GIT Department of Computer Engineering
CSE 344 - Spring 2020
Homework 4 Report

Esra EMIRLI
151044069

May 24, 2020

Problem Solution Approach

A structure called ingredients was created to hold 6 chef's unlimited ingredients. In addition to flour, milk, walnuts, sugar ingredients, a variable named i, which indicates the chef number, was also kept. In addition, the material to be delivered as added in the homework pdf was added to the ingredients as a dynamic array (*ingp*) kept in heap.

Semaphores that names mutex for chefs and mainmutex for wholesaler, to be used have been defined globally to access all threads.

First, argument checks were made with **getopt**. Causes of errors:

1. Missing argument
2. Unknown option (except i)
3. More arguments

Then, the **file was checked**. To read the line, 3 bytes (ingredient-ingredient-newline) were read byte byte. Causes of errors:

1. If the number of lines is less than 10.
2. If a letter other than the letters M, S, F, W is read
3. If there are more than two ingredients in one line (SMW)
4. If there is 1 ingredient in a row (S)
5. If the same two ingredients are given in a row (SS)
6. If there is a line left blank according to "*two letters at each line*" in the report

An error message was given by specifying the line number.

In case there is no error in the file,

Table 1: Unlimited material distributions

Chef	**Flour**	**Walnuts**	**Sugar**	**Milk**
1	1	1	0	0
2	1	0	1	0
3	1	0	0	1
4	0	1	1	0
5	0	1	0	1
6	0	0	1	1

After each chef's ingredient assignments, a thread was created for all of them and these unlimited ingredients were sent to the thread function as a parameter.

Since all the ingredients in **ingp**, which holds the ingredients to be read and sent from the file, are empty, even if the threads run, they cannot exceed the conditions.

When entered into the appropriate thread:

- necessary commands are printed on the screen
- The ingredients sent in **ingp** are reset.
- Dessert preparation time is set randomly and sleeped for that time.
- When the dessert is ready, the wholesaler mutex is posted and delivered.
- Finally, the chef also posts and exits his own mutex.

Wholesaler The wholesaler who receives the post from the chef continues to read the file and refills the incoming ingredients and **ingp**. Then the wholesaler mutex is put on hold and the appropriate one of the threads is provided to work.

This process continues until the file reading is finished. When the file is finished, the globally defined **lcv** variable is changed to 0 and the loop of the thread is broken.

With Join, the threads are expected to be finished. *mutex (chefs)* , *main-mutex (wholesaler)* are destroyed. ing deallocate is done and the program is terminated.