 57 64 65 72 73 80 81 88 89 96 97 104 105

1 58 59 66 67 74 75 82 83 90 91 98 99 106 107

3 60 61 68 69 76 77 84 85 92 93 100 101 108 109

5 62 63 70 71 78 79 86 87 94 95 102 103 110 111

114

6 57 64 65 72 73 80 81 88 89 96 97 104 105 112 113

0 58 59 66 67 74 75 82 83 90 91 98 99 106 107

2 60 61 68 69 76 77 84 85 92 93 100 101 108 109

4 62 63 70 71 78 79 86 87 94 95 102 103 110 111

115

5 57 64 65 72 73 80 81 88 89 96 97 104 105 112 113

7 59 66 67 74 75 82 83 90 91 98 99 106 107

1 60 61 68 69 76 77 84 85 92 93 100 101 108 109

3 62 63 70 71 78 79 86 87 94 95 102 103 110 111

116

4 57 64 65 72 73 80 81 88 89 96 97 104 105 112 113

6 59 66 67 74 75 82 83 90 91 98 99 106 107 114 115

0 60 61 68 69 76 77 84 85 92 93 100 101 108 109

2 62 63 70 71 78 79 86 87 94 95 102 103 110 111

117

3 57 64 65 72 73 80 81 88 89 96 97 104 105 112 113

5 59 66 67 74 75 82 83 90 91 98 99 106 107 114 115

7 61 68 69 76 77 84 85 92 93 100 101 108 109

1 62 63 70 71 78 79 86 87 94 95 102 103 110 111

118

2 57 64 65 72 73 80 81 88 89 96 97 104 105 112 113

4 59 66 67 74 75 82 83 90 91 98 99 106 107 114 115

6 61 68 69 76 77 84 85 92 93 100 101 108 109 116 117

0 62 63 70 71 78 79 86 87 94 95 102 103 110 111

119

1 57 64 65 72 73 80 81 88 89 96 97 104 105 112 113

3 59 66 67 74 75 82 83 90 91 98 99 106 107 114 115

5 61 68 69 76 77 84 85 92 93 100 101 108 109 116 117

7 63 70 71 78 79 86 87 94 95 102 103 110 111

120

0 57 64 65 72 73 80 81 88 89 96 97 104 105 112 113

2 59 66 67 74 75 82 83 90 91 98 99 106 107 114 115

4 61 68 69 76 77 84 85 92 93 100 101 108 109 116 117

6 63 70 71 78 79 86 87 94 95 102 103 110 111 118 119

\*\*\*\*\*\*\*\*\*\*\* Average Statistics \*\*\*\*\*\*\*\*\*\*\*

=======================================

61 customers waited for an average of 28 ticks.

The longest time a customer waited was 56 ticks.

60 customers remain in the lines.

--------------------------------

Process exited after 13.32 seconds with return value 0

Press any key to continue . . .