AOD SHABU

นาย กิตติพิชญ์ ฟองจันทร์	**66360504**
นาย ภูดิศ พุ่มจันทร์	**66363932**
นาย อมรเทพ ไทยศรี	**66365806**
นาย ชัยวัฒน์ พยุงวงค์	**66361150**
นาย คณาธิป เมื่อเกิด	**66360658**
นาย นุติพงษ์ เกิดด่าน	**66362928**
นาย บวรเดช ชินประภาพ	**66362959**
นาย สุวิจักขณ์ สุวิทยาภรณ์	**66365530**
นาย ปิติพงศ์ จิตอารี	**66363222**

รายงานนี้เป็นส่วนหนึ่งของวิชา Data Structures and Algorithms

ปีการศึกษา 2567

ลิขสิทธิ์เป็นของกลุ่ม 1 Elon Aod

ชื่อรายงาน: โปรเจคระบบหลังบ้าน AOD SHABU

ผู้จัดทำ: กลุ่ม 1 Elon Aod

บทคัดย่อ

รายงานนี้เกี่ยวกับการพัฒนาระบบหลังบ้านของร้าน AOD SHABU ซึ่งใช้ภาษา C++ ในการพัฒนาและทำงานผ่าน Terminal ระบบนี้ถูกสร้างขึ้นมาเพื่อช่วยจัดการจองโต๊ะ จัดการสต็อกวัตถุดิบ และคิวของอาหารแต่ละโต๊ะ ระบบออกแบบให้ใช้งานง่ายและช่วยให้เจ้าของร้านและพนักงานทำงานได้สะดวกยิ่งขึ้น

การพัฒนาระบบนี้เน้นไปที่การจัดการข้อมูลในร้านและการแสดงผลที่เข้าใจง่ายผ่าน Terminal ทำให้สามารถควบคุมการทำงานต่าง ๆ ได้อย่างมีประสิทธิภาพ โดยเฉพาะในด้านการตรวจสอบสต็อกและจองโต๊ะที่สามารถทำได้ทันทีผ่านคำสั่งใน Terminal

สารบัญ

ภาพรวมโปรเจค

1.1 ระบบจองโต๊ะ (Arrays)

ใช้ Arrays เพื่อดำเนินการ ในกรณีที่โต๊ะว่าง ให้กำหนดสถานะของโต๊ะเป็น "0" และหากโต๊ะจองหรือใช้งานอยู่ ให้กำหนดสถานะของโต๊ะเป็น "1" อีกทั้งยังสามารถตรวจสอบสถานะของโต๊ะแลแสดงผลลัพธ์ออกมาได้ด้วย

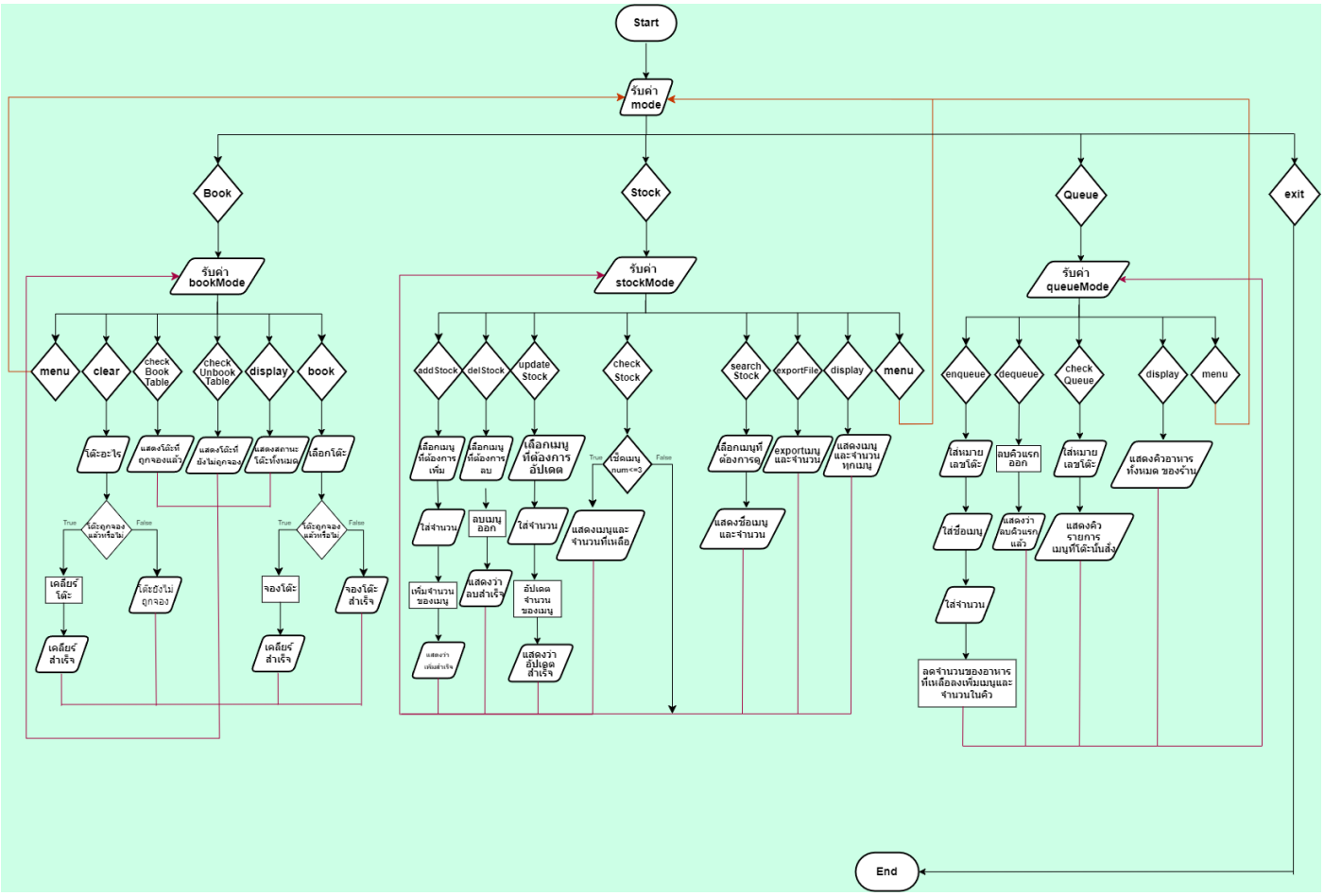
1.2 ระบบจัดการ Stock (create,sorting,search,insert,delete)

เราสามารถดำเนินการเพิ่ม (insert), ลบ (delete), อัปเดต (updateStock), เช็ค (checkStock), และค้นหา (search) สต็อกอาหารได้ นอกจากนี้ยังสามารถส่งออกไฟล์ในรูปแบบข้อความ .txt และไฟล์นี้จะมีการใช้การเรียงลำดับ (sorting) เพื่อจัดเรียงจำนวนอาหารที่เหลือในสต็อกจากน้อยไปมากด้วย

1.3 ระบบจัดการ Queue (enqueue,dequeue)

มีการเพิ่มข้อมูลในคิว (enqueue) ลบข้อมูลออกจากคิว (dequeue) การตรวจสอบคิว (checkQueue) และการแสดงผลลัพธ์ออกมา

Flowchart



https://drive.google.com/file/d/10TA6MUEQ_jFqUGqE9scZrZe9N4H2jaCs/view

การอธิบาย Code

Array.h

```
#ifndef  ARRAY_HPP
#define ARRAY_HPP

#include <iostream>

using namespace std;

class Array{
    protected:
        static const int max_size = 50;
        int arr[max_size];
        int size = 0;
    public:
        Array(){
            size = 0;
        }
```

static const int max_size = 50; : กำหนดขนาดสูงสุดของอาเรย์ (50) ซึ่งเป็นค่าคงที่ที่ไม่สามารถเปลี่ยนแปลงได้

int arr[max_size]; : อาเรย์ที่เก็บข้อมูลประเภท int

int size = 0; : ตัวแปรที่เก็บขนาดปัจจุบันของอาเรย์ (จำนวนของข้อมูลที่ใช้)

Array() : Constructor ของคลาสที่ใช้ในการเริ่มต้นตัวแปร size เป็น 0 เมื่อลงโปรแกรม เพื่อให้แน่ใจว่าอาเรย์เริ่มต้นที่ว่างเปล่า

```
    int getSize(){
        return size;
    }

    int getValue(int index){
        return arr[index];
    }

    void setValue(int index,int value){
        arr[index] = value;
    }

    void insert(int value){
        arr[size] = value;
        size ++;
    }
```

int getSize() : คืนค่าขนาดปัจจุบันของอาเรย์

int getValue(int index) : คืนค่าของอาเรย์ที่ตำแหน่ง index

void setValue(int index, int value) : ตั้งค่าให้กับอาเรย์ที่ตำแหน่ง index

void insert(int value) : แทรก value เข้าไปในอาเรย์ที่ตำแหน่ง size ปัจจุบัน แล้วเพิ่มค่าของ size ขึ้น 1

```
void display(){
    for (int i = 0; i < size; i++){
        cout << arr[i] << "  ";
    }
    cout << endl;
}

int search(int value){
    bool isfound = false;
    for (int i = 0; i < size; i++){
        if (value == arr[i]){
            return i;
            isfound = true;
            break;
        }
    }
    if (!isfound){
        return -1;
    }
}
```

void display() : แสดงข้อมูลทั้งหมดในอาเรย์ โดยพิมพ์ค่าของแต่ละตำแหน่งในอาเรย์

int search(int value) : ค้นหาตำแหน่งของ value ในอาเรย์ ถ้าพบจะคืนดัชนีที่พบ ถ้าไม่พบจะคืน -1

```
void del(int value){
    int index = search(value);
    for (int i = index; i < size-1; i++){
        arr[i] = arr[i+1];
    }
    size--;
    display();
}
```

void del(int value) : ลบข้อมูลที่มีค่า value

```
void selectionsort() {
    int min, tempindex;
    for (int i = 0; i < size - 1; i++) {
        cout << "I = " << i << endl;
        min = arr[i];
        tempindex = i;

        for (int j = i + 1; j < size; j++) {
            if (arr[j] < min) {
                min = arr[j];
                tempindex = j;
            }
        }

        if (tempindex != i) {
            arr[tempindex] = arr[i];
            arr[i] = min;
        }

        display();
    }
}
```

void selectionsort() : ใช้วิธี Selection Sort เพื่อจัดเรียงข้อมูลในอาเรย์

```
void insertionsort() {
    int tempdata;
    for (int i = 1; i < size; i++) {
        cout << "I = " << i << endl;
        for (int j = i; j > 0; j--) {
            if (arr[j] < arr[j - 1]) {
                tempdata = arr[j];
                arr[j] = arr[j - 1];
                arr[j - 1] = tempdata;
            }
        }
        display();
    }
}
```

void insertionsort() : ใช้วิธี Insertion Sort เพื่อจัดเรียงข้อมูลในอาเรย์

```
void bubblesort(){
    bool isswap;
    for (int i = 0; i < size; i++){
        cout << "I = " << i << endl;
        isswap = false;
        for (int j = 0; j < size-i-1; j++){
            if (arr[j+1] < arr[j]){
                int tmp;
                tmp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = tmp;
                isswap = true;
            }
        }

        display();
        if(!isswap){
            break;
        }
    }
}
```

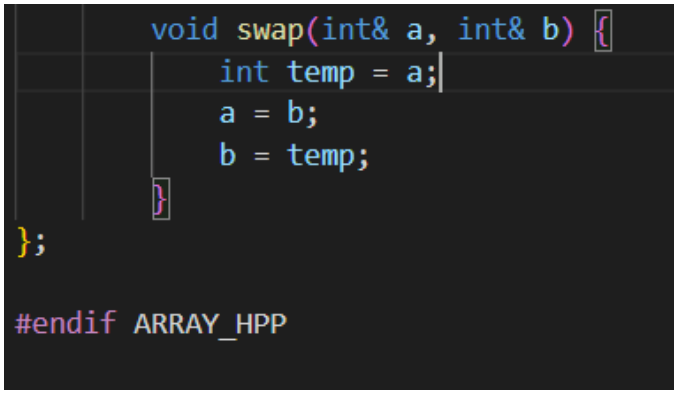
void bubblesort() : ใช้วิธี Bubble Sort

```
void quicksort(int left, int right) {
    if (left < right) {
        int pivotIndex = partition(left, right);
        quicksort(left, pivotIndex - 1);
        quicksort(pivotIndex + 1, right);
    }
}

int partition(int left, int right) {
    int pivot = arr[right];
    int i = left - 1;
    for (int j = left; j < right; j++) {
        if (arr[j] < pivot) {
            i++;
            swap(arr[i], arr[j]);
        }
    }
    swap(arr[i + 1], arr[right]);
    return i + 1;
}
```

void quicksort(int left, int right) : ใช้ Quick Sort ในการจัดเรียงข้อมูลในอาเรย์ โดยการเรียงลำดับซ้ายและขวาของพาร์ติชันที่แบ่งตาม pivot

int partition(int left, int right) : แบ่งพาร์ติชันของข้อมูลตามค่า pivot โดยใช้ตำแหน่งของค่า pivot เป็นเกณฑ์ในการจัดเรียงข้อมูล



```
        void swap(int& a, int& b) {
            int temp = a;
            a = b;
            b = temp;
        }
};

#endif ARRAY_HPP
```

void swap(int& a, int& b) : ฟังก์ชันที่ใช้ในการสลับค่าของสองตัวแปร a และ b

ArraySI

```
#ifndef ARRAYSI_HPP
#define ARRAYSI_HPP

#include <iostream>

using namespace std;

class ArraySI{
    protected:
        static const int max_size = 30;
        string arrs[max_size];
        int arri[max_size];
        int size = 0;
    public:
        ArraySI(){
            size = 0;
        }
```

static const int max_size = 30; : กำหนดขนาดสูงสุดของอาเรย์ (30) ซึ่งเป็นค่าคงที่ที่ไม่สามารถเปลี่ยนแปลงได้

string arrs[max_size]; : อาเรย์ที่เก็บข้อมูลประเภท string ซึ่งใช้สำหรับเก็บข้อความหรือคำอธิบาย
int arri[max_size]; : อาเรย์ที่เก็บข้อมูลประเภท int ซึ่งใช้สำหรับเก็บค่าเลขที่เกี่ยวข้องกับแต่ละสตริงใน arrs
int size = 0; : ตัวแปรเพื่อเก็บขนาดปัจจุบันของข้อมูลที่มีในอาเรย์

ArraySI() : Constructor ของคลาสที่ใช้ในการเริ่มต้นตัวแปร size เป็น 0 เมื่อลงโปรแกรม เพื่อให้แน่ใจว่าอาเรย์เริ่มต้นที่ว่างเปล่า

```
public:
    ArraySI(){
        size = 0;
    }

    int getValue(string data){
        return arri[search(data)];
    }

    void setValue(string data,int value){
        arri[search(data)] = value;
    }

    void insert(string text, int value){
        arrs[size] = text;
        arri[size] = value;
        size++;
    }
```

int getValue(string data) : คืนค่าที่เก็บใน arri ตามค่าของ data โดยการค้นหาด้วย search(data)

void setValue(string data, int value) : ตั้งค่าใหม่ใน arri ตาม data โดยการค้นหาตำแหน่งด้วย search(data) และอัปเดตค่าที่ตำแหน่งนั้น

void insert(string text, int value) : แทรก text และ value เข้าไปในอาเรย์ที่ตำแหน่ง size แล้วเพิ่มค่าของ size ขึ้น 1

```
void display(){
    for (int i = 0; i < size; i++){
        cout << arrs[i] << " : " << arri[i] << endl;
    }
}

int search(string data){
    for(int i = 0; i < size; i++){
        if (arrs[i] == data){
            return i;
        }
    }
    return -1;
}
```

void display() : แสดงข้อมูลทั้งหมดในอาเรย์ โดยพิมพ์คู่ของ string และ int ออกมาทีลคู่

intsearch(string data): ค้นหาตำแหน่งของ data ในอาเรย์ arrs ถ้าพบจะคืนดัชนีที่พบ ถ้าไม่พบจะคืน -1

```
void del(string data){
    int index = search(data);
    if (index != -1) {
        for (int i = index; i < size - 1; i++){
            arrs[i] = arrs[i + 1];
            arri[i] = arri[i + 1];
        }
        size--;
    }
    display();
}
```

void del(string data) : ลบข้อมูลที่มี data โดยการย้ายข้อมูลที่ตามหลังไปตำแหน่งก่อนหน้า จากนั้นลดขนาดของอาเรย์ และแสดงผลลัพธ์หลังจากการลบ

```
void bubblesort(){
    bool isswap;
    for (int i = 0; i < size; i++){
        isswap = false;
        for (int j = 0; j < size - i - 1; j++){
            if (arri[j + 1] < arri[j]){
                swap(arrs[j], arrs[j + 1]);
                swap(arri[j], arri[j + 1]);
                isswap = true;
            }
        }
        if (!isswap){
            break;
        }
    }
}
```

void bubblesort() : ใช้วิธี Bubble Sort เพื่อจัดเรียงข้อมูลในอาเรย์ตามค่าของ arri โดยสลับตำแหน่งของข้อมูลจนกว่าข้อมูลจะเรียงลำดับ

```
        void quicksort(int left, int right){
            if (left < right) {
                int pivotIndex = partition(left, right);
                quicksort(left, pivotIndex - 1);
                quicksort(pivotIndex + 1, right);
            }
        }
        int partition(int left, int right){
            int pivot = arri[right];
            int i = left - 1;
            for (int j = left; j < right; j++){
                if (arri[j] <= pivot){
                    i++;
                    swap(arrs[i], arrs[j]);
                    swap(arri[i], arri[j]);
                }
            }
                swap(arrs[i + 1], arrs[right]);
                swap(arri[i + 1], arri[right]);
                return i + 1;
            }
};
#endif ARRAYSI_HPP
```

void quicksort(int left, int right) : ใช้ Quick Sort ในการจัดเรียงข้อมูลในอาเรย์ โดยการเรียงลำดับส่วนที่แบ่งพาร์ติชัน

int partition(int left, int right) : แบ่งพาร์ติชันของข้อมูลตามค่า pivot โดยใช้ตำแหน่งของค่า pivot เป็นเกณฑ์ในการจัดเรียงข้อมูล

ArrayISI

```hpp
#ifndef ARRAYISI_HPP
#define ARRAYISI_HPP

#include <iostream>

using namespace std;

class ArrayISI{
    protected:
        static const int max_size = 50;
        int arri1[max_size];
        string arrs[max_size];
        int arri2[max_size];
        int size = 0;
    public:
        ArrayISI(){
            size = 0;
        }
```

static const int max_size = 50; : กำหนดขนาดสูงสุดของอาเรย์ (30) ซึ่งเป็นค่าคงที่ที่ไม่สามารถเปลี่ยนแปลงได้

int arri1[max_size]; : อาเรย์ที่เก็บข้อมูลประเภท int ซึ่งใช้สำหรับเก็บค่าเลขที่เกี่ยวข้องกับแต่ละสตริงใน arrs

string arrs[max_size]; : อาเรย์ที่เก็บข้อมูลประเภท string ซึ่งใช้สำหรับเก็บข้อความหรือคำอธิบาย

int arri2[max_size]; : อาเรย์ที่เก็บข้อมูลประเภท int ซึ่งใช้สำหรับเก็บค่าเลขที่เกี่ยวข้องกับแต่ละสตริงใน arrs

int size = 0; : ตัวแปรเพื่อเก็บขนาดปัจจุบันของข้อมูลที่มีในอาเรย์

ArrayISI() : Constructor ของคลาสที่ใช้ในการเริ่มต้นตัวแปร size เป็น 0 เมื่อลงโปรแกรม เพื่อให้แน่ใจว่าอาเรย์เริ่มต้นที่ว่างเปล่า

```
        void insert(int value1, string  data, int value2){
            arri1[size] = value1;
            arrs[size] = data;
            arri2[size] = value2;
            size ++;
        }
```

void insert(int value1, string data, int value2) : แทรก text และ value เข้าไปในอาเรย์ที่ตำแหน่ง size แล้วเพิ่มค่าของ size ขึ้น 1

```
        void display(){
            for (int i = 0; i < size; i++){
                cout << arri1[i] << " : " << arrs[i] << " : " << arri2[i];
            }
            cout << endl;
        }

};
#endif ARRAYISI_HPP
```

void display() : แสดงข้อมูลทั้งหมดในอาเรย์ โดยพิมพ์คู่ของ string และ int ออกมาทีละคู่

Booktable

```
class bookTables : public Array {
public:
    bookTables(int table) {
        size = 0;
        for (int i = 0; i <= table-1; i++) {
            insert(0);  // Initialize all tables as unbooked (0)
        }
    }
```

Class bookTables : เป็นคลาสที่ถูกสร้างขึ้นมาเพื่อจัดการการจองโต๊ะ โดยมาจากคลาส Array
constructor bookTables (int table) : เป็นฟังก์ชันสร้างคลาส bookTables ที่รับค่าจาก table เข้ามาเป็นจำนวนโต๊ะทั้งหมด
size = 0 : กำหนดขนาดของ array เป็น 0 ในตอนแรก
for loop : ใช้for loop ลูปเพื่อเพิ่มข้อมูลเข้าไปใน array จำนวน table ครั้ง โดยแต่ละครั้งจะใส่ค่า 0 เข้าไปแสดงถึงโต๊ะที่ว่างอยู่ (0 แทนโต๊ะที่ว่าง, 1 แทนโต๊ะที่จอง)
insert(0): เป็นฟังก์ชันที่ถูกเรียกใช้จากคลาส Array เพื่อเพิ่มข้อมูลเข้าไปใน array

```
void books(int index) {
    if (arr[index - 1] == 0) {
        arr[index - 1] = 1;
        cout << "Book Success!!" << endl;
    } else {
        cout << "That table is booked already!!" << endl;
    }
}
```

void books(int index) : ฟังก์ชันที่ใช้ในการจองโต๊ะ.
if (arr [index - 1] == 0) : ตรวจสอบว่าตารางที่ระบุโดย index ยังไม่ถูกจอง (0 หมายถึงไม่ถูกจอง)
arr [index - 1] = 1; : ตั้งค่าโต๊ะที่ระบุเป็นจอง (1 หมายถึงจองแล้ว)
cout << " Book Success!! " << endl; : แสดงข้อความว่า "จองสำเร็จ"
else : ถ้าโต๊ะถูกจองแล้ว, แสดงข้อความว่า "โต๊ะนี้ถูกจองแล้ว"

checkbookTable

```
void checkbookTable(){
    for (int i = 0; i < size; i++){
        if(arr[i] == 1){
            cout << "Table " << i+1 << " : Booked" << endl;
        }
        else{
            continue;
        }
    }
}
```

void checkBookTable() : ฟังก์ชันสำหรับตรวจสอบสถานะของโต๊ะที่ถูกจอง
for (int i = 0; i < size; i++): ลูปผ่านทุกโต๊ะ
if (arr[i] == 1): ตรวจสอบว่าโต๊ะถูกจอง
cout << "Table " << i + 1 << " : Booked" << endl; : พิมพ์ข้อความสถานะของโต๊ะ

checkunbookTable

```
void checkunbookTable(){
    for (int i = 0; i < size; i++){
        if(arr[i] == 0){
            cout << "Table " << i+1 << " : unbooked" << endl;
        }
        else{
            continue;
        }
    }
}
```

void checkunbookTable() : ฟังก์ชันสำหรับตรวจสอบสถานะของโต๊ะที่ยังไม่ถูกจอง
for (int i = 0; i < size; i++): ลูปผ่านทุกโต๊ะ
if (arr[i] == 0): ตรวจสอบว่าโต๊ะยังไม่ถูกจอง
cout << "Table " << i + 1 << " : unbooked" << endl; : พิมพ์ข้อความสถานะของโต๊ะ

display

```
void display(){
    for(int i = 0; i < size; i++){
        string tableStatus;
        if (arr[i] == 0){
            tableStatus = "Unbooked";
        }
        if (arr[i] == 1){
            tableStatus = "Booked";
        }
        cout << "Table " << i+1 << " : " << tableStatus << endl;
    }
}
```

void display() : ฟังก์ชันสำหรับแสดงสถานะของทุกโต๊ะ

for (int i = 0; i < size; i++): ลูปผ่านทุกโต๊ะ

string tableStatus = (arr[i] == 0) ? "Unbooked" : "Booked"; : ใช้การเลือกเงื่อนไข (ternary operator) เพื่อกำหนดสถานะของโต๊ะ

cout << "Table " << i + 1 << " : " << tableStatus << endl; : พิมพ์สถานะของโต๊ะ

Stock

Class Stock

```
class Stock : public ArraySI {
public:

```

คลาส Stock เป็นคลาสที่สืบทอดจากคลาส ArraySI ซึ่งจะใช้ฟังก์ชันและตัวแปรของ ArraySI เพื่อจัดการกับสต็อกของสินค้า

stockcheck

```
void stockcheck(){
    for (int i = 0; i < size; i++){
        if (arri[i] <= 3){
            cout << arrs[i] << " Has only " << arri[i] << " stock left. Please top up quickly." << endl;
        }
    }
}
```

ฟังก์ชัน stockcheck ทำการตรวจสอบสต็อกสินค้าทั้งหมด หากสินค้าชนิดใดมีสต็อกเหลือน้อยกว่า 4 ชิ้น (เช่น 3 ชิ้นหรือน้อยกว่า) จะแสดงข้อความเตือนว่าให้เติมสต็อกสินค้านั้น

Function importfile

```
void importfile(string filename){
    string name;
    int value;
    ifstream file(filename);
    if (file.is_open()){
            while (file >> name >> value){
                insert(name,value);
        }
    }
    else{
        cout << "Cant read file" << endl;
    }
    file.close();
}
```

Importfile(string filename) : ฟังก์ชั่นสำหรับการนำเข้าข้อมูลในฐานระบบจากไฟล์ที่ระบุในฟังก์ชั่น

Function exportfile

```
void exportfile(){
    ofstream file("Stock.txt");
    for (int i = 0;i < size; i++){
        quicksort(0,size-1);
        file << arrs[i] << " : " << arri[i];
        if(arri[i] <= 3){
            file << ": There is little stock left. Please top up quickly.";
        }
        file <<endl;
    }
    file.close();
    cout << "Export succesfully!! Please check the file"<<endl;
}
;
```

Exportfile() : ฟังก์ชั่นสำหรับการส่งออกข้อมูล โดยจะทำการ Sorting ก่อน และตรวจสอบว่ามีสต็อคไหนน้อยกว่า 3 ถ้ามีให้แจ้งว่า There is little stock left. Please top up quickly และส่งออกไฟล์ในไฟล์ Stock.txt

Foodqueue

```
#include <iostream>
#include "Header/ArrayISI.h"

using namespace std;

class foodQueue: public ArrayISI{
    public:
        void enqueue(int value1, string data, int value2){
            insert(value1,data,value2);
        }

        int dequeue(){
            int front = arri1[0];
            for (int i = 0; i < size; i++){
                arri1[i] = arri1[i+1];
                arrs[i] = arrs[i+1];
                arri2[i] = arri2[i+1];
            }
            if(size == 0){
                cout << "NO QUEUE" << endl;
            }
            cout << "Queue for Table : " << front << " is derivered." << endl;
            size--;
            return front;
        }
```

enqueue (int value1, string data, int value2) : ฟังก์ชั่นสำหรับการเพิ่มข้อมูลในรูปแบบ queue โดยจะ insert ข้อมูล value1,data และ value2 ที่ตำแหน่งท้ายสุด

dequeue () คือการลบข้อมูลในรูปแบบ queue โดยจะลบข้อมูลตำแหน่งแรกออกก่อน หากไม่มีคิวจะแจ้งว่า No queue แต่หากมี จะทำการแจ้งว่าคิวสำหรับโต๊ะนั้นส่งออกไปแล้ว

```
void display(){
    for(int i = 0; i < size; i++){
        cout << "Table " << arri1[i] << " : " << arrs[i] << " X" << arri2[i] << endl;
    }
    if(size == 0){
        cout << "NO QUEUE" << endl;
    }
}

void checkQueue(int x){
    for(int i = 0; i < size; i++){
        if (arri1[i] == x){
            cout << "Table " << arri1[i] << " : " << arrs[i] << " X" << arri2[i] << endl;
        }
    }
}
```

display() : ฟังก์ชั่นที่ใช้สำหรับแสดงคิว หากไม่มีจะแจ้งว่า NO QUEUE

checkQueue(int x) : ฟังก์ชั่นที่ใช้เช็คคิวของโต๊ะที่ระบุค่าไป

MAIN

ส่วนประกาศและการเตรียมค่าเริ่มต้น

```cpp
#include <iostream>
#include "bookTables.cpp"
#include "stock.cpp"
#include "foodQueue.cpp"

using namespace std;

bookTables t = bookTables(5);
Stock s;
foodQueue q;

void initComponent(){
    s.importfile("test.txt");
}
```

#include <iostream> ใช้เพื่อรับและแสดงข้อมูลผ่านหน้าจอคอนโซล bookTables.cpp, stock.cpp, และ foodQueue.cpp ซึ่งแต่ละไฟล์มีคลาสที่ใช้ในการจัดการโต๊ะ, สต็อกอาหาร และคิวอาหาร bookTables โดยกำหนดให้มี 5 โต๊ะ, สร้างอ็อบเจกต์ของ Stock และ foodQueue initComponent จะทำการนำเข้าข้อมูลจากไฟล์ test.txt ซึ่งเป็นข้อมูลสต็อกเบื้องต้น

ฟังก์ชัน main

```
int main(){
    string mode;
    string text;

    initComponent();

    while (true){
        cout << "Select mode (book, stock, queue, exit) : ";
        cin >> mode;
        if (mode == "book"){
            while (true){
                cout << "Enter a command (book, clear, checkBookTable, checkUnbookTable, display, menu): ";
                cin >> text;
```

main จะเป็นลูปหลักที่คอยรับคำสั่งจากผู้ใช้

ผู้ใช้สามารถเลือกโหมดการทำงาน 4 แบบ คือ book (จองโต๊ะ), stock (จัดการสต็อก), queue (จัดการคิวอาหาร), และ exit (ออกจากโปรแกรม)

โหมด book (การจองโต๊ะ)

```
    if (mode == "book"){
        while (true){
            cout << "Enter a command (book, clear, checkBookTable, checkUnbookTable, display, menu): ";
            cin >> text;

            if (text == "book"){
                int x;
                cout << "Enter the table number to book: ";
                cin >> x;
                if ( x > t.getSize() or x == 0){
                    cout << "We don't have that table number." << endl;
                }
                else{
                    t.books(x);
                }
            }
            else if (text == "clear"){
                int x;
                cout << "Enter the table number to clear: ";
                cin >> x;
                if( x > t.getSize() or x == 0){
                    cout << "We don't have that table number." << endl;
                }
                else{
                    t.clear(x);
                }
            }
            else if(text == "checkBookTable"){
                t.checkbookTable();
            }

            else if (text == "checkUnbookTable"){
                t.checkunbookTable();
            }
            else if (text == "display"){
                t.display();
            }
            else if (text == "menu"){
                break;
            }
        }
    }
```

รับคำสั่งภายในโหมด book เช่น book เพื่อจองโต๊ะ, clear เพื่อเคลียร์โต๊ะ, checkBookTable เพื่อตรวจสอบโต๊ะที่ถูกจองแล้ว, checkUnbookTable เพื่อตรวจสอบโต๊ะที่ยังว่างอยู่, display เพื่อแสดงสถานะของโต๊ะ, และ menu เพื่อกลับไปยังเมนูหลัก

โหมด stock (การจัดการสต็อกอาหาร)

```
if (mode == "stock"){
    while (true){
        cout << "Enter a command (addStock, delStock, updateStock checkStock, searchStock, exportfile, display, menu) : " << endl;
        cin >>  text;
        if (text == "addStock"){
            string name;
            int value;
            cout << "What do you want to add : ";
            cin >> name;
            if (s.search(name) != -1){
                cout << "That menu is already exist" << endl;
            }
            else{
                cout << "How many do you add : ";
                cin >> value;
                s.insert(name,value);
                cout << "Add Success!!"<<endl;
            }
        }

        else if (text == "delStock"){
            string name;
            int value;
            cout << "What do you want to delete : ";
            cin >> name;
            if (s.search(name) == -1){
                cout << "That menu is not available in stock." << endl;
            }
            else{
                s.del(name);
                cout << "Delete Success!!"<<endl;
            }
        }

        else if (text == "updateStock"){
            string name;
            int value;
            cout << "What do you want to update : ";
            cin >> name;
            if (s.search(name) == -1){
                cout << "That menu is not available in stock." << endl;
            }
            else{
                cout << "How many do you update : ";
                cin >> value;
                s.setValue(name,value);
                cout << "Update Success!!"<<endl;
            }
        }

        else if (text == "checkStock"){
            s.stockcheck();
        }

        else if (text == "searchStock"){
            string name;
            cout << "Which food would you like to look into? : ";
            cin >> name;
            if (s.search(name) == -1){
                cout << "That menu is not available in stock." << endl;
            }
            else{
                cout << name << " : " << s.getValue(name) << endl;
            }
        }

        else if (text == "exportfile"){
            s.exportfile();
        }
        else if (text == "display"){
            s.display();
        }
        else if (text == "menu"){
            break;
        }
    }
}
```

รับคำสั่งภายในโหมด stock เช่น addStock เพื่อเพิ่มเมนูอาหารใหม่, delStock เพื่อลบเมนู, updateStock เพื่ออัปเดตจำนวนสต็อก, checkStock เพื่อตรวจสอบสต็อกทั้งหมด, searchStock เพื่อค้นหาเมนู, exportfile เพื่อส่งออกข้อมูลสต็อก, display เพื่อแสดงสต็อกทั้งหมด, และ menu เพื่อกลับไปยังเมนูหลัก

โหมด queue (การจัดการคิวอาหาร)

```
if (mode == "queue"){

    while (true){
        cout << "Enter a command (enqueue, dequeue, checkQueue, display, menu) : ";
        cin >>  text;

        if (text == "enqueue"){
            int table;
            string menu;
            int number;
            cout << "Table : ";
            cin >> table;
            if(t.getValue(table-1) == 0){
                cout << "This table is currently not booked." << endl;
            }
            else{
                cout << "Menu : ";
                cin >> menu;
                if(s.search(menu) == -1){
                    cout <<  "That menu is not available in stock." <<endl;
                }
                else{
                    cout << "Number :";
                    cin >> number;
                    if (s.getValue(menu) < number){
                        cout << "There is not enough stock on that menu." <<endl;
                    }
                    else{
                        q.enqueue(table,menu,number);
                        s.setValue(menu,s.getValue(menu)-number);
                    }
                }

            }
        }

        else if (text == "dequeue"){
            q.dequeue();
        }

        else if (text == "checkQueue"){
            int x;
            cout << "Which table would you like to look into? : ";
            cin >> x;
            if(t.getValue(x-1) == 0){
                cout << "This table is currently not booked." << endl;
            }
            else{
                q.checkQueue(x);
            }
        }

        else if(text ==  "display"){
            q.display();
        }
        else if(text == "menu"){
            break;
        }
    }
}
```

รับคำสั่งภายในโหมด queue เช่น enqueue เพื่อเพิ่มรายการอาหารในคิวสำหรับโต๊ะที่จองแล้ว, dequeue เพื่อเอารายการอาหารออกจากคิว, checkQueue เพื่อตรวจสอบคิวของโต๊ะ, display เพื่อแสดงคิวทั้งหมด, และ menu เพื่อกลับไปยังเมนูหลัก

โหมด exit (การออกจากโปรแกรม)

```
        if (mode == "exit"){
            cout << "See you again next time"<<endl;
            break;
        }

    }

    return 0;
```

เมื่อต้องการออกจากโปรแกรม ระบบจะแสดงข้อความ "See you again next time" และสิ้นสุดการทำงาน

ตัวอย่างการรันโปรแกรม

1.Create

```
bookTables t = bookTables(5);
Stock s;
foodQueue q;
```

2.Search

```
Enter a command (book, clear, checkBookTable, checkUnbookTable, display, menu): checkBookTable
Table 1 : Booked
Enter a command (book, clear, checkBookTable, checkUnbookTable, display, menu): checkUnbookTable
Table 2 : unbooked
Table 3 : unbooked
Table 4 : unbooked
Table 5 : unbooked
Enter a command (book, clear, checkBookTable, checkUnbookTable, display, menu): menu
Select mode (book, stock, queue, exit) : stock
Enter a command (addStock, delStock, updateStock checkStock, searchStock, exportfile, display, menu) :
searchStock
What food do you want to check? : Tofu
Tofu : 5
Enter a command (enqueue, dequeue, checkQueue, display, menu) : checkQueue
What table do you check : 1
Table 1 : Tofu X2
```

3.Insert

```
Select mode (book, stock, queue, exit) : stock
Enter a command (addStock, delStock, updateStock checkStock, searchStock, exportfile, display, menu) :
display
Chicken_Thighs : 999999
Shrimp : 2
Tofu : 5
Pork_Belly : 4
Fish_Balls : 11
Enoki_Mushrooms : 7
Napa_Cabbage : 8
Udon_Noodles : 10
Wagyu_Beef : 1
Clams : 12
Spinach : 3
Crab_Sticks : 6
Sweet_Corn : 14
Enter a command (addStock, delStock, updateStock checkStock, searchStock, exportfile, display, menu) :
addStock
What do you want to add : Weeds
How many do you add : 69
Add Success!!
Enter a command (addStock, delStock, updateStock checkStock, searchStock, exportfile, display, menu) :
display
Chicken_Thighs : 999999
Shrimp : 2
Tofu : 5
Pork_Belly : 4
Fish_Balls : 11
Enoki_Mushrooms : 7
Napa_Cabbage : 8
Udon_Noodles : 10
Wagyu_Beef : 1
Clams : 12
Spinach : 3
Crab_Sticks : 6
Sweet_Corn : 14
Weeds : 69
```

```
Enter a command (addStock, delStock, updateStock checkStock, searchStock, exportfile, display, menu) :
display
Chicken_Thighs : 999999
Shrimp : 2
Tofu : 5
Pork_Belly : 4
Fish_Balls : 11
Enoki_Mushrooms : 7
Napa_Cabbage : 8
Udon_Noodles : 10
Wagyu_Beef : 1
Clams : 12
Spinach : 3
Crab_Sticks : 6
Sweet_Corn : 14
Enter a command (addStock, delStock, updateStock checkStock, searchStock, exportfile, display, menu) :
delStock
What do you want to delete : Clams
Chicken_Thighs : 999999
Shrimp : 2
Tofu : 5
Pork_Belly : 4
Fish_Balls : 11
Enoki_Mushrooms : 7
Napa_Cabbage : 8
Udon_Noodles : 10
Wagyu_Beef : 1
Spinach : 3
Crab_Sticks : 6
Sweet_Corn : 14
Delete Success!!
```

```
Enter a command (addStock, delStock, updateStock checkStock, searchStock, exportfile, display, menu) :
display
Chicken_Thighs : 999999
Shrimp : 2
Tofu : 5
Pork_Belly : 4
Fish_Balls : 11
Enoki_Mushrooms : 7
Napa_Cabbage : 8
Udon_Noodles : 10
Wagyu_Beef : 1
Spinach : 3
Crab_Sticks : 6
Sweet_Corn : 14
```

การแบ่งงานภายในทีม

นาย ภูดิศ พุ่มจันทร์	**66363932**	Main/Leader	**20%**
นาย กิตติพิชญ์ ฟองจันทร์	**66360504**	Stock	**10%**
นาย อมรเทพ ไทยศรี	**66365806**	BookTable	**10%**
นาย ชัยวัฒน์ พยุงวงค์	**66361150**	BookTable	**10%**
นาย คณาธิป เมื่อเกิด	**66360658**	Foodqueue	**10%**
นาย นุติพงษ์ เกิดด่าน	**66362928**	Foodqueue	**10%**
นาย บวรเดช ชินประภาพ	**66362959**	Stock	**10%**
นาย สุวิจักขณ์ สุวิทยาภรณ์	**66365530**	Stock	**10%**
นาย ปิติพงศ์ จิตอารี	**66363222**	Foodqueue	**10%**