**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN LẬP TRÌNH TÍNH TOÁN**

**ĐỀ TÀI: 702**

**TÊN ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG ĐẶT MÓN ĂN VÀ THANH TOÁN ĐƠN HÀNG**

**Giáo viên hướng dẫn: PGS. TS. TRẦN HỒ THỦY TIÊN**

**Sinh viên thực hiện:**

**Lê Hoàng Long Lớp: 20TCLC\_Nhat2 NHÓM: 20.12B**

**Nguyễn Đình Hưng Lớp: 20TCLC\_Nhat2 NHÓM: 20.12B**

**Đà Nẵng, 06/2021**

MỤC LỤC

[MỤC LỤC 1](#_Toc74525652)

[MỞ ĐẦU 1](#_Toc74525653)

[1. TỔNG QUAN ĐỀ TÀI 1](#_Toc74525654)

[2. CƠ SỞ LÝ THUYẾT 1](#_Toc74525655)

[2.1. Ý tưởng 1](#_Toc74525656)

[2.2. Cơ sở lý thuyết 1](#_Toc74525657)

[3. TỔ CHỨC CẤU TRÚC DỮ LIỆU VÀ THUẬT TOÁN 3](#_Toc74525658)

[3.1. Phát biểu bài toán 3](#_Toc74525659)

[3.2. Cấu trúc dữ liệu 5](#_Toc74525660)

[3.3. Thuật toán 8](#_Toc74525661)

[3.3.1. Thuật toán tìm kiếm tuyến tính (linear searching) 8](#_Toc74525662)

[3.3.2. Thuật toán nhập mật khẩu tạo kí tự \* 10](#_Toc74525663)

[4. CHƯƠNG TRÌNH VÀ KẾT QUẢ 13](#_Toc74525664)

[4.1. Tổ chức chương trình 13](#_Toc74525665)

[4.1.1. Các hằng số được định nghĩa và thư viện được sử dụng 13](#_Toc74525666)

[4.1.2. Các hàm được xây dựng trong chương trình 14](#_Toc74525667)

[4.1.3 Sơ đồ của một số hàm trong chương trình 15](#_Toc74525668)

[4.2. Ngôn ngữ cài đặt 22](#_Toc74525669)

[4.3. Kết quả 23](#_Toc74525670)

[4.3.1. Giao diện chính của chương trình 23](#_Toc74525671)

[4.3.2. Kết quả thực thi của chương trình 34](#_Toc74525672)

[4.3.3. Nhận xét đánh giá 38](#_Toc74525673)

[5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 39](#_Toc74525674)

[5.1. Kết luận 39](#_Toc74525675)

[5.2. Hướng phát triển 39](#_Toc74525676)

[TÀI LIỆU THAM KHẢO 40](#_Toc74525677)

[PHỤ LỤC 40](#_Toc74525678)

**Danh mục hình vẽ**

[Hình 1. Struct lồng struct trong chương trình 3](#_Toc74522457)

[Hình 2. Danh sách đặc được triển khai như một mảng tĩnh 6](#_Toc74522458)

[Hình 3. Thuật toán tìm kiếm tuyến tính trong hàm priceOfFood 10](#_Toc74522459)

[Hình 4. Sơ đồ làm việc của thuật toán nhập mật khẩu tạo kí tự \* 12](#_Toc74522460)

[Hình 5. Sơ đồ tổ chức của chương trình 13](#_Toc74522461)

[Hình 6. Sơ đồ làm việc của hàm orderFood 16](#_Toc74522462)

[Hình 7. Sơ đồ làm việc của hàm choiceAndQuantity 17](#_Toc74522463)

[Hình 8. Sơ đồ làm việc của hàm addFood 18](#_Toc74522464)

[Hình 9. Sơ đồ làm việc của hàm orderMore 19](#_Toc74522465)

[Hình 10. Sơ đồ làm việc của hàm changeFood 20](#_Toc74522466)

[Hình 11. Sơ đồ làm việc của hàm priceOfFood 21](#_Toc74522467)

[Hình 12. Sơ đồ làm việc của hàm discount 22](#_Toc74522468)

[Hình 13. Giao diện mở đầu của chương trình với phần giới thiệu thông tin đồ án 23](#_Toc74522469)

[Hình 14. Giao diện nhập user name 23](#_Toc74522470)

[Hình 15. Giao diện khi nhập sai user name 23](#_Toc74522471)

[Hình 16. Giao diện khi nhập ID thành công và in nhân viên (staff) trong ngày 24](#_Toc74522472)

[Hình 17. In ra nhân viên (staff) khác trong ngày 24](#_Toc74522473)

[Hình 18. Giao diện khi nhập sai ID 24](#_Toc74522474)

[Hình 19. Giao diện khi đăng nhập thành công, chương trình in ra menu 25](#_Toc74522475)

[Hình 20. Giao diện khi nhập “1”, chương trình bắt đầu yêu cầu nhân viên nhập mã món và số lượng 25](#_Toc74522476)

[Hình 21. Giao diện khi nhập sai mã món hoặc số lượng, yêu cầu nhân viên nhập lại 26](#_Toc74522477)

[Hình 22. Giao diện sau khi tiến trình chọn món kết thúc, nhập số bàn, và nhập lựa chọn 26](#_Toc74522478)

[Hình 23. Giao diện khi đồng ý gọi thêm món. “ORDERED” là phần đã gọi món, “ORDER MORE” là phần nhập thêm món 27](#_Toc74522479)

[Hình 24. Giao diện gọi thêm món thành công 27](#_Toc74522480)

[Hình 25. Giao diện khi nhập sai lựa chọn 28](#_Toc74522481)

[Hình 26. Giao diện khi thay đổi thông tin món ăn. 28](#_Toc74522482)

[Hình 27. Giao diện khi nhập sai số thứ tự gọi món 29](#_Toc74522483)

[Hình 28. Giao diện khi nhập đúng. Chương trình yêu cầu thay đổi mã món và số lượng 29](#_Toc74522484)

[Hình 29. Giao diện khi nhập sai lựa chọn 30](#_Toc74522485)

[Hình 30. Giao diện in ra thông tin tổng tiền và yêu cầu nhập số tiền thanh toán 30](#_Toc74522486)

[Hình 31. Giao diện khi nhập sai số tiền thanh toán 30](#_Toc74522487)

[Hình 32. Giao diện khi nhập thành công số tiền thanh toán 31](#_Toc74522488)

[Hình 33. Giao diện khi nhập số tiền thanh toán lớn hơn tổng tiền. Chương trình hiển thị tiền thừa (change) 31](#_Toc74522489)

[Hình 34. Giao diện khi nhập lựa chọn “n” – không in ra bill 31](#_Toc74522490)

[Hình 35. Giao diện khi nhập lựa chọn “y” – in ra bill với tiền thanh toán bằng tổng tiền 32](#_Toc74522491)

[Hình 36. Giao diện khi nhập lựa chọn “y” – in ra bill với tiền thanh toán lớn hơn tổng tiền 32](#_Toc74522492)

[Hình 37. Giao diện khi có hóa đơn được giảm giá và in bill hóa đơn đó. 33](#_Toc74522493)

[Hình 38. Giao diện đăng xuất 33](#_Toc74522494)

[Hình 39. Giao diện bảng thống kê doanh thu trong ngày làm việc 34](#_Toc74522495)

[Hình 40. Dữ liệu của khách hàng 1 được ghi ra file với phiên làm việc của nhân viên 1 35](#_Toc74522496)

[Hình 41. Dữ liệu của khách hàng 2 được ghi ra file với phiên làm việc của nhân viên 1 35](#_Toc74522497)

[Hình 42. Tổng doanh thu trong ngày của phiên làm việc nhân viên 1 được lưu vào file TotalDay.txt 36](#_Toc74522498)

[Hình 43. Dữ liệu của khách hàng 1 được ghi ra file với phiên làm việc của nhân viên 2 36](#_Toc74522499)

[Hình 44. Dữ liệu của khách hàng 2 được ghi ra file với phiên làm việc của nhân viên 2 37](#_Toc74522500)

[Hình 45. Dữ liệu của khách hàng 3 được ghi ra file với phiên làm việc của nhân viên 2 37](#_Toc74522501)

[Hình 46. Tổng doanh thu trong ngày của phiên làm việc nhân viên 2 được lưu vào file TotalDay.txt 38](#_Toc74522502)

MỞ ĐẦU

Đầu tiên, chúng em xin gửi lời chân thành cảm ơn tới Thạc Sĩ Trần Hồ Thủy Tiên đã giúp đỡ chúng em hoàn thành đồ án cơ sở này. Trong quá trình xây dựng đồ án, chúng em đã gặp nhiều khó khăn nhưng cô đã hướng dẫn, góp ý tận tình cho chúng em về ý tưởng thiết kế, xây dựng khung chương trình, các chức năng hoạt động của chương trình cũng như các kỹ năng xây dựng khác.

Chân thành cảm ơn đến các bạn bè trong lớp đã hỗ trợ chúng mình, giải đáp khúc mắc để có thể hoàn thành đồ án này.

Mặc dù đã dành rất nhiều tâm huyết, công sức để hoàn thành đồ án, song ứng dụng của chúng em sẽ không thể tránh khỏi những thiếu sót. Chúng em rất mong nhận được phản hồi tích cực và góp ý chân thành từ phía thầy cô.

**Lê Hoàng Long**

**Nguyễn Đình Hưng**

**1. MỤC ĐÍCH THỰC HIỆN ĐỀ TÀI**

- Nghiên cứu sáng tạo và vận dụng các kiến thức đã học, để xây dựng và phát triển ứng dụng giải quyết các vấn đề trong thực tế đời sống (ứng dụng đặt món ăn và thanh toán đơn hàng giúp hỗ trợ nhân viên kinh doanh thuận lợi, dễ dàng hơn)

- Có nền tảng vững chắc và được nâng cao các kỹ năng phân tích bài toán, giải quyết các vấn đề trong xây dựng ứng dụng thực tế

**2. MỤC TIÊU ĐỀ TÀI**

- Vận dụng các kiến thức đã học để viết ứng dụng đặt món hàng và thanh toán đơn hàng dựa trên ngôn ngữ lập trình C/C++:

- Sử dụng các mảng struct, mảng song song để lưu dữ liệu quan hệ.

- Sử dụng các mảng một chiều để lưu các món ăn: tên món, đơn giá.

- Sử dụng các kĩ thuật thao tác với file

**3. PHẠM VI VÀ ĐỐI TƯỢNG NGHIÊN CỨU**

- Tìm hiểu cấu trúc dữ liệu( mảng, struct), làm việc với file và các thuật toán cần thiết để viết 1 chương trình.

- Liên kết các kiến thức đã học,nghiên cứu để xây dựng bài toán hoàn chỉnh nhất.

**4. PHƯƠNG PHÁP NGHIÊN CỨU**

- Làm việc theo nhóm.

- Phân tích vấn đề và tổng hợp.

- Phương pháp nghiên cứu tài liệu về lập trình C/C++.

**5. CẤU TRÚC ĐỒ ÁN CỦA MÔN HỌC:**

**PHẦN 1**: Tổng quan đề tài.

**PHẦN 2**: Cơ sở lý thuyết.

**PHẦN 3**: Tổ chức dữ liệu và thuật toán.

**PHẦN 4**:Chương trình và kết quả.

**PHẦN 5**: Kết luận

1. TỔNG QUAN ĐỀ TÀI

Trong những năm gần đây, sự phát triển của Công nghệ thông tin đã mang lại những thay đổi đáng kể trong cuộc sống con người. Các lĩnh vực khoa học, quân sự, nghệ thuật, kinh tế v.v.. đang áp dụng từng ngày và càng trở nên phổ biến hơn. Tuy vậy, trong các dịch vụ đời sống thường ngày như nhà ăn hay quán cà phê, một số vẫn còn sử dụng phương pháp đặt món một cách trực tiếp và thanh toán thủ công. Điều này có thể dẫn đến tính toán sai cũng như mất nhiều thời gian, công sức. Tính toán sai sẽ khiến việc thống kê doanh thu cuối ngày của nhà ăn, quán cà phê thua lỗ, hiệu quả kinh doanh thấp. Mất nhiều thời gian sẽ khiến khách hàng phàn nàn, khó chịu vì phải đợi.

Vì lí do trên, dựa vào những nghiên cứu từ các môn học đại cương (Cấu trúc dữ liệu, Phương pháp tính, Kĩ thuật lập trình…), nhóm em đã xây dựng nên một ứng dụng hỗ trợ đặt món ăn và thanh toán các đơn hàng. Ứng dụng này có các ưu điểm như dễ sử dụng, giải quyết được nhiều vấn đề thực tế trong cuộc. Nhóm em hi vọng đồ án thực tế này có thể giúp tiết kiệm nhiều thời gian, công sức, tiền bạc, hiệu quả kinh doanh tăng cao hơn.

1. CƠ SỞ LÝ THUYẾT
   1. Ý tưởng

- Bắt đầu một ngày, thu ngân nhập vào một mã giao dịch

- Chương trình in ra Menu các món ăn của nhà hàng. Cho phép khách hàng nhập vào các món ăn cần đặt với số lượng tương ứng. Chương trình sẽ tính tổng, giảm giá nếu có và in ra hóa đơn.

- Khách hàng được phép nhập tối đa 5 món và nhập vào 0 0 khi không chọn thêm món. Vì vậy, khi khách hàng đã chọn đủ 5 món thì tiến trình chọn món cũng kết thúc ngay lập tức và in ra hóa đơn mà không cần nhập vào 0 0

- Khách hàng được giảm giá 25% nếu Tổng đơn hàng >= 2 triệu

- Kết thúc một ngày, thu ngân nhập vào mã giao dịch trên. Chương trình thống kê in ra toàn bộ các đơn hàng đã có trong ngày và ghi ra file.

* 1. Cơ sở lý thuyết

- Sử dụng mảng song song để lưu dữ liệu quan hệ.

- Sử dụng các mảng một chiều để lưu các món ăn: tên món, đơn giá v.v..

- Sử dụng các thuật toán để thao tác với mảng.

- Sử dụng các kĩ thuật thao tác với file.

- Sử dụng kiểu cấu trúc bản ghi (struct) lưu các trường dữ liệu: tên món ăn, mã món, số lượng, đơn giá món, thành tiền.

typedef struct InfoFoodCustomer {

// tên món ăn

char \*item;

// mã món

int choice;

// số lượng

int quantity;

// đơn giá

int price;

// thành tiền

int amount;

} Food;

- Sử dụng kiểu cấu trúc bản ghi lồng nhau (struct lồng struct) có các trường sau:

typedef struct Customer {

// thời gian gọi món

char \*time;

// đếm số lần order món ăn

int countFood;

// con trỏ kiểu struct để lưu thông tin mỗi lần order món ăn

Food \*listInforFood;

// lưu vị trí bàn

int table;

// tổng tiền

int total;

// thanh toán

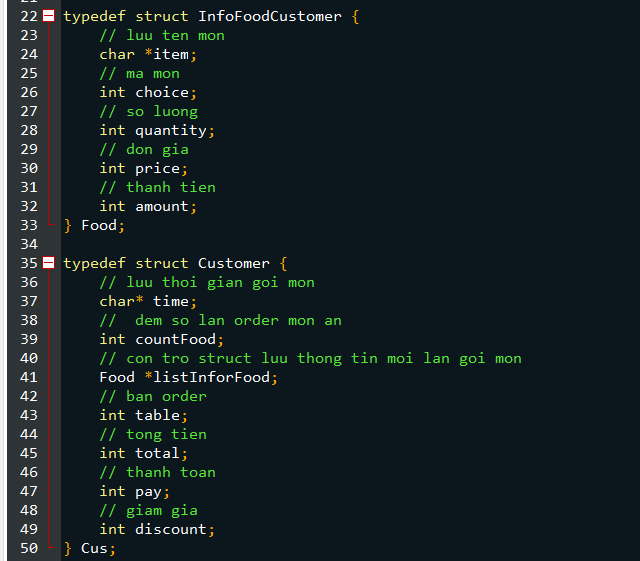
int pay;

// giảm giá

int discount;

} Cus;

Struct lồng struct được viết trong chương trình:



Hình 1. Struct lồng struct trong chương trình

1. TỔ CHỨC CẤU TRÚC DỮ LIỆU VÀ THUẬT TOÁN
   1. Phát biểu bài toán

- Để bắt đầu một ngày làm việc, nhân viên (staff) của quán phải nhập tài khoản (username) và mật khẩu (mã ID của quán). Thông tin tài khoản và mã ID được lưu trong file userName\_and\_ID.txt

- Chương trình bắt đầu với yêu cầu nhân viên nhập tên tài khoản (username). Nếu nhập sai, chương trình sẽ thông báo: “INCORRECT USERNAME!”. Nhân viên cần phải nhập đúng username mới có thể đến bước nhập ID. Tương tự, bước nhập ID cũng sẽ hiển thị thông báo: “INCORRECT ID!” nếu nhân viên nhập sai mã ID.

- Khi hoàn tất hai bước nhập trên, chương trình sẽ in ra Menu, hiển thị các tính năng cho nhân viên lựa chọn để bắt đầu làm việc. Có hai tính năng trong chương trình được thể hiện bởi chữ số tương ứng (1: ORDER ; 0: EXIT). Khi khách hàng order, nhân viên nhập “1” để bắt đầu nhập thông tin. Khi kết thúc phiên làm việc trong ngày, nhập “0” để lưu tất cả dữ liệu của khách hàng ra file và kết thúc chương trình. Nếu nhập sai dữ liệu, chương trình thông báo “WRONG INPUT! PLEASE RE\_ENTERED!” và yêu cầu nhân viên nhập lại.

- Quá trình nhập order của khách hàng diễn ra qua bốn giai đoạn chính:

* Giai đoạn 1: Nhập mã món và số lượng

+ Nhân viên nhập mã món ăn trước và số lượng món ăn sau. Thông tin được cách nhau bởi phím space. Ví dụ: khách hàng gọi món 1, số lượng 2 thì nhân viên nhập là “1 2”. Nếu nhập số 1 và 2 sát nhau “12” thì chương trình hiểu là đang gọi món 12 và chưa nhập số lượng món, dẫn đến nhập thông tin sai.

+ Trường hợp nhân viên nhập mã món không tồn tại trong menu thì chương trình hiển thị thông báo “WRONG INPUT! PLEASE RE\_ENTERED” và yêu cầu nhập lại. Nếu nhập mã món có tồn tại nhưng số lượng món là một số nguyên bé hơn 1 thì cũng hiển thị thông báo “WRONG INPUT! PLEASE RE\_ENTERED” và yêu cầu nhập lại.

+ Khách hàng có thể gọi bao nhiêu món tùy thích, cho đến khi nhân viên nhập vào “0 0” thì tiến trình chọn món kết thúc. Sau đó nhập số bàn gọi món và chuyển sang giai đoạn 2

* Giai đoạn 2: Gọi thêm món ăn.

+ Sau khi nhập xong, chương trình thông báo “Do you want to order more?”

+ Trường hợp khách hàng muốn order thêm món sau khi tiến trình chọn món ở giai đoạn 1 đã hoàn tất, nhân viên có thể nhập thêm thông tin. Các thông tin này được tính vào trong cùng 1 bill

+ Nếu nhân viên nhập “y” hoặc “Y”, chương trình sẽ hiển thị các món ăn đã order trước đó và yêu cầu nhân viên nhập thêm món ăn mới. Nếu nhập “n” hoặc “N”, bước order thêm món ăn kết thúc và chuyển sang giai đoạn 3. Nếu nhập dữ liệu sai, chương trình thông báo “WRONG INPUT! PLEASE RE\_ENTERED” và yêu cầu nhập lại

* Giai đoạn 3: Chỉnh sửa thông tin món ăn

+ Tiến trình order thêm món kết thúc, chương trình sẽ hiển thị thông báo “Change your order?”. Nếu nhập “y” hoặc “Y”, nhân viên có thể chỉnh sửa thông tin theo ý của khách hàng. Nếu nhập “n” hoặc “N”, bước chỉnh sửa kết thúc và sang giai đoạn 4. Nếu nhập sai, chương trình thông báo “WRONG INPUT! PLEASE RE\_ENTERED” và yêu cầu nhập lại.

* Giai đoạn 4: Thanh toán

+ Chương trình hiển thị tổng tiền và giảm giá (nếu có), và yêu cầu nhân viên nhập số tiền thanh toán của khách hàng.

+ Sau khi thanh toán xong, khách hàng có thể lựa chọn việc xem bill của mình. Nếu có, nhân viên nhập “y” hoặc “Y” và chương trình sẽ in ra bill. Nếu không thì nhập “n” hoặc “N”. Nếu nhập sai, chương trình hiển thị thông báo “WRONG INPUT! PLEASE RE\_ENTERED” và yêu cầu nhập lại. Sau giai đoạn 4, dữ liệu của khách hàng sẽ được lưu vào file đảm bảo không bị thiếu sót, nhầm lẫn

- Để kết thúc phiên làm việc, nhân viên nhập “0” và chương trình yêu cầu nhập lại ID. Nếu nhập đúng, chương trình sẽ hiển thị bảng thống kê doanh thu trong ngày và lưu dữ liệu vào file. Nếu nhập sai ID, chương trình thông báo “INCORRECT ID!” và yêu cầu nhập lại.

* 1. Cấu trúc dữ liệu

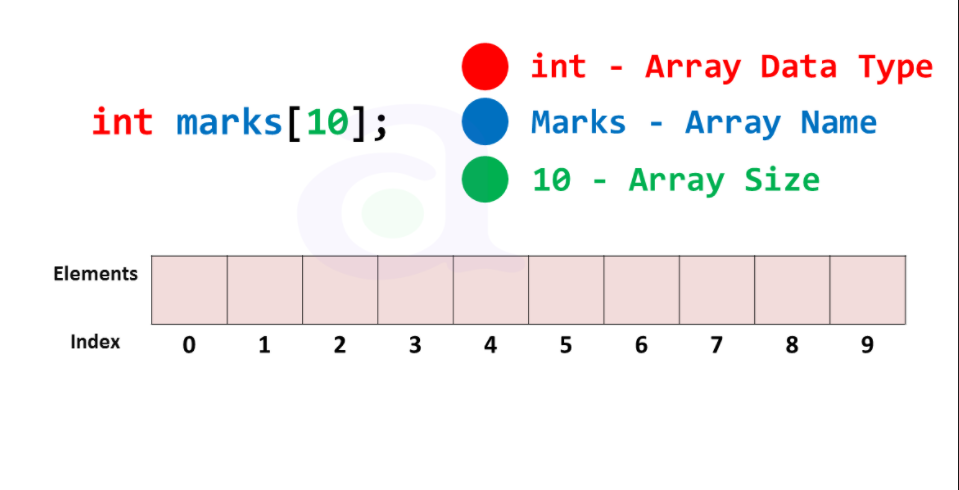
**3.2.1. Danh sách liên kết đặc**

- Cấu trúc dữ liệu là hình thức tổ chức một nhóm dữ liệu. Với cấu trúc dữ liệu, ngoài chức năng lưu trữ dữ liệu của nhiều phần tử như mảng thì nó còn có các hàm để thao tác với dữ liệu. Các cấu trúc dữ liệu thông dụng có thể kể đến như danh sách liên kết, hàng đợi (stack), hàng đợi ưu tiên (queue), cây nhị phân (binary search tree) v.v…

- Danh sách liên kết có thể được cài đặt bằng mảng hoặc bằng con trỏ.

**-** Danh sách liên kết đặc là một cấu trúc dữ liệu mà các phần tử được lưu trữ kế tiếp nhau trong bộ nhớ dưới hình thức một mảng.

- Mảng là một kiểu dữ liệu có cấu trúc gồm nhiều thành phần có cùng kiểu dữ liệu, mỗi thành phần được gọi là một phần tử. Các phần tử được lưu trữ trên các ô nhớ liền kề nhau. Mảng cho phép truy cập phần tử một cách ngẫu nhiên dựa vào chỉ mục (index). Một mảng có kích thước N phần tử thì phần tử đầu tiên có index bằng 0 và phần tử cuối cùng có index bằng N – 1. Cú pháp truy cập phần tử: ten\_mang[index]



Hình 2. Danh sách đặc được triển khai như một mảng tĩnh

- Ưu điểm và nhược điểm:

|  |  |
| --- | --- |
| **Ưu điểm** | **Nhược điểm** |
| Dễ viết chương trình | Phải xác định trước kích thước của danh sách |
| Tiết kiệm bộ nhớ hơn cho mỗi phần tử vì không chứa trường lưu địa chỉ của phần tử khác (các phần tử nằm trên các ô nhớ liên tiếp nhau) | Kích thước của danh sách là cố định trong suốt quá trình thực thi chương trình |
| Khả năng truy xuất nhanh | Nếu vượt quá kích thước tối đa, chương trình sẽ báo lỗi |

**3.2.2. Kĩ thuật thao tác File**

**-**  Mọi dữ liệu trong chương trình sẽ mất nếu như chương trình kết thúc. Ngôn ngữ lập trình C có các thư viện hỗ trợ việc đọc ghi file giúp thao tác dữ liệu của chương trình theo ý muốn.

- File có hai dạng là file văn bản (text files) và file nhị phân (binary files). Hầu hết ta làm việc với file văn bản vì nó gần gũi hơn trong đời sống (các file đều được viết ở dạng file văn bản)

- Thao tác file có các bước quan trọng sau:

* **Khai báo con trỏ file kiểu FILE:**

+ Khi làm việc với file, con trỏ kiểu FILE chính là cầu nối giữa chương trình và tập tin cần thao tác. Không có con trỏ này, ta không thể làm việc được với file.

+ Cú pháp khai báo con trỏ file: FILE \*ten\_con\_tro\_file;

* **Thao tác mở file:**

+ Thư viện stdio.h hỗ trợ hàm fopen( ) để mở file. Bước mở file là bước cơ bản để chúng ta có thể đọc dữ liệu từ file vào chương trình hoặc ghi dữ liệu từ chương trình ra file.

+ Cú pháp khai báo hàm fopen( ) trong thư viện stdio.h: FILE \*fopen(const char \*name\_file, const char \*mode)

+ Ví dụ:

FILE \*fptr;

fptr = fopen (“Menu.txt”, “r”);

+ Mode là tham số cần được chỉ định, thể hiện cách chúng ta thao tác với file (đọc dữ liệu, ghi dữ liệu dưới dạng văn bản, ghi dữ liệu dưới dạng nhị phân, ghi dữ liệu theo chế độ append…)

+ Khi làm việc với file văn bản, ta cần phải nhớ các mode sau:

|  |  |  |
| --- | --- | --- |
| **MODE** | **Ý NGHĨA** | **GHI CHÚ** |
| r | Mở file chỉ cho phép đọc | Nếu file không tồn tại, fopen( ) trả về NULL |
| w | Mở file chỉ cho phép ghi | Nếu file đã tồn tại, nội dung sẽ bị ghi đè. Nếu file không tồn tại, nó sẽ được tạo tự động. |
| a | Mở file chỉ cho phép ghi theo chế độ “append” | Nếu file đã tồn tại, nội dung sẽ được ghi vào cuối nội dung đã có. Nếu file không tồn tại, nó sẽ được tạo tự động. |
| r+ | Mở file cho phép cả đọc và ghi | Nếu file không tồn tại, fopen( ) trả về NULL |
| w+ | Mở file cho phép cả đọc và ghi | Nếu file đã tồn tại, nội dung sẽ bị ghi đè. Nếu file không tồn tại, nó sẽ được tạo tự động. |
| a+ | Mở file cho phép cả đọc và ghi “append” | Nếu file đã tồn tại, nội dung sẽ được ghi vào cuối nội dung đã có. Nếu file không tồn tại, nó sẽ được tạo tự động. |

+ Các mode dùng để làm việc với file nhị phân chỉ cần thêm chữ cái b (rb, wb, ab, rb+, wb+, ab+). Về mặt ý nghĩa và ghi chú thì hoàn toàn tương tự.

* **Đọc dữ liệu từ file vào chương trình:**

**+** Thư viện stdio.h hỗ trợ 2 hàm đọc dữ liệu từ file vào chương trình là fscanf( ) và fread( ). Hàm fscanf( ) đọc dữ liệu file văn bản, fread( ) đọc dữ liệu file nhị phân.

**+** Hầu hết ta làm việc với file văn bản nên ta sử dụng fscanf( ) nhiều nhất.

**+** fscanf( ) đọc từng từ trong tập tin cho đến eof (end of file).

**+** Khai báo fscanf( ) trong stdio.h: int fscanf(FILE \*ten\_con\_tro\_file, const char \*format);

* **Ghi dữ liệu từ chương trình ra file:**

+ Thư viện stdio.h hỗ trợ 2 hàm ghi dữ liệu từ chương trình ra file là fprintf( ) và fwrite( ). Hàm fprintf( ) ghi dữ liệu lên file văn bản, fwrite( ) ghi dữ liệu lên file nhị phân.

+ Tương tự, ta chủ yếu thao tác với file văn bản nên ta sử dụng fprintf( ) nhiều nhất

+ Khai báo fprintf( ) trong stdio.h: int fprintf(FILE \*ten\_con\_tro\_file, const char \*format);

* **Thao tác đóng file:**

**+** Thư viện stdio.h hỗ trợ hàm fclose( ) giúp chúng ta đóng file.

**+** Sau khi làm việc với tập tin xong, luôn có bước đóng file.

**+** Khai báo fclose( ) trong stdio.h: int fclose(FILE \*ten\_con\_tro\_file);

**+** Ví dụ: fclose(fptr);

* 1. Thuật toán

**3.3.1. Thuật toán tìm kiếm tuyến tính (linear searching)**

**-** Tìm kiếm tuyến tính (linear searching) còn được gọi là tìm kiếm tuần tự (sequential searching). Đây là một kỹ thuật tìm kiếm phổ biến và dễ áp dụng. Về mặt ý tưởng, thuật toán tiến hành so sánh phần tử cho trước với các phần tử trong mảng hoặc một cấu trúc dữ liệu. Hoạt động tìm kiếm được diễn ra qua tất cả các phần tử

- Mỗi phần tử đều được kiểm tra và nếu trùng với phần tử cho trước thì phần tử cụ thể đó được trả về, nếu không thì quá trình tìm kiếm tiếp tục diễn ra cho tới khi tìm kiếm hết dữ liệu.

- Giải thuật tìm kiếm tuyến tính chỉ hiệu quả khi cần tìm kiếm trên một mảng, một cấu trúc dữ liệu đủ nhỏ hoặc chưa được sắp xếp thứ tự.

- Ví dụ về tìm kiếm tuyến tính:

void linearSearch(int arr[], int value) {

int flag = 1;

for (int i = 0; i < 9; i++) {

if (arr[i] == value) {

printf("value = %d trong mang co vi tri index la %d\n", i);

flag = 0;

break;

}

}

if (flag) {

printf("Khong tim thay value = %d trong mang!\n", value);

}

}

int main() {

int value1 = 4;

int value2 = 3;

int arr[] = {1, 6, -3, 5, -2, 0, 4, 9, -1};

linearSearch(arr, value1); // value = 4 trong mang co vi tri index la 6

linearSearch(arr, value2); //Khong tim thay value = 3 trong mang

return 0;

}

**Cài đặt: Thuật toán tìm kiếm tuyến tính trong chương trình được sử dụng ở hàm priceOfFood**

****

Hình 3. Thuật toán tìm kiếm tuyến tính trong hàm priceOfFood

Độ phức tạp thuật toán: O(n)

**3.3.2. Thuật toán nhập mật khẩu tạo kí tự \***

- Mật khẩu là dữ liệu luôn cần phải được nâng cao bảo mật, tránh trường hợp bị lộ mật khẩu ra ngoài. Do đó, nhóm em đã viết thêm thuật toán chuyển hóa 1 kí tự nhập từ bàn phím thành kí tự \*. Như vậy, nếu nhập là 123456789, mật khẩu sẽ hiển thị là \*\*\*\*\*\*\*\*\*

- Ý tưởng của thuật toán:

+ Mỗi lần nhập 1 kí tự bất kì, chương trình sẽ đọc kí tự đó, sau đó tự động chuẩn hóa kí tự đó sang kí tự \*. Để làm được điều này, ta không thể dùng hàm fgets( ) hay hàm gets( ) vì với hai hàm này, chương trình sẽ cho chúng ta nhập 1 chuỗi trước, và chương trình chỉ đọc chuỗi này khi người dùng nhấn phím Enter (kí tự ‘\n’). Nếu áp dụng fgets( ) hay gets( ) thì sau mỗi lần nhập một kí tự, ta nhấn Enter và tiếp tục nhập kí tự tiếp theo thì mới thỏa mãn ý tưởng của thuật toán đã đưa ra. Điều này đồng nghĩa nhập bao nhiêu kí tự sẽ có bấy nhiêu lượt nhấn phím Enter, hoàn toàn không thực tế và mất tự nhiên. Tương tự, ta cũng không thể dùng hàm scanf( ) với lí do tương tự. Như vậy, ta sẽ sử dụng hàm getch( ) trong thư viện conio.h

+ Hàm getch( ) dùng để lấy kí tự từ bàn phím hoặc một console nào đó, và kí tự này được lưu giữ trong vùng đệm nhưng không echoing với màn hình. Lợi dụng tính chất này, mỗi lần nhập một kí tự, chương trình sẽ lấy kí tự đó và ta có thể thao tác để chuyển thành kí tự \*.

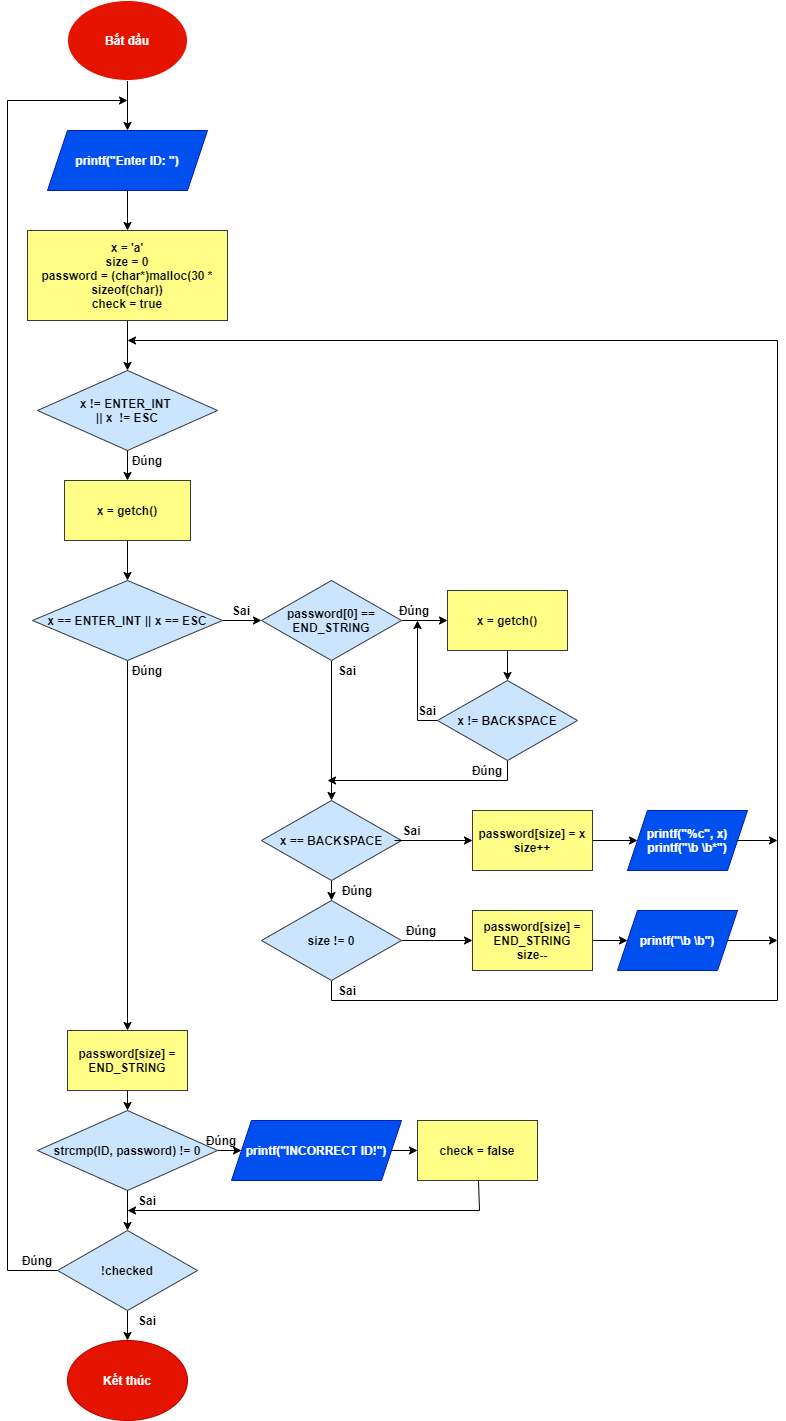
+ Đầu tiên, ta sẽ cho chương trình “dừng lại” và đợi người dùng nhập các kí tự từ bàn phím. Việc nhập sẽ kết thúc khi người dùng nhập phím Enter. Khi đó chương trình sẽ lần lượt lấy từng kí tự trong chuỗi kí tự vừa nhập vào. Vì độ dài của chuỗi kí tự hoàn toàn phụ thuộc vào khâu nhập của người dùng nên ta không thể dùng vòng lặp for để duyệt. Vòng lặp while là cách tốt nhất để thực hiện điều này. Việc lặp sẽ tiếp tục cho đến khi chương trình gặp được kí tự ‘\n’ (Nhập từ phím Enter) thì dừng lại và thoát khỏi vòng lặp.

+ Trong quá trình lặp, ta phải kiểm tra một số điều kiện quan trọng như kí tự nhập vào có dạng Backspace hay không (trong ASCII, Backspace tương ứng với số 8)…

+ Sau khi lấy được kí tự, ta hiển thị kí tự đó lên màn hình trước, sau đó chuyển hóa kí tự đó thành kí tự \*.

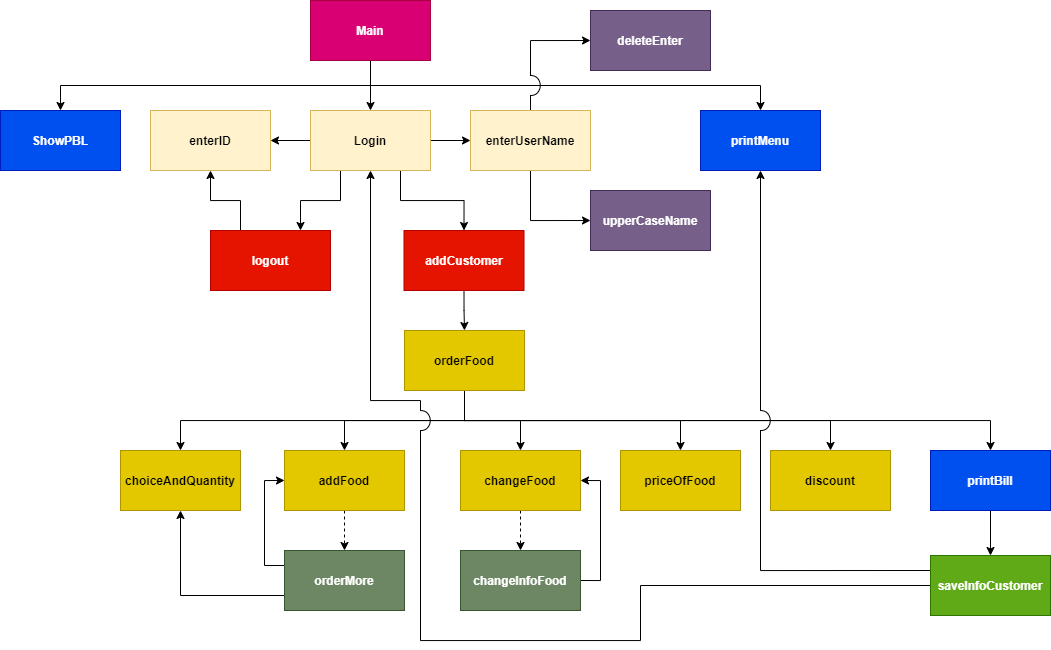
+ Nếu kí tự nhập vào có dạng Backspace thì ta không hiển thị kí tự \* mà sẽ làm mất kí tự \*

- Thuật toán nhập mật khẩu tạo kí tự \* được thể hiện theo sơ đồ flowchart như sau:



Hình 4. Sơ đồ làm việc của thuật toán nhập mật khẩu tạo kí tự \*

1. CHƯƠNG TRÌNH VÀ KẾT QUẢ
   1. Tổ chức chương trình



Hình 5. Sơ đồ tổ chức của chương trình

**4.1.1. Các hằng số được định nghĩa và thư viện được sử dụng**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<stdbool.h>

#include<time.h>

#include<conio.h>

#define ENTER\_CHAR '\n'

#define YES\_LOWER 'y'

#define YES\_UPPER 'Y'

#define NO\_LOWER 'n'

#define NO\_UPPER 'N'

#define A\_LOWER 'a'

#define Z\_LOWER 'z'

#define SPACE ' '

#define END\_STRING '\0'

#define N\_MAX 100

#define ENTER\_INT 13

#define ESC 27

#define BACKSPACE 8

**4.1.2. Các hàm được xây dựng trong chương trình**

Hàm trả về thời gian khách hàng order món:

char \*displayTime(char \*saveTime)

Hàm xóa kí tự xuống dòng khi dùng hàm fgets( ) để nhập tên nhân viên:

void deleteEnter(char \*name)

Hàm in hoa các chữ cái đầu của họ và tên đệm:

void upperCaseName(char \*name)

Hàm nhập tài khoản (user name) của nhân viên:

char \*enterUserName( )

Hàm nhập mã ID của quán:

void enterID( )

Hàm giúp đăng nhập để khởi động menu và bắt đầu phiên làm việc:

char \*login( )

Hàm giúp đăng xuất để kết thúc phiên làm việc và thoát chương trình:

void logout(Cus \*listCustomer, int listCus, long totalDay)

Hàm xử lí tính số tiền đơn giá, thành tiền, tính tổng tiền, lưu trữ tên món ăn:

void priceOfFood(Cus \*customer)

Hàm giúp nhập mã món và số lượng món:

void choiceAndQuantity(Cus \*customer)

Hàm xử lí thông tin món ăn được nhập thêm:

void orderMore(Cus \*customer)

Hàm hiển thị lựa chọn gọi thêm món:

void addFood(Cus \*customer)

Hàm xử lí thay đổi thông tin món ăn:

void changeInfoFood(Cus \*customer)

Hàm hiển thị lựa chọn thay đổi thông tin món:

void changeFood(Cus \*customer)

Hàm xử lí số tiền được giảm giá:

void discount(Cus \*customer)

Hàm giúp nhập toàn bộ thông tin món ăn của khách hàng:

void orderFood(Cus \*customer)

Hàm thêm thông tin khách hàng vào danh sách trong ngày:

Cus addCustomer( )

Hàm in hóa đơn:

void printBill(Cus customer, char \*staff)

Hàm lưu dữ liệu khách hàng vào file theo chế độ append:

void saveInfoCustomer(Cus customer, char \*staff, int listCus)

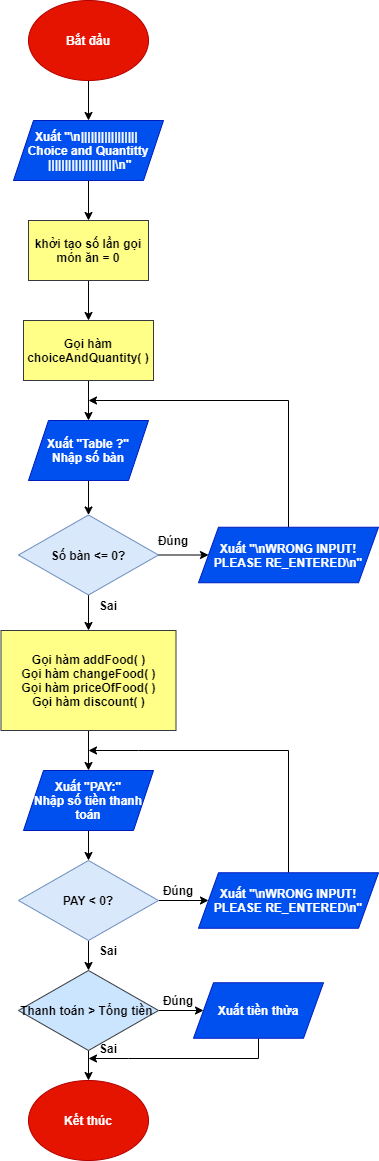
Hàm in menu quán:

void printMenu( )

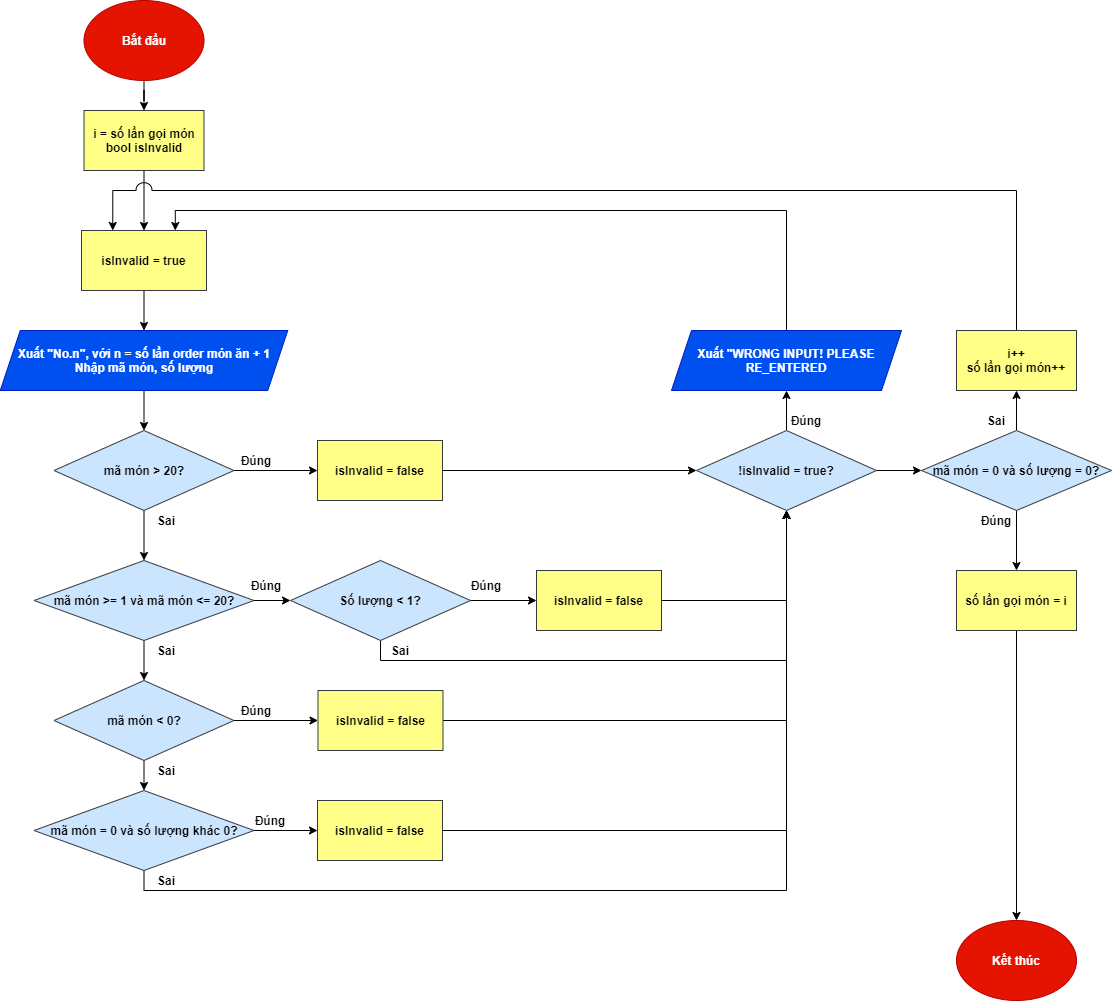
Hàm hiển thị thông tin đồ án PBL:

Void showPBL( )

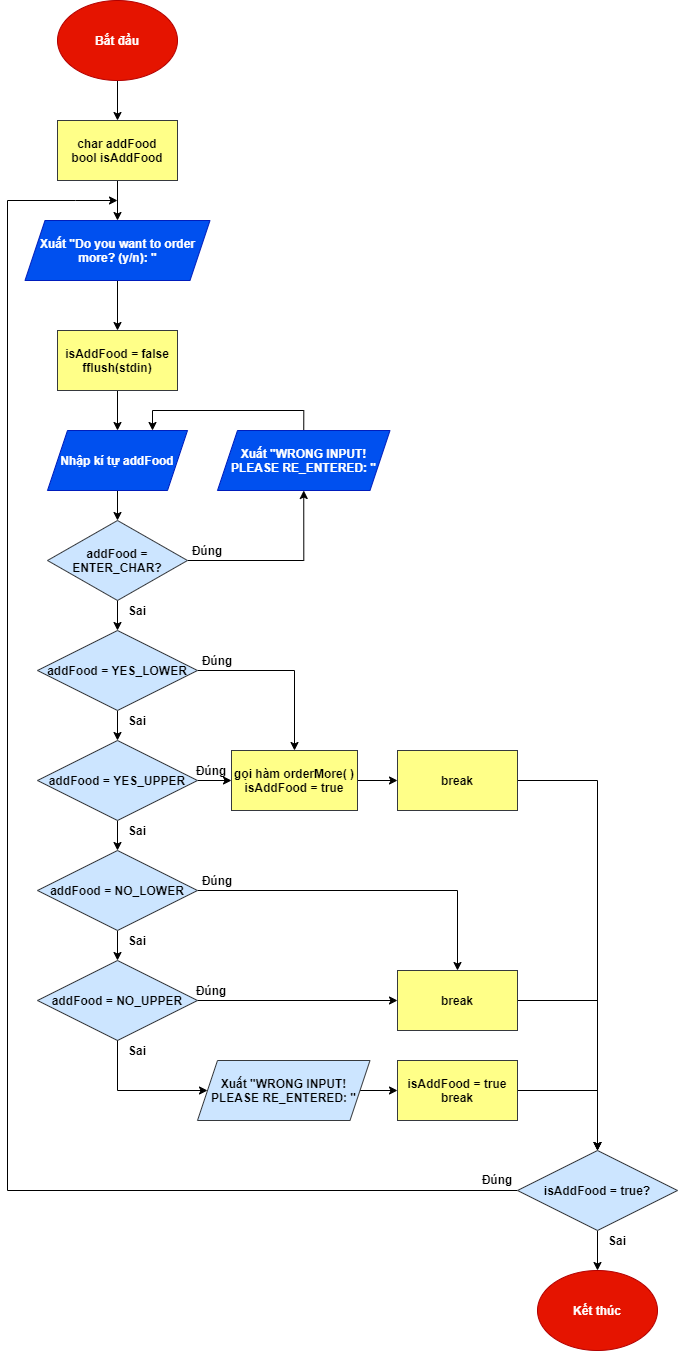
**4.1.3 Sơ đồ của một số hàm trong chương trình**



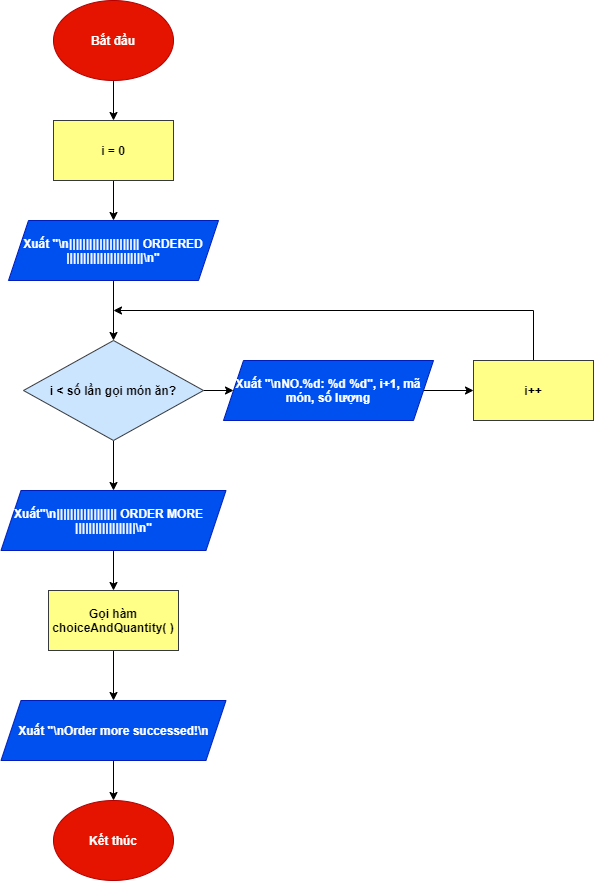
Hình 6. Sơ đồ làm việc của hàm orderFood



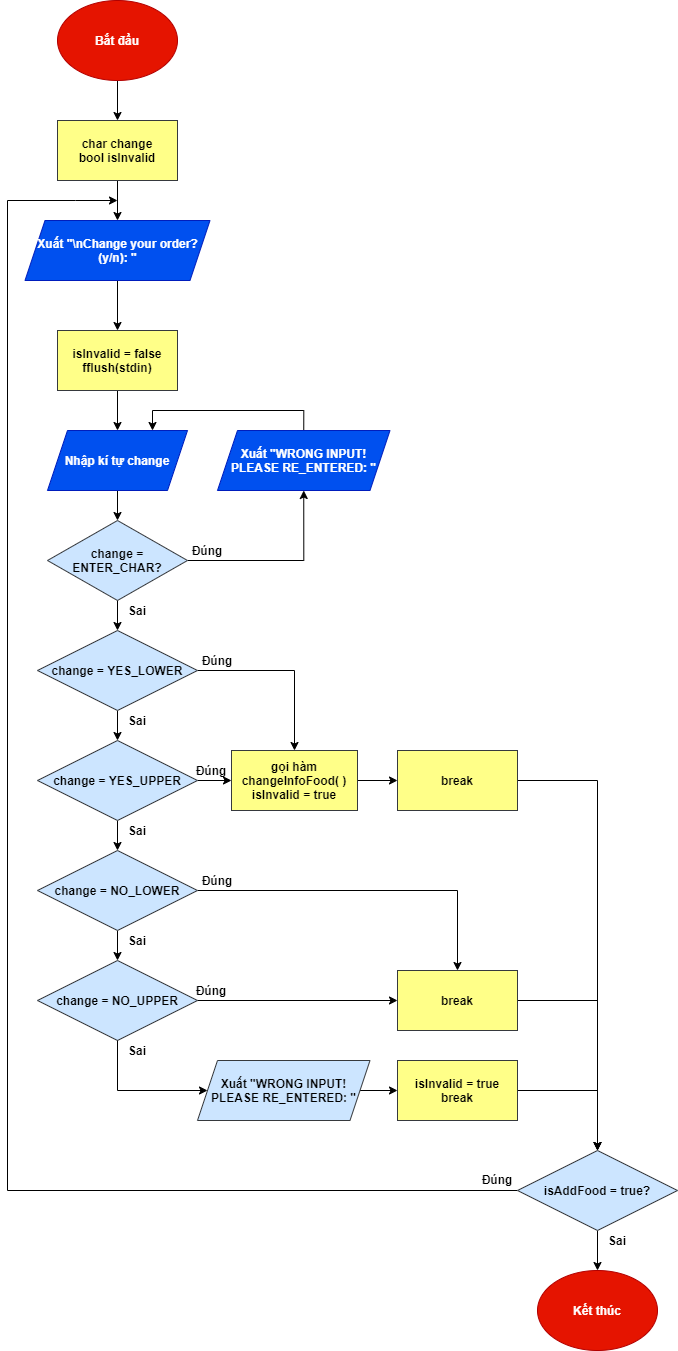
Hình 7. Sơ đồ làm việc của hàm choiceAndQuantity



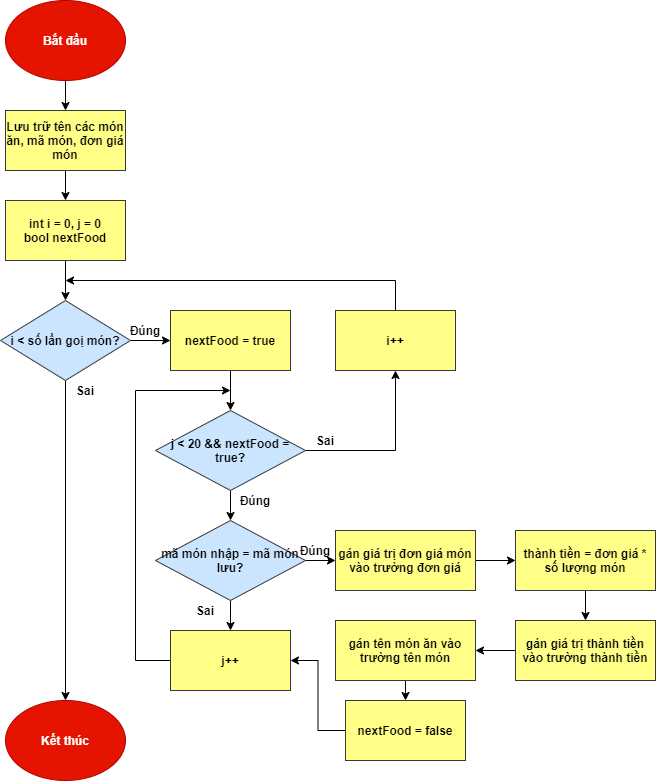
Hình 8. Sơ đồ làm việc của hàm addFood



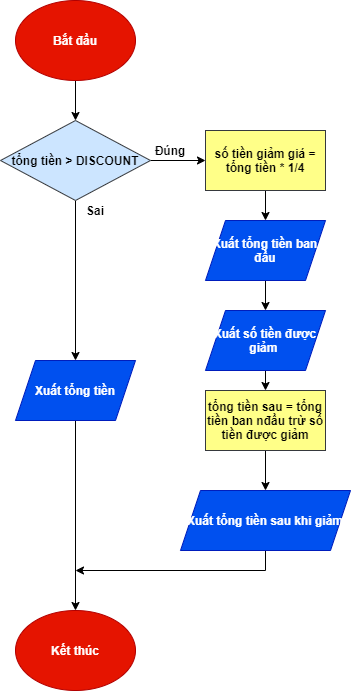
Hình 9. Sơ đồ làm việc của hàm orderMore



Hình 10. Sơ đồ làm việc của hàm changeFood



Hình 11. Sơ đồ làm việc của hàm priceOfFood

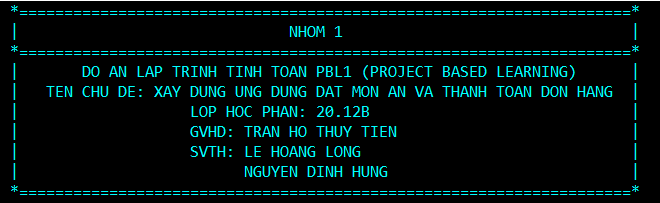


Hình 12. Sơ đồ làm việc của hàm discount

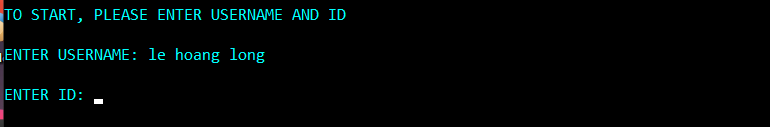
* 1. Ngôn ngữ cài đặt

Chương trình được viết và thực thi bằng ngôn ngữ lập trình C

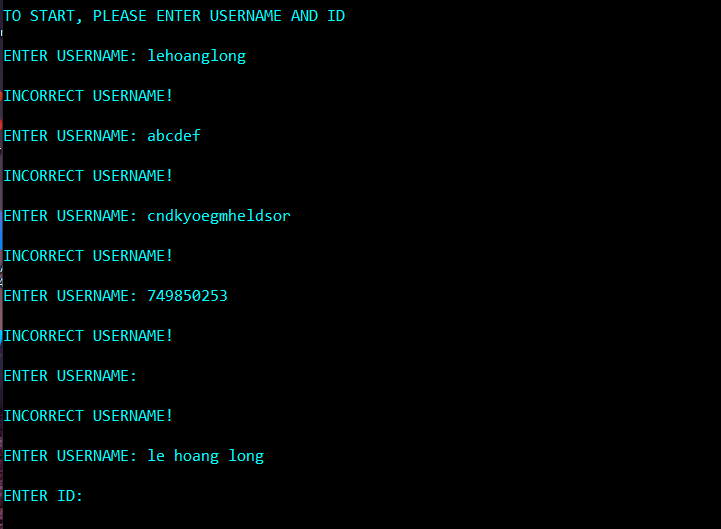
* 1. Kết quả
     1. Giao diện chính của chương trình



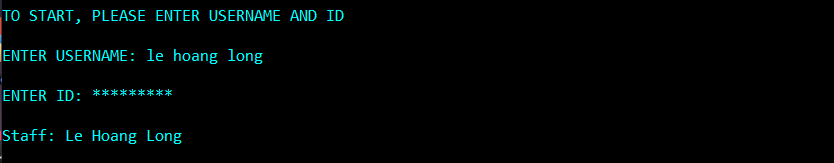
Hình 13. Giao diện mở đầu của chương trình với phần giới thiệu thông tin đồ án



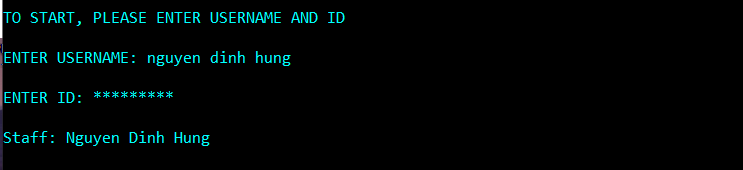
Hình 14. Giao diện nhập user name



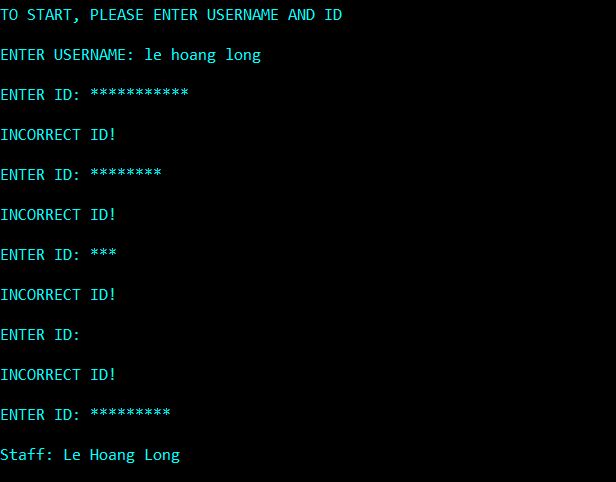
Hình 15. Giao diện khi nhập sai user name



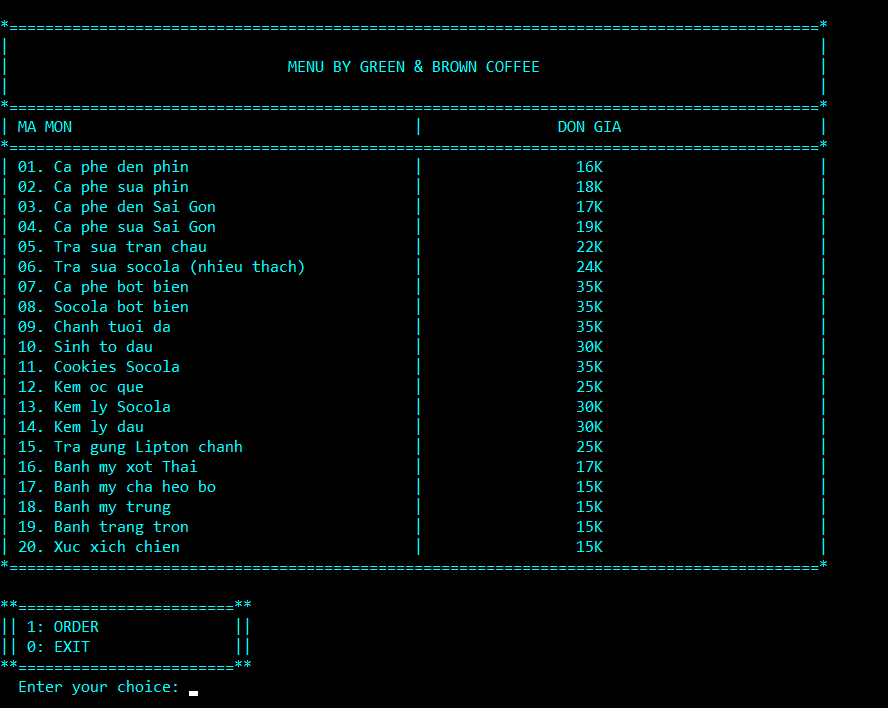
Hình 16. Giao diện khi nhập ID thành công và in nhân viên (staff) trong ngày



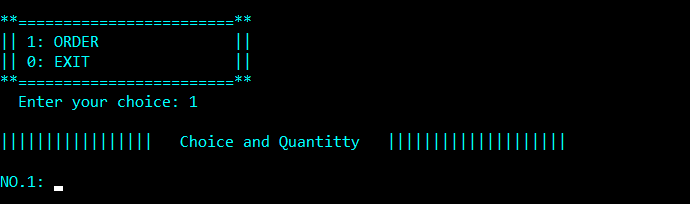
Hình 17. In ra nhân viên (staff) khác trong ngày



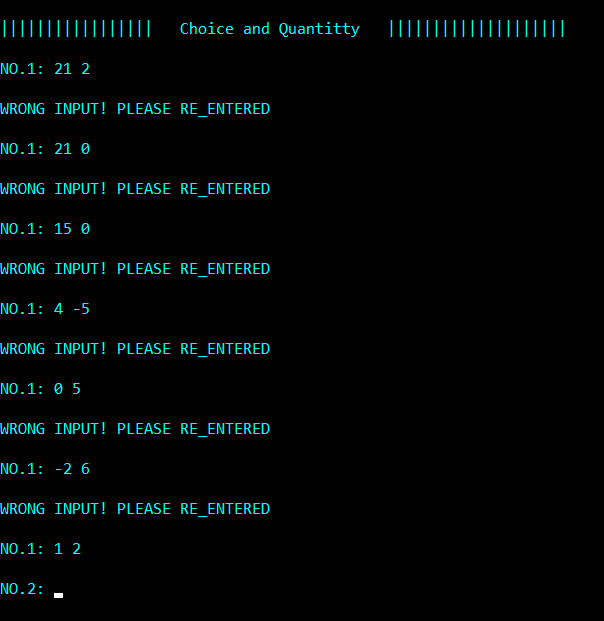
Hình 18. Giao diện khi nhập sai ID



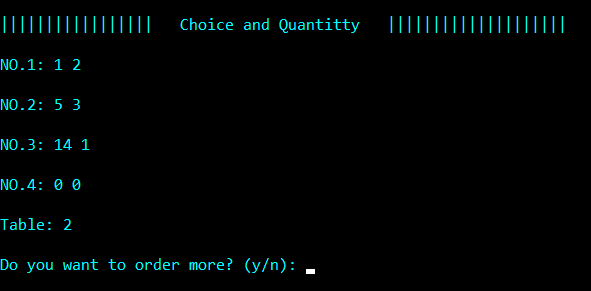
Hình 19. Giao diện khi đăng nhập thành công, chương trình in ra menu



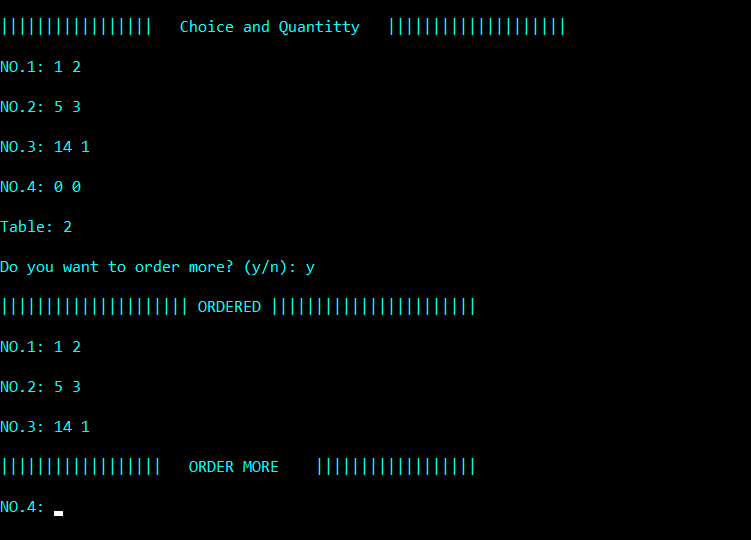
Hình 20. Giao diện khi nhập “1”, chương trình bắt đầu yêu cầu nhân viên nhập mã món và số lượng



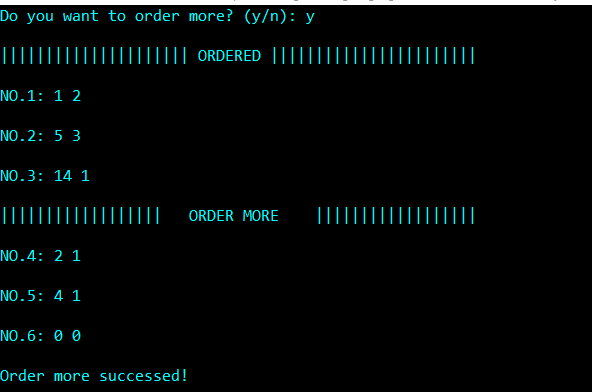
Hình 21. Giao diện khi nhập sai mã món hoặc số lượng, yêu cầu nhân viên nhập lại



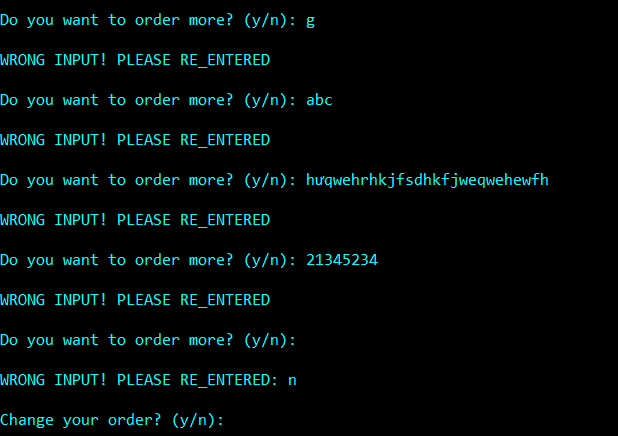
Hình 22. Giao diện sau khi tiến trình chọn món kết thúc, nhập số bàn, và nhập lựa chọn



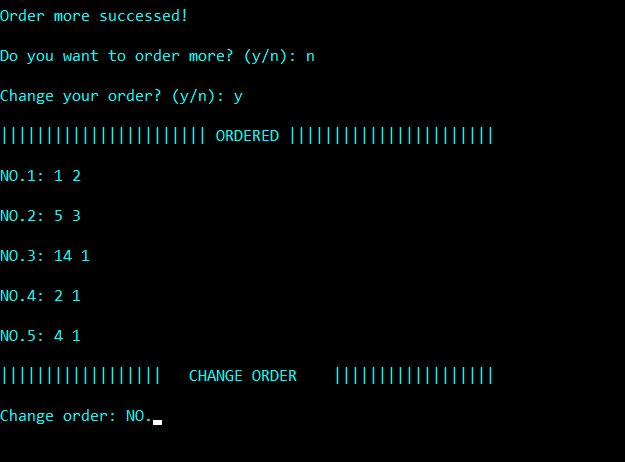
Hình 23. Giao diện khi đồng ý gọi thêm món. “ORDERED” là phần đã gọi món, “ORDER MORE” là phần nhập thêm món



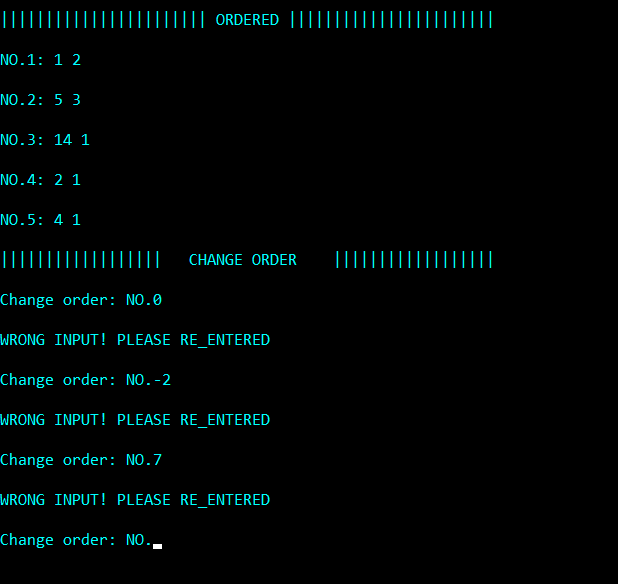
Hình 24. Giao diện gọi thêm món thành công



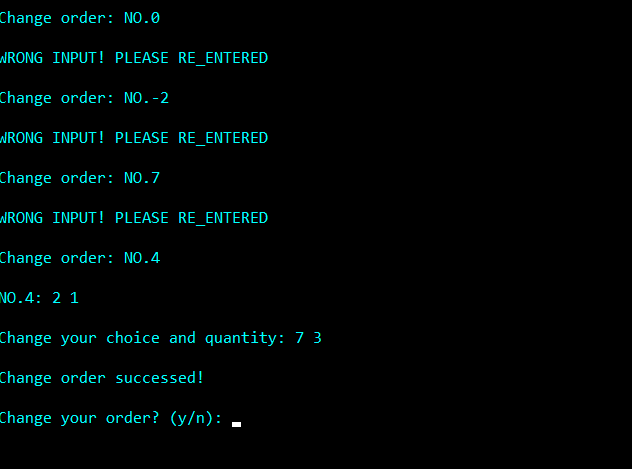
Hình 25. Giao diện khi nhập sai lựa chọn



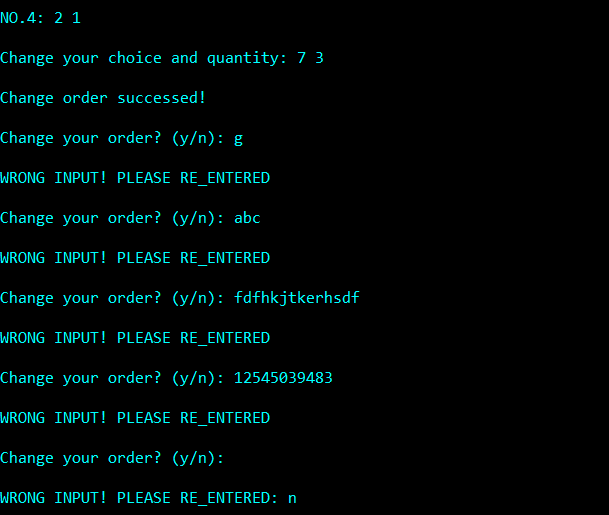
Hình 26. Giao diện khi thay đổi thông tin món ăn.



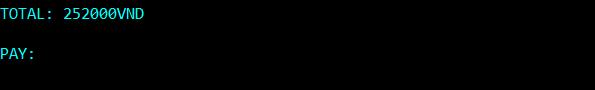
Hình 27. Giao diện khi nhập sai số thứ tự gọi món



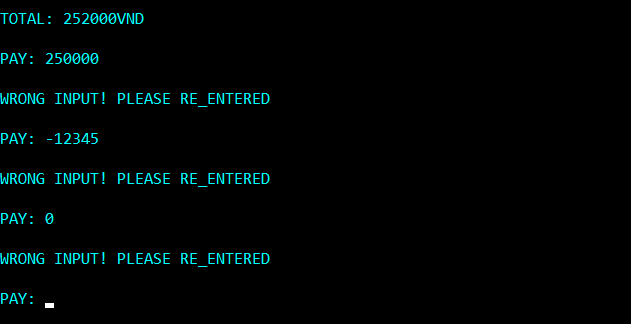
Hình 28. Giao diện khi nhập đúng. Chương trình yêu cầu thay đổi mã món và số lượng



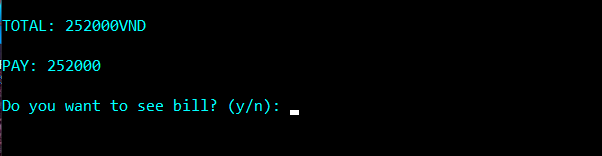
Hình 29. Giao diện khi nhập sai lựa chọn



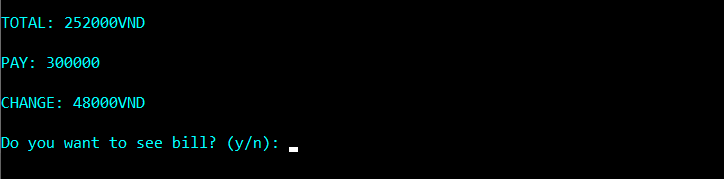
Hình 30. Giao diện in ra thông tin tổng tiền và yêu cầu nhập số tiền thanh toán



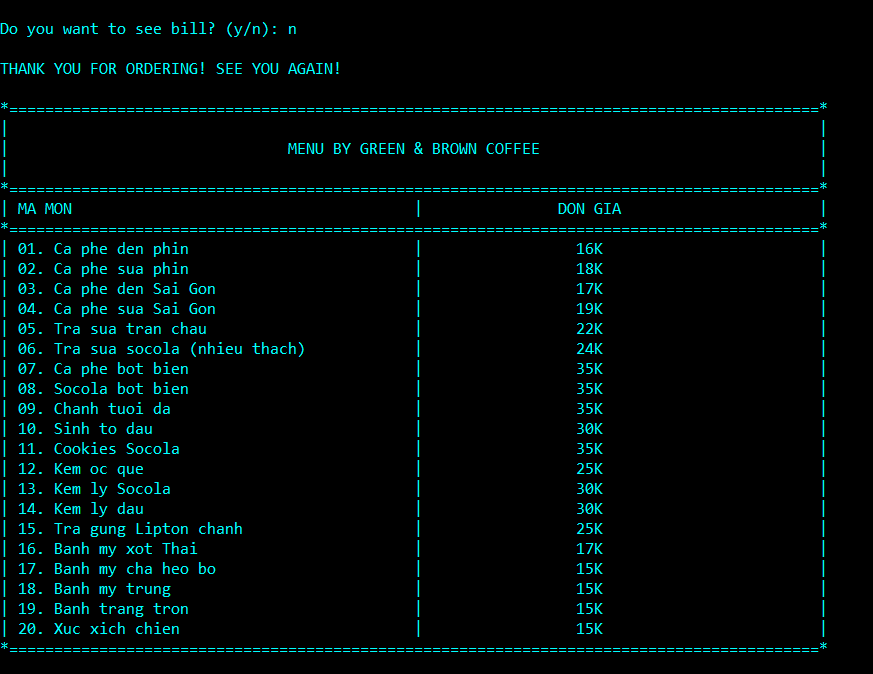
Hình 31. Giao diện khi nhập sai số tiền thanh toán



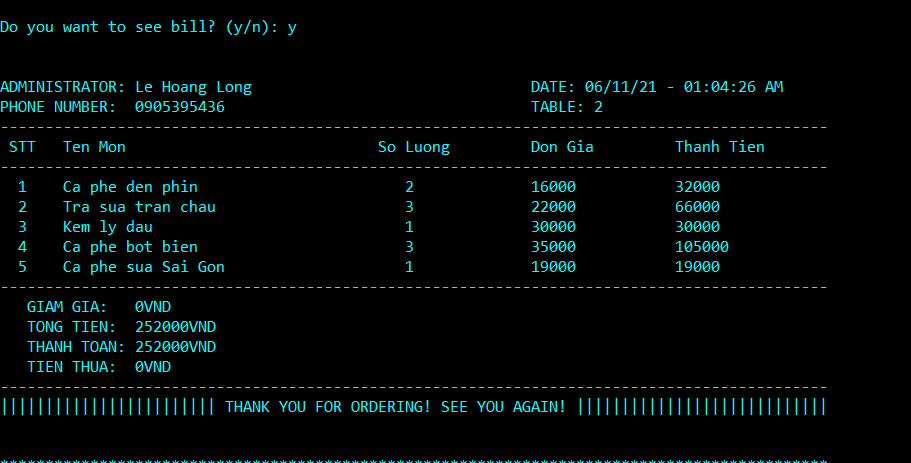
Hình 32. Giao diện khi nhập thành công số tiền thanh toán



Hình 33. Giao diện khi nhập số tiền thanh toán lớn hơn tổng tiền. Chương trình hiển thị tiền thừa (change)



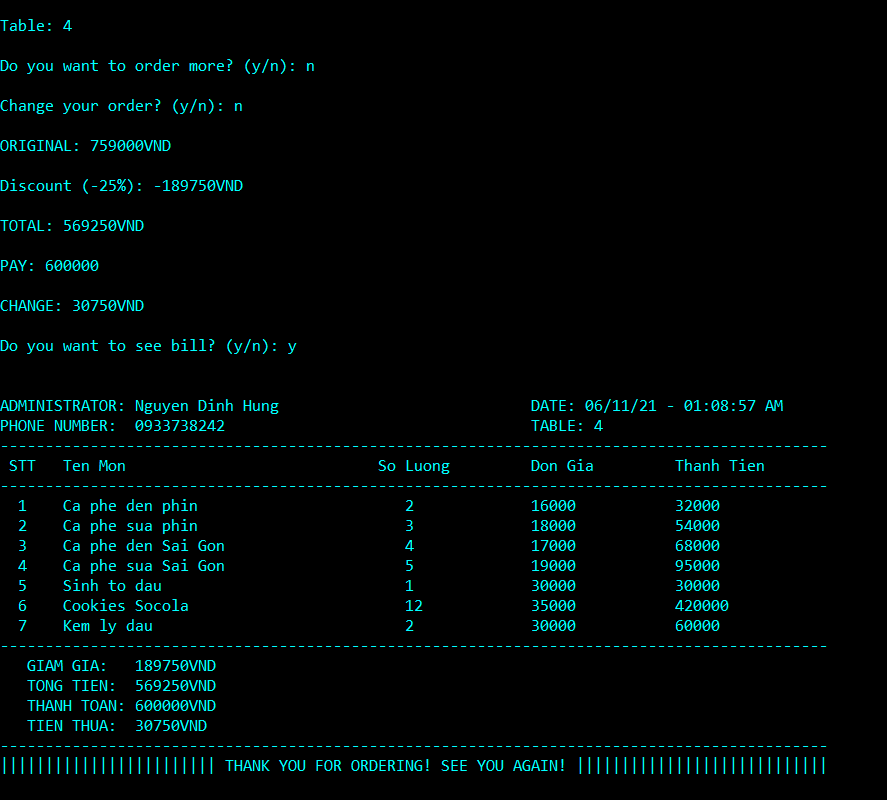
Hình 34. Giao diện khi nhập lựa chọn “n” – không in ra bill



Hình 35. Giao diện khi nhập lựa chọn “y” – in ra bill với tiền thanh toán bằng tổng tiền



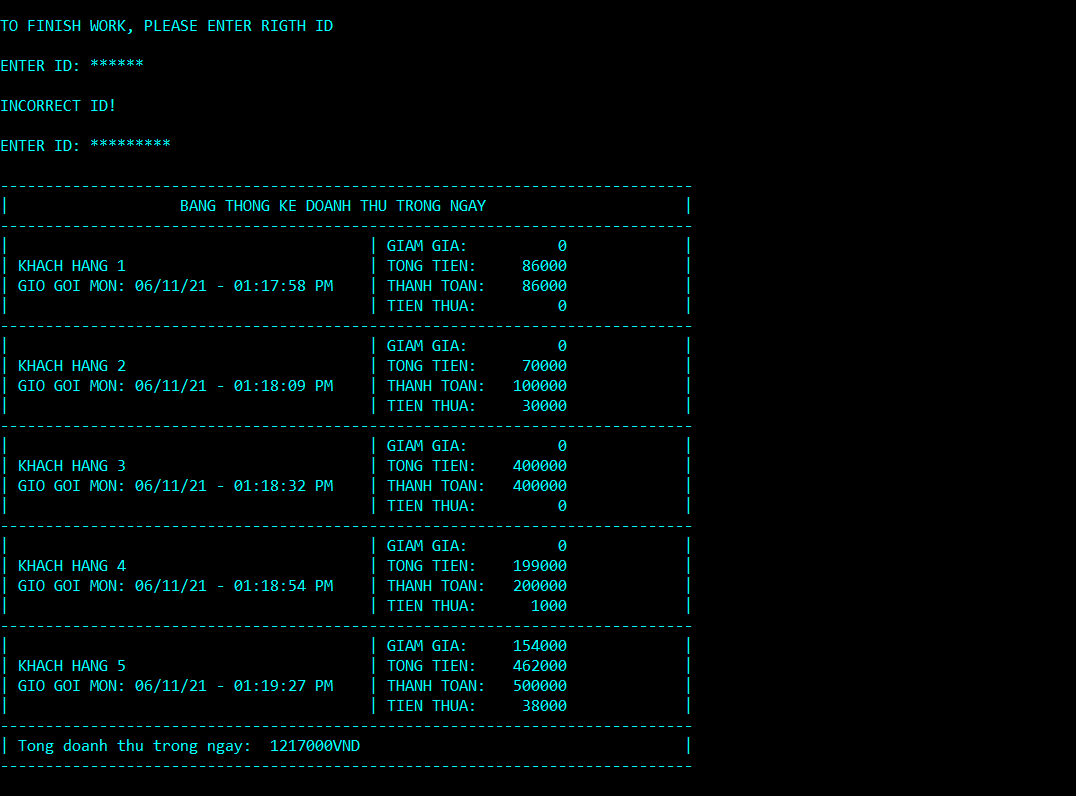
Hình 36. Giao diện khi nhập lựa chọn “y” – in ra bill với tiền thanh toán lớn hơn tổng tiền



Hình 37. Giao diện khi có hóa đơn được giảm giá và in bill hóa đơn đó.



Hình 38. Giao diện đăng xuất

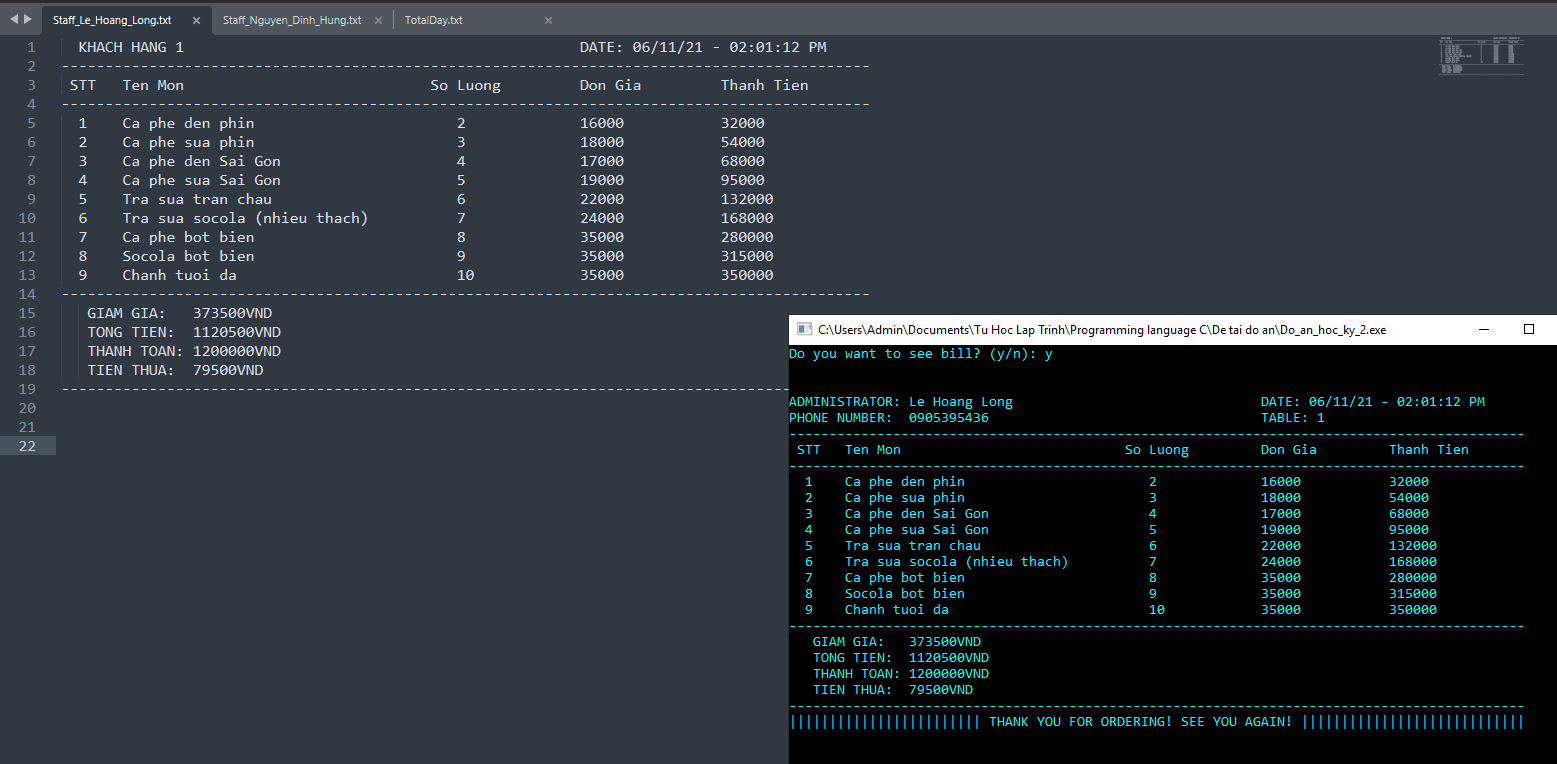


Hình 39. Giao diện bảng thống kê doanh thu trong ngày làm việc

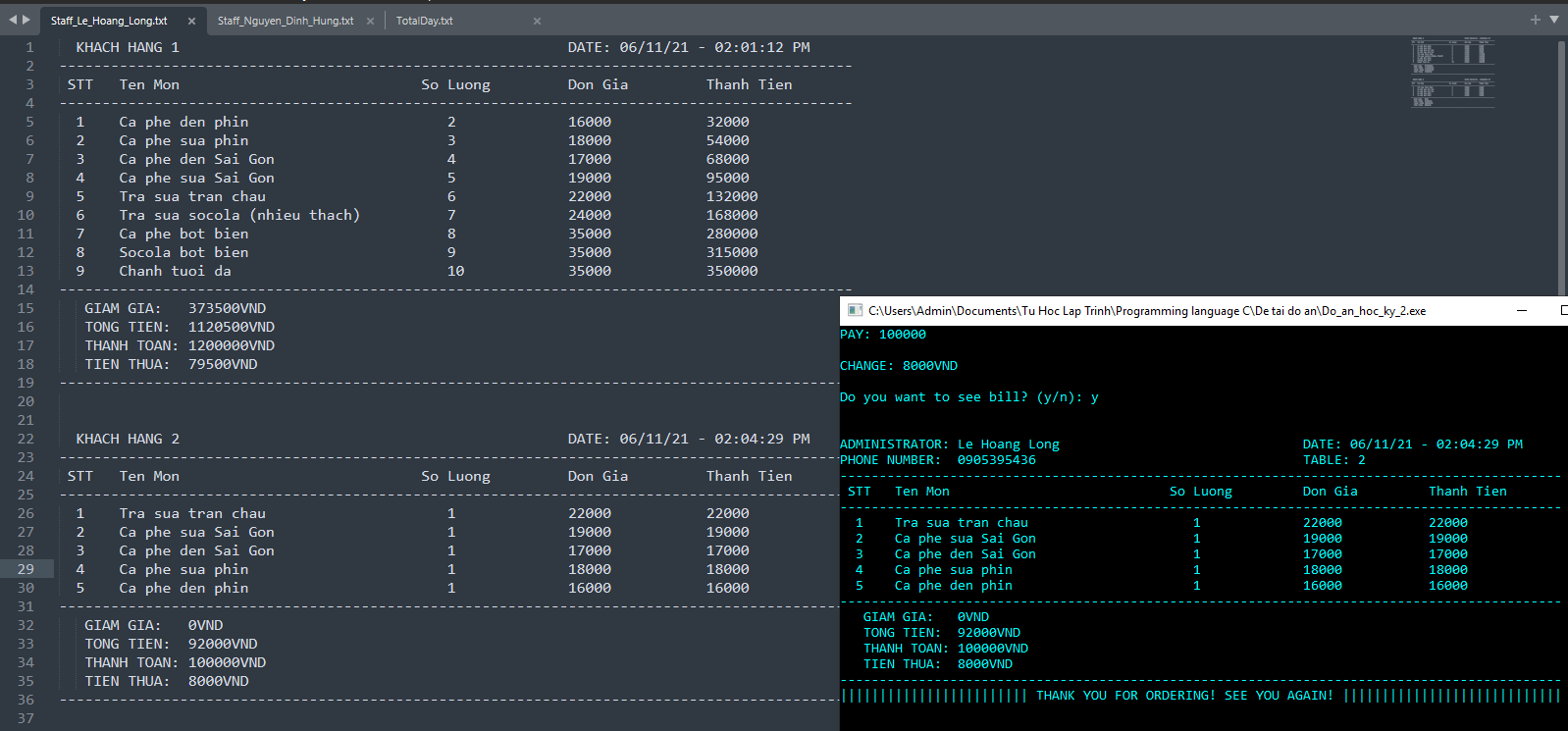
* + 1. Kết quả thực thi của chương trình

- Xuyên suốt quá trình làm việc, dữ liệu của mỗi khách hàng sẽ được ghi đè lên file. Như vậy, nhân viên có thể nhập thông tin order của khách hàng và sau khi thanh toán xong, dữ liệu của khách hàng đó sẽ được lưu trong file mà không bị mất đi, đảm bảo không bị thiếu sót. Mỗi nhân viên sẽ có một file tương ứng để lưu dữ liệu phiên làm việc của người đó trong ngày.

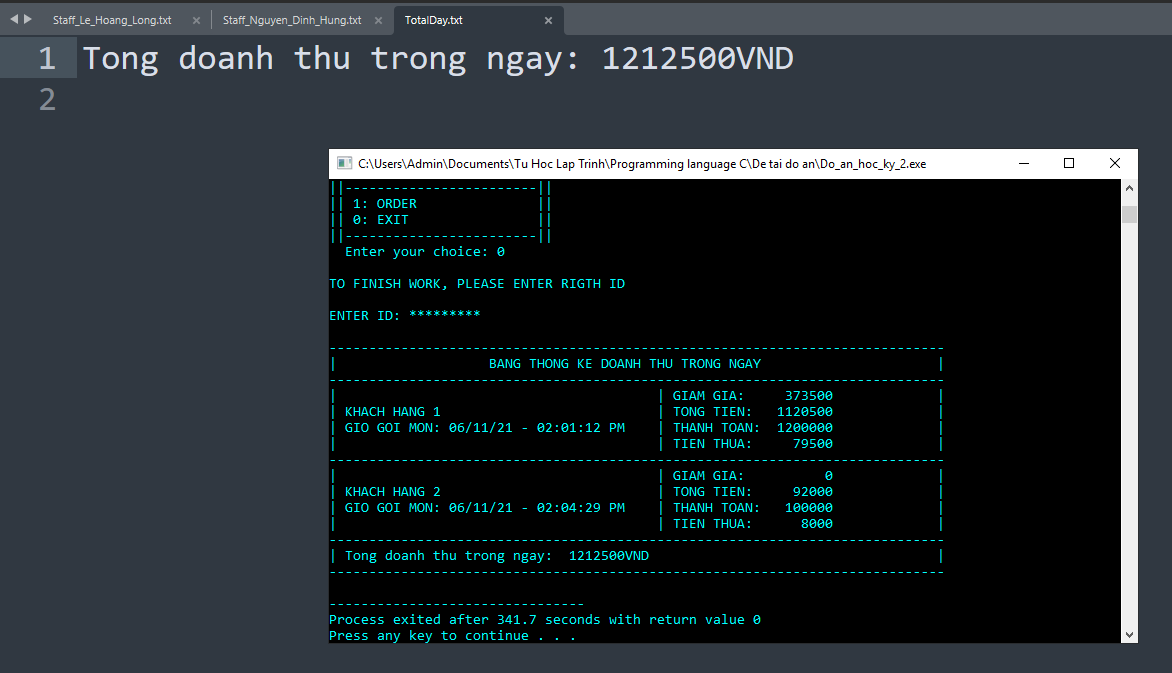
- Kết thúc phiên làm việc, chương trình sẽ in ra màn hình bảng thống kê doanh thu trong ngày và lưu tổng doanh thu vào file TotalDay.txt



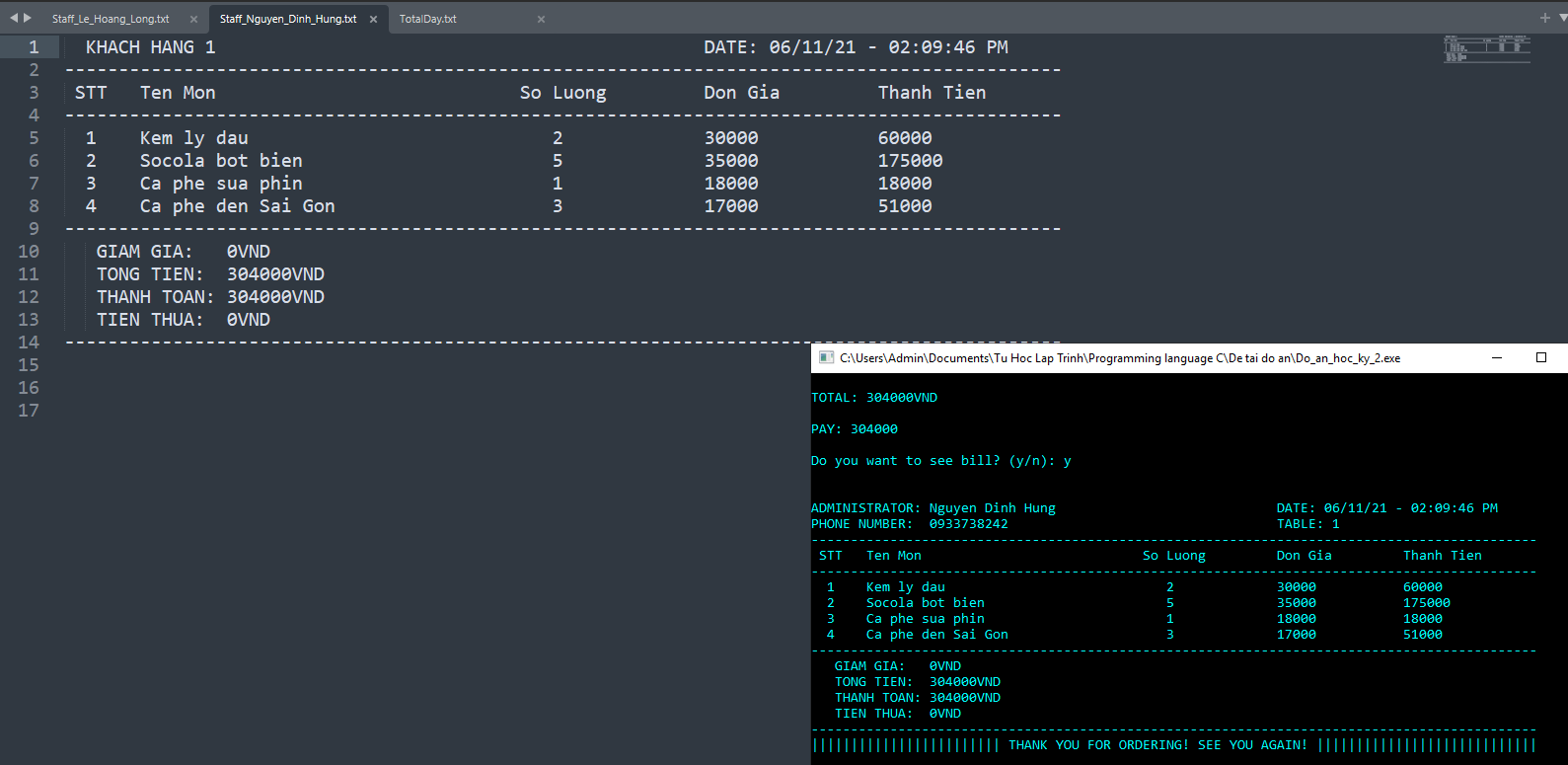
Hình 40. Dữ liệu của khách hàng 1 được ghi ra file với phiên làm việc của nhân viên 1



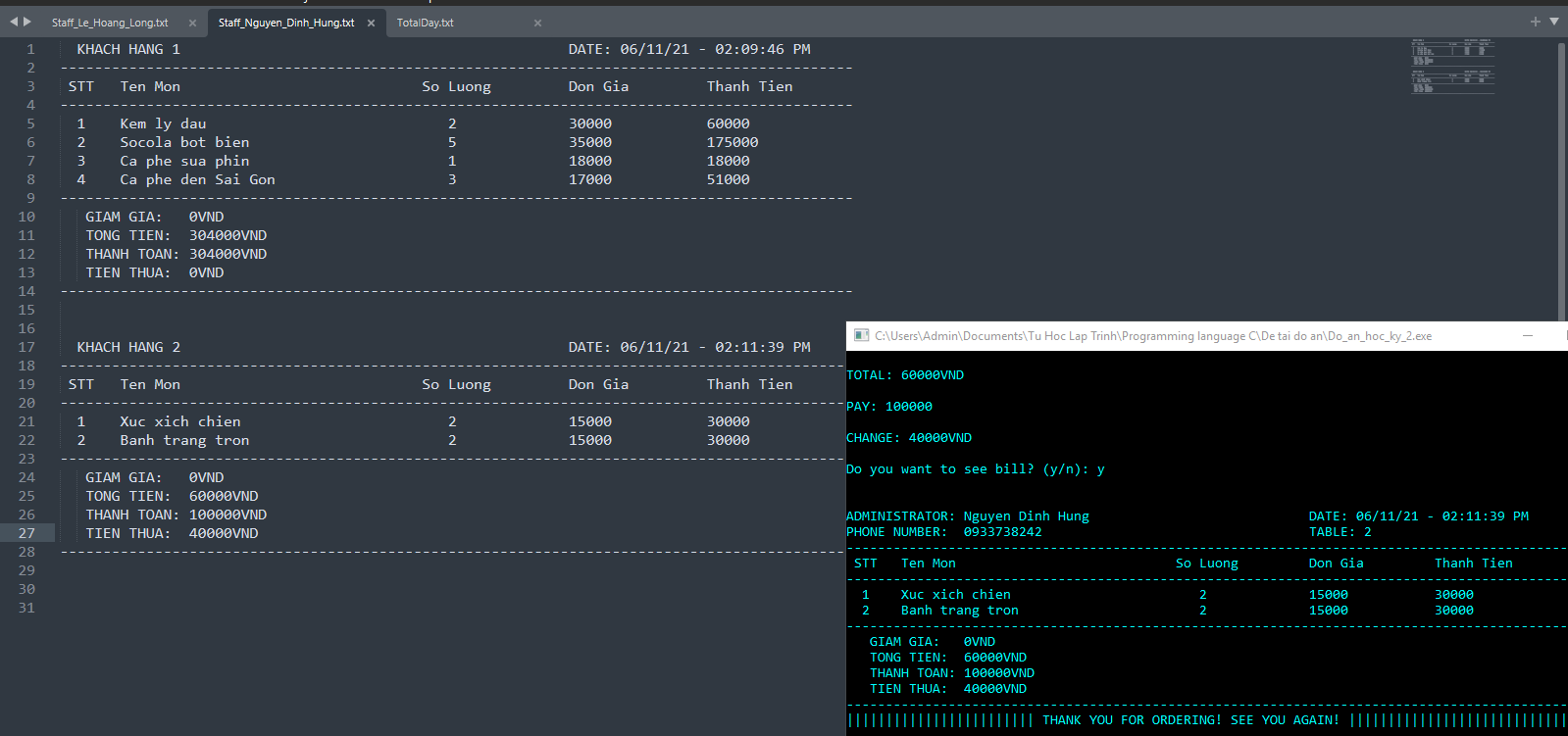
Hình 41. Dữ liệu của khách hàng 2 được ghi ra file với phiên làm việc của nhân viên 1



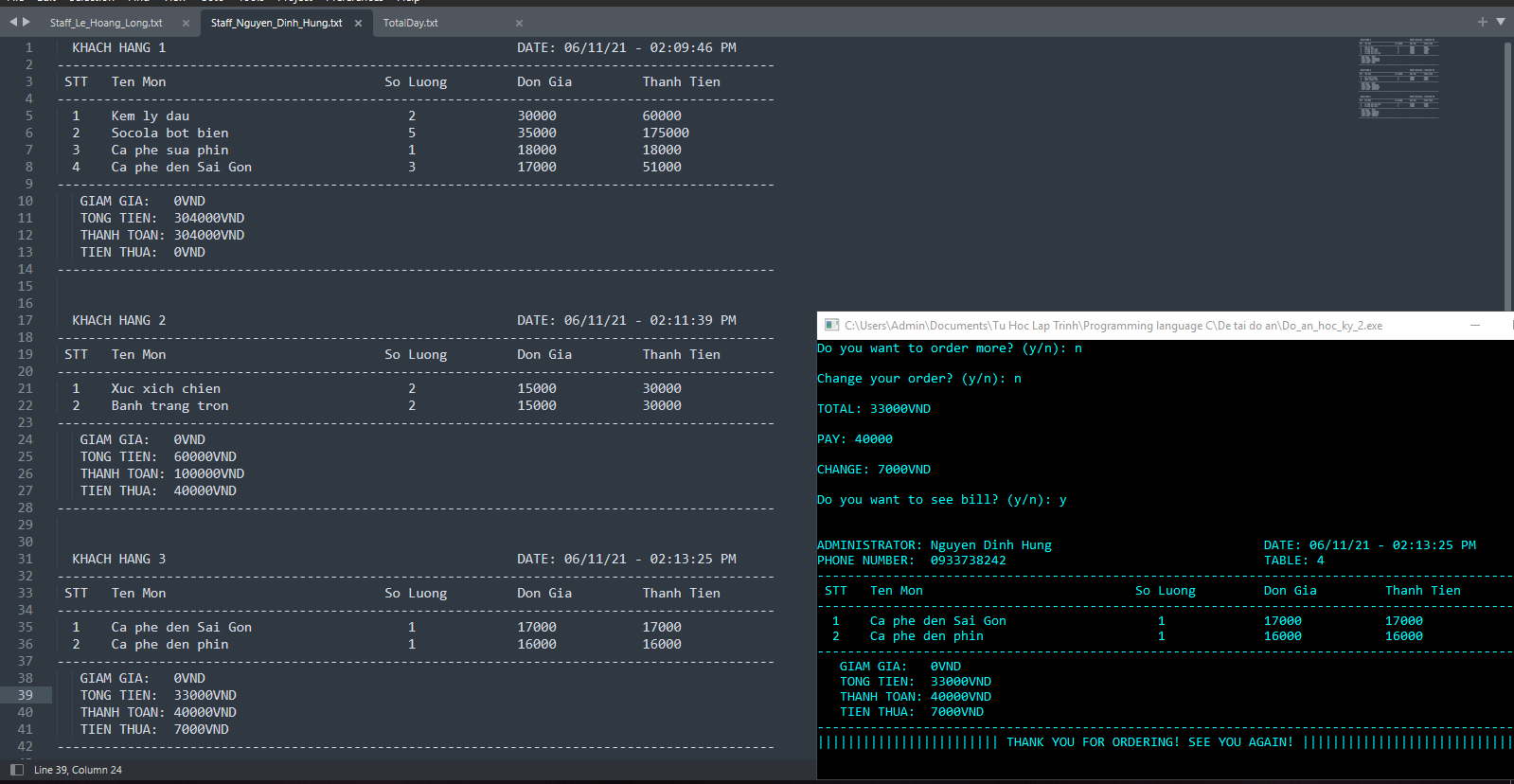
Hình 42. Tổng doanh thu trong ngày của phiên làm việc nhân viên 1 được lưu vào file TotalDay.txt



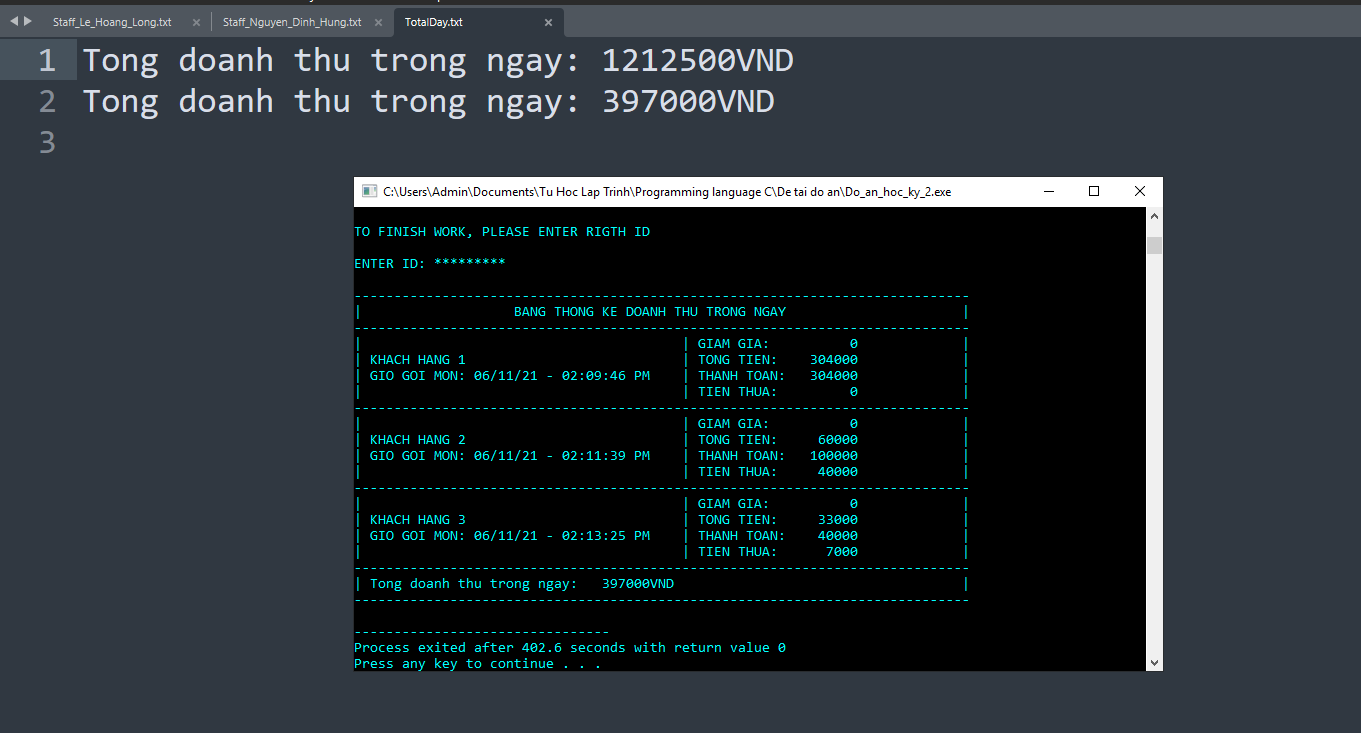
Hình 43. Dữ liệu của khách hàng 1 được ghi ra file với phiên làm việc của nhân viên 2



Hình 44. Dữ liệu của khách hàng 2 được ghi ra file với phiên làm việc của nhân viên 2



Hình 45. Dữ liệu của khách hàng 3 được ghi ra file với phiên làm việc của nhân viên 2



Hình 46. Tổng doanh thu trong ngày của phiên làm việc nhân viên 2 được lưu vào file TotalDay.txt

* + 1. Nhận xét đánh giá

- Về cơ bản, chúng em đã xây dựng nên một chương trình và hoạt động đúng ý tưởng ban đầu của đề tài như:

+ Nhập vào mã giao dịch ID để khởi động menu và bắt đầu phiên làm việc.

+ Nhập mã món và số lượng, nhập 0 0 để kết thúc tiến trình chọn món.

+ Được giảm giá nếu tổng đơn hàng đạt đủ điều kiện.

+ In hóa đơn của khách hàng.

+ Kết thúc phiên làm việc, nhập lại mã giao dịch ID. Thống kê doanh thu trong ngày và lưu dữ liệu vào file.

- Ngoài các ý tưởng trên, nhóm em đã phát triển thêm một số tính năng mới nhằm phát triển chương trình được tốt hơn. Có thể kể đến là:

+ Tạo tài khoản (username) tương ứng với mỗi nhân viên trong quán. Mã ID khi được nhập từ bàn phím sẽ chuyển thành kí tự \*, giúp tăng tính bảo mật hơn.

+ In hóa đơn phải có ngày giờ thanh toán

+ Phát triển tính năng xử lí trường hợp khách hàng có thể gọi thêm món.

+ Phát triển tính năng thay đổi thông tin món ăn của khách hàng.

+ Không giới hạn số lần gọi món. Khách hàng có thể gọi bao nhiêu món tùy thích, không bị giới hạn gọi 5 món tối đa như ý tưởng ban đầu.

1. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN
   1. Kết luận

- Đồ án lập trình tính toán này đã giúp chúng em củng cố và học hỏi thêm các kiến thức nâng cao của ngôn ngữ lập trình C. Từ các lệnh thao tác cơ bản, các câu lệnh điều kiện, các vòng lặp for, while và do-while đến các kiến thức về mảng 1 chiều, mảng 2 chiều, con trỏ, cấp phát động, thao tác xử lí chuỗi, thao tác dữ liệu file v.v… Ban đầu, nhóm em đã gặp nhiều khó khăn vì không biết nên bắt đầu từ đâu, tổ chức xây dựng chương trình như thế nào. Nhưng nhờ sự giúp đỡ tận tình, chu đáo của cô Trần Hồ Thủy Tiên đã giúp chúng em hoàn thành tốt đồ án lần này.

- Mặc dù đã dành nhiều tâm huyết và công sức để phát triển chương trình song vẫn không thể tránh khỏi các lỗi nhỏ nhặt. Chúng em hi vọng với sản phẩm đồ án này có thể đáp ứng được nhu cầu phát triển ngày càng cao của xã hội, sát nhập công nghệ gần hơn với đời sống.

* 1. Hướng phát triển

- Trong tương lai, chương trình của nhóm em có thể sẽ được mở rộng và phát triển hơn nữa nhằm đáp ứng nhu cầu càng cao và nâng cao trải nghiệm cho khách hàng. Một số tính năng như là:

+ Phát triển ứng dụng có sử dụng đồ họa nhằm mô phỏng không gian và vị trí đặt bàn của quán. Ứng dụng này sẽ giúp khách hàng có cái nhìn tổng quát hơn về không gian của quán cũng như biết được bàn nào đang có khách sử dụng, bàn nào đang trống.

+ Phát triển thêm tính năng thanh toán bằng thẻ tín dụng, thẻ ngân hàng v.v… Các thanh toán trong chương trình hiện chỉ dừng lại ở mức thanh toán bằng tiền mặt.

+ Phát triển các hệ thống tích lũy điểm và phần thưởng nhằm giúp khách hàng có trải nghiệm phong phú, tốt hơn.

- Ngoài hướng phát triển, nhóm em sẽ tiếp tục giải quyết những lỗi còn tồn tại trong chương trình (nếu có), các vấn đề còn tồn đọng như sử dụng nhiều bộ nhớ v.v…

TÀI LIỆU THAM KHẢO

**1. Thiết kế thuật toán nhập mật khẩu tạo kí tự \***

<https://www.youtube.com/watch?v=lko3n7NCYBU&t=57s>

**2. Đọc ghi file trong C**

<https://nguyenvanhieu.vn/doc-ghi-file-trong-c/>

<https://freetuts.net/doc-ghi-file-trong-c-2991.html>

**3. Cấp phát động trong C**

<https://nguyenvanhieu.vn/cap-phat-bo-nho-dong-trong-c/>

**4. Struct trong C**

<https://nguyenvanhieu.vn/struct-trong-c/>

PHỤ LỤC