**ĐẠI HỌC ĐÀ NẴNG**

**ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN**

**ĐỒ ÁN CƠ SỞ LẬP TRÌNH**

**Quản lý kí túc xá**

**GIẢNG VIÊN HƯỚNG DẪN: Th.S Lê Thị Mỹ Hạnh**

**SINH VIÊN THỰC HIỆN:**

**Châu Văn Thành Đạt \_ 102230338 LỚP: 23T-DT4 NHÓM: 11**

**Dương Phước Thành \_ 102230373 LỚP: 23T-DT4 NHÓM: 11**

**MỤC LỤC**

[1. PHÂN TÍCH CHỨC NĂNG HỆ THỐNG 1](#_Toc186163268)

[1.1. Giới thiệu đề tài 1](#_Toc186163269)

[1.2. Mục đích thực hiện đề tài 1](#_Toc186163270)

[1.3. Mục tiêu đề tài 1](#_Toc186163271)

[1.4. Nội dung, phạm vi và đối tượng thực hiện 1](#_Toc186163272)

[2. PHÂN TÍCH CHỨC NĂNG HỆ THỐNG 3](#_Toc186163273)

[2.1. Chức năng hệ thống 3](#_Toc186163274)

[2.1.1. Quản lý thông tin sinh viên 3](#_Toc186163275)

[2.1.2. Quản lý phòng ký túc xá 3](#_Toc186163276)

[2.1.3. Quản lý chi phí sinh hoạt 3](#_Toc186163277)

[2.1.4. Giao diện người dùng 3](#_Toc186163278)

[2.1.5. Bảo mật 3](#_Toc186163279)

[2.2. Cơ sở lý thuyết 4](#_Toc186163280)

[2.2.1 Ngôn ngữ lập trình C++ 4](#_Toc186163281)

[2.2.2 Lập trình hướng đối tượng (OOP) 4](#_Toc186163282)

[2.2.3. Cấu trúc dữ liệu 4](#_Toc186163283)

[2.2.4. Quản lý file và dữ liệu 5](#_Toc186163284)

[2.2.5. Thuật toán tìm kiếm và sắp xếp 5](#_Toc186163285)

[2.2.6. Quản lý bộ nhớ động 5](#_Toc186163286)

[2.2.7. Phân quyền và bảo mật 5](#_Toc186163287)

[2.2.8. Giao diện dòng lệnh (CLI) 6](#_Toc186163288)

[2.2.9. Kiểm thử (Testing) 6](#_Toc186163289)

[3. TỔ CHỨC CẤU TRÚC DỮ LIỆU VÀ THUẬT TOÁN 6](#_Toc186163290)

[3.1. Phát biểu bài toán 6](#_Toc186163291)

[3.2. Cấu trúc tổ chức dữ liệu đầu vào 8](#_Toc186163292)

[3.2.1 Dữ liệu vào ra 8](#_Toc186163293)

[3.2.2 Cấu trúc dữ liệu 9](#_Toc186163294)

[4. CHƯƠNG TRÌNH VÀ KẾT QUẢ 10](#_Toc186163295)

[4.1 Các Class: 10](#_Toc186163296)

[4.1.1.Giao diện chính của chương trình 12](#_Toc186163297)

[4.1.2.Nhận xét 14](#_Toc186163298)

[5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 14](#_Toc186163299)

[5.1. Kết luận 14](#_Toc186163300)

[5.2. Hướng phát triển 14](#_Toc186163301)

DANH MỤC HÌNH VẼ

[Hình 1: Sơ đồ quan hệ 8](#_Toc186162965)

[Hình 2: Đăng Nhập 12](#_Toc186162966)

[Hình 3: Giao diện khi đăng nhập thành công 13](#_Toc186162967)

[Hình 4: Thêm Hợp Đồng 13](#_Toc186162968)

[Hình 5: Xem danh sách sinh viên 13](#_Toc186162969)

[Hình 6: Xem danh sách phòng 13](#_Toc186162970)

[Hình 7: Thêm sinh viên 13](#_Toc186162971)

[Hình 8: Thêm phòng 13](#_Toc186162972)

[Hình 9: Xem danh sách cho xếp phòng 13](#_Toc186162973)

[Hình 10: Xem danh sách hợp đồng 14](#_Toc186162974)

[Hình 11: Thoát khỏi chương trình 14](#_Toc186162975)

LỜI MỞ ĐẦU

Hiện nay, các trường đại học, cao đẳng thường có hàng trăm đến hàng nghìn sinh viên cư trú trong ký túc xá, là nơi tập trung nhiều thông tin và hoạt động cần quản lý. Việc quản lý lượng lớn thông tin về sinh viên, phòng ở, và các hoạt động liên quan trở nên phức tạp và dễ xảy ra sai sót nếu không có hệ thống quản lý tự động. Nhiều trường đại học vẫn còn đang sử dụng các phương pháp quản lý thủ công, gây khó khăn trong việc theo dõi và xử lý thông tin. Một số vấn đề thường gặp phải kể đến như là: Các ký túc xá thường sử dụng bảng biểu giấy hoặc file Excel để quản lý thông tin sinh viên. Thiếu công nghệ hỗ trợ và khó khăn trong việc kiểm soát số lượng sinh viên cư trú. Điều này dễ dẫn đến sai sót, khó tra cứu và cập nhật kịp thời. Khó khăn trong việc phân bổ phòng ở, việc sắp xếp chỗ ở cho sinh viên thường gặp vấn đề khi ký túc xá không có công cụ hỗ trợ tự động phân bổ phòng, dẫn đến tình trạng quá tải ở một số khu vực. Khi có các vấn đề phát sinh như sự cố hạ tầng hoặc thắc mắc, việc liên lạc và xử lý yêu cầu giữa sinh viên và ban quản lý thường chậm chạp do thiếu tính tương tác giữa ban quản lý và sinh viên.

Việc tạo ra một hệ thống quản lý ký túc xá được xây dựng nhằm khắc phục những hạn chế của phương pháp quản lý thủ công, giúp lưu trữ, tra cứu và xử lý thông tin một cách chính xác, nhanh chóng. Một hệ thống quản lý ký túc xá sẽ giúp ban quản lý nắm bắt thông tin sinh viên một cách nhanh chóng và chính xác hơn, giảm thiểu các lỗi do thao tác thủ công. Tự động phân tích các yêu cầu và số lượng sinh viên, từ đó phân bổ phòng ở một cách hợp lý, tránh tình trạng quá tải. Hệ thống không chỉ giúp quản lý thông tin sinh viên một cách chi tiết mà còn hỗ trợ sinh viên trong việc tra cứu thông tin cá nhân, kiểm tra tình trạng phòng ở, và các khoản phí phải thanh toán, bên cạnh đó c có thể tích hợp chức năng thông báo, phản hồi giữa sinh viên và ban quản lý, giúp tăng cường tương tác và giải quyết nhanh chóng các vấn đề phát sinh. Việc sử dụng công nghệ để quản lý giúp tiết kiệm thời gian, nhân lực và nguồn lực vật chất, tạo điều kiện thuận lợi cho sự phát triển bền vững của ký túc xá. Với những lợi ích được nêu trên, việc phát triển một chương trình quản lý ký túc xá là vô cùng cần thiết nhằm giải quyết các khó khăn hiện tại và nâng cao chất lượng dịch vụ cho sinh viên.

# PHÂN TÍCH CHỨC NĂNG HỆ THỐNG

# 1.1. Giới thiệu đề tài

Đề tài "Chương trình quản lý ký túc xá" hướng tới việc xây dựng một hệ thống phần mềm quản lý chỗ ở của sinh viên tại ký túc xá một cách hiệu quả và chính xác. Việc quản lý ký túc xá bằng phương pháp thủ công thường dễ gặp sai sót, mất thời gian, và khó kiểm soát thông tin liên quan đến sinh viên, phòng ở, cũng như các khoản thu chi. Vì vậy, một hệ thống phần mềm tự động sẽ giúp ban quản lý ký túc xá thực hiện công việc này dễ dàng hơn.

# 1.2. Mục đích thực hiện đề tài

Chương trình quản lý ký túc xá nhằm mục đích tối ưu hóa quá trình quản lý và vận hành ký túc xá, giúp ban quản lý có thể dễ dàng theo dõi thông tin sinh viên, phòng ở, các khoản thu chi, cũng như đảm bảo việc phân bổ và sử dụng tài nguyên ký túc xá một cách hiệu quả. Ngoài ra, chương trình cũng giúp sinh viên có thể tra cứu thông tin cá nhân, yêu cầu thay đổi phòng ở, và kiểm tra các khoản phí một cách tiện lợi.

# 1.3. Mục tiêu đề tài

* Tạo ra một hệ thống phần mềm giúp quản lý ký túc xá hiệu quả, giảm thiểu sai sót trong việc quản lý thông tin sinh viên, phòng ở, và các khoản phí.
* Giúp ban quản lý dễ dàng truy cập, cập nhật thông tin về sinh viên, phòng ở, và lịch sử thanh toán.
* Hỗ trợ sinh viên tra cứu thông tin cá nhân và quản lý chỗ ở một cách tiện lợi.
* Đảm bảo an ninh và bảo mật thông tin của sinh viên và ký túc xá.

# 1.4. Nội dung, phạm vi và đối tượng thực hiện

* + **Nội dung và chức năng chính của hệ thống:**Hệ thống quản lý ký túc xá sẽ bao gồm các chức năng chính như sau:
* **Quản lý sinh viên**: Lưu trữ thông tin cá nhân của từng sinh viên, bao gồm mã số sinh viên, tên, ngày sinh, lớp học, phòng đang ở và trạng thái thanh toán.
* **Quản lý phòng ở**: Theo dõi thông tin phòng ở trong ký túc xá như số phòng, loại phòng, số lượng sinh viên tối đa và danh sách sinh viên đang cư trú.
* **Quản lý thanh toán**: Ghi nhận các khoản phí mà sinh viên cần thanh toán, tình trạng thanh toán và tạo báo cáo thu chi.
* **Tương tác sinh viên**: Sinh viên có thể xem thông tin cá nhân, phòng ở, lịch sử thanh toán và gửi yêu cầu thay đổi phòng hoặc phản ánh về điều kiện ký túc xá.
  + **Đối tượng sử dụng ứng dụng**:
* **Ban quản lý ký túc xá**: Là người sử dụng chính, có quyền truy cập vào tất cả các chức năng của hệ thống, bao gồm quản lý sinh viên, phòng ở, và các khoản thu chi.
* **Sinh viên**: Là người ở trong ký túc xá, có thể sử dụng ứng dụng để tra cứu thông tin cá nhân, kiểm tra phòng ở, thanh toán phí ký túc xá, và gửi yêu cầu thay đổi phòng ở.
  + **Đối tượng quản lý trong hệ thống**:
* **Sinh viên**: Là những người đang ở trong ký túc xá. Các thuộc tính bao gồm: mã sinh viên, họ tên, ngày sinh, lớp học, phòng đang ở, các khoản phí phải trả. Hành vi bao gồm: tra cứu thông tin cá nhân, kiểm tra khoản phí, gửi yêu cầu thay đổi phòng.
* **Phòng ở**: Là các phòng trong ký túc xá. Các thuộc tính bao gồm: số phòng, loại phòng (phòng đơn, phòng đôi, v.v.), số lượng sinh viên tối đa, danh sách sinh viên đang ở. Hành vi bao gồm: thêm sinh viên vào phòng, xóa sinh viên khỏi phòng.
* **Quản lý thanh toán**: Các khoản phí sinh viên phải trả liên quan đến việc ở trong ký túc xá. Các thuộc tính bao gồm: mã sinh viên, số tiền, ngày thanh toán, trạng thái thanh toán (đã thanh toán/chưa thanh toán). Hành vi bao gồm: cập nhật trạng thái thanh toán, kiểm tra lịch sử thanh toán.
  + **Mối quan hệ giữa các đối tượng**:
* Mối quan hệ giữa **sinh viên** và **phòng ở** là quan hệ **1-1**: một sinh viên chỉ có thể có một phòng, và một phòng có thể có nhiều sinh viên.
* Mối quan hệ giữa **sinh viên** và **quản lý thanh toán** là quan hệ **một-nhiều**: mỗi sinh viên sẽ có nhiều khoản phí cần thanh toán trong suốt thời gian ở ký túc xá.
  + **Dự kiến cấu trúc dữ liệu**:
* **Sinh viên**: Sử dụng cấu trúc bảng trong cơ sở dữ liệu với các thuộc tính như mã sinh viên, họ tên, ngày sinh, giới tính, thông tin liên lạc, phòng ở hiện tại, tình trạng thanh toán.

“ma\_sinh\_vien”: “string”,

“ho\_ten”: “string”,

“ngay\_sinh”: “date”

“gioi\_tinh”: “string”,

“thong\_tin\_lien\_lac”: “string”,

“ma\_phong”: “string”,

“tinh\_trang\_thanh\_toan”: “string”.

* **Phòng ở**: Mỗi phòng sẽ có bảng thông tin riêng với các thuộc tính như số phòng, loại phòng, số lượng sinh viên tối đa, danh sách sinh viên, tình trạng phòng.

“ma\_phong”: “string”,

“loai\_phong”: “string”,

“so\_luong\_sinh\_vien\_toi\_da”: “integer”,

“tinh\_trang\_phong”: “string”.

“danh\_sach\_sinh\_vien”: “”.

...

* **Thanh toán**: Một bảng dữ liệu chứa các thông tin thanh toán của từng sinh viên, bao gồm mã sinh viên, số tiền, ngày thanh toán, và trạng thái thanh toán.

“ma\_sinh\_vien”: “string”,

“so\_tien”: “float”,

“ngay\_thanh\_toan”: “date”,

“tinh\_trang\_thanh\_toan”: “string”,

* + **Cấu trúc file lưu trữ thông tin**:
* File thông tin sinh viên: Gồm mã sinh viên, họ tên, ngày sinh, lớp học, phòng ở hiện tại.
* File thông tin phòng ở: Gồm số phòng, loại phòng, danh sách sinh viên.
* File thông tin thanh toán: Gồm mã sinh viên, số tiền, ngày thanh toán, trạng thái thanh toán.
* Mỗi file sẽ lưu trữ một danh sách các đối tượng tương ứng và có thể được truy cập và sửa đổi dễ dàng bởi ứng dụng, đồng thời đảm bảo tính toàn vẹn và bảo mật của thông tin.

# PHÂN TÍCH CHỨC NĂNG HỆ THỐNG

## 2.1. Chức năng hệ thống

### 2.1.1. Quản lý thông tin sinh viên

* + **Chức năng chính**: Chương trình sẽ quản lý thông tin của sinh viên, bao gồm các thông tin cá nhân và chi tiết liên quan đến ký túc xá.
  + **Tính năng cụ thể**:
* Thêm, sửa, xóa thông tin sinh viên.
* Tìm kiếm sinh viên theo mã sinh viên, tên, hoặc phòng ở.
* Hiển thị danh sách sinh viên theo từng phòng hoặc từng dãy nhà.

### 2.1.2. Quản lý phòng ký túc xá

* + **Chức năng chính**: Chương trình sẽ theo dõi thông tin phòng ký túc xá, bao gồm số phòng, số lượng giường, và danh sách sinh viên ở trong mỗi phòng.
  + **Tính năng cụ thể**:
* Quản lý trạng thái phòng (phòng còn trống hay đầy).
* Tìm kiếm phòng trống hoặc kiểm tra phòng của một sinh viên cụ thể.

### 2.1.3. Quản lý chi phí sinh hoạt

* + **Chức năng chính**: Theo dõi và quản lý các chi phí sinh hoạt của sinh viên như tiền phòng, điện, nước, internet, v.v.
  + **Tính năng cụ thể**:
* Tạo hóa đơn hàng tháng cho từng sinh viên.
* Tính toán chi phí dựa trên số liệu điện nước thực tế.
* Gửi thông báo khi sắp đến hạn thanh toán.

### 2.1.4. Giao diện người dùng

* + **Chức năng chính**: Cung cấp giao diện dễ sử dụng để tương tác với hệ thống.
  + **Tính năng cụ thể**:
* Xây dựng giao diện console đơn giản, dễ sử dụng, có menu điều hướng.
* Sử dụng các cấu trúc dữ liệu như mảng, vector, danh sách liên kết để quản lý thông tin.
* Tương tác với người dùng qua các tùy chọn như nhập, tìm kiếm, sửa, xóa thông tin.

### 2.1.5. Bảo mật

* + **Chức năng chính**: Đảm bảo an toàn dữ liệu và phân quyền truy cập cho các người dùng khác nhau (quản trị viên, sinh viên).
  + **Tính năng cụ thể**:
* Mã hóa dữ liệu quan trọng như thông tin cá nhân và chi phí.
* Phân quyền truy cập (chỉ quản trị viên mới được phép thay đổi thông tin phòng hoặc sinh viên).
* **Công nghệ và phương pháp:**
  + **Ngôn ngữ C++**: Sử dụng để lập trình hệ thống với các tính năng như quản lý file, tạo giao diện dòng lệnh.
  + **Cấu trúc dữ liệu** Danh sách liên kết, bảng băm có thể được sử dụng để lưu trữ và xử lý dữ liệu hiệu quả.
  + **Xử lý file**: Lưu trữ thông tin sinh viên, phòng ở, và chi phí trong file text hoặc file CSV, sau đó đọc và ghi dữ liệu từ đó.

## 2.2. Cơ sở lý thuyết

### 2.2.1 Ngôn ngữ lập trình C++

* **Đặc điểm**: C++ là ngôn ngữ lập trình hướng đối tượng (OOP), hỗ trợ cả lập trình cấu trúc và hướng đối tượng, giúp xây dựng các ứng dụng có cấu trúc phức tạp.
* **Ứng dụng trong dự án**: Trong dự án quản lý ký túc xá, C++ sẽ được sử dụng để xây dựng chương trình với các tính năng quản lý sinh viên, phòng, và chi phí. Các khái niệm quan trọng trong C++ bao gồm:
  + **Biến và kiểu dữ liệu**: Các loại dữ liệu cơ bản như int, float, string, cùng với các biến phức hợp như struct, class sẽ được sử dụng để lưu trữ thông tin sinh viên và phòng ký túc xá.
  + **Hàm (functions)**: Các hàm sẽ giúp chia nhỏ chương trình thành các đơn vị chức năng riêng biệt, chẳng hạn như hàm thêm sinh viên, tìm kiếm phòng, và tính toán chi phí.
  + **Quản lý bộ nhớ**: C++ cho phép quản lý bộ nhớ động bằng con trỏ (pointers), cần thiết khi quản lý dữ liệu lớn trong ký túc xá (danh sách sinh viên, phòng).

### 2.2.2 Lập trình hướng đối tượng (OOP)

* **Lý thuyết**: Lập trình hướng đối tượng giúp tổ chức chương trình dưới dạng các đối tượng có các thuộc tính (data) và phương thức (method).
* **Ứng dụng trong dự án**:
  + **Lớp (class)**: Dự án có thể định nghĩa các lớp như Student (Sinh viên) và Room (Phòng), trong đó mỗi lớp sẽ có các thuộc tính như name, studentID, roomNumber, và các phương thức để thực hiện các thao tác như thêm, xóa, hoặc tìm kiếm thông tin.
  + **Tính đóng gói (Encapsulation)**: Đảm bảo dữ liệu bên trong đối tượng chỉ có thể được truy cập và sửa đổi thông qua các phương thức được xác định rõ.
  + **Kế thừa (Inheritance)**: Có thể tạo các lớp con kế thừa từ lớp cha, ví dụ như Room có thể là lớp con kế thừa từ lớp Building để tận dụng các thuộc tính chung.
  + **Tính đa hình (Polymorphism)**: Giúp các phương thức có thể hoạt động trên nhiều đối tượng khác nhau, chẳng hạn phương thức calculateFees() có thể được áp dụng cho cả sinh viên trong nhiều loại phòng khác nhau.

### 2.2.3. Cấu trúc dữ liệu

* **Lý thuyết**: Các cấu trúc dữ liệu là thành phần quan trọng trong quản lý và tổ chức dữ liệu hiệu quả.
* **Ứng dụng trong dự án**:
  + **Vector**: Một kiểu dữ liệu động trong C++, dùng để quản lý danh sách sinh viên/phòng với kích thước không cố định, dễ dàng thêm hoặc xóa phần tử.
  + **Danh sách liên kết (Linked List)**: Nếu dữ liệu thay đổi liên tục, có thể sử dụng danh sách liên kết để quản lý thông tin.
  + **Bảng băm (Hash Table)**: Giúp tối ưu hóa việc tìm kiếm sinh viên hoặc phòng bằng cách sử dụng mã sinh viên/phòng làm khóa (key) để truy cập nhanh.

### 2.2.4. Quản lý file và dữ liệu

* **Lý thuyết**: Quản lý file là một kỹ năng quan trọng khi xây dựng các hệ thống lưu trữ dữ liệu dài hạn.
* **Ứng dụng trong dự án**:
  + **Đọc và ghi file**: Chương trình cần đọc và ghi thông tin sinh viên, phòng và chi phí vào các file văn bản hoặc file CSV. Trong C++, thao tác với file thường thông qua các luồng vào/ra (ifstream và ofstream).
  + **Xử lý file tuần tự**: Khi mở và thao tác file, dữ liệu sẽ được xử lý tuần tự, mỗi dòng tương ứng với một thông tin sinh viên hoặc phòng.

### 2.2.5. Thuật toán tìm kiếm và sắp xếp

* **Lý thuyết**: Các thuật toán tìm kiếm và sắp xếp giúp tối ưu hóa việc truy xuất và quản lý dữ liệu trong hệ thống.
* **Ứng dụng trong dự án**:
  + **Tìm kiếm tuyến tính (Linear Search)**: Có thể dùng để tìm kiếm sinh viên hoặc phòng trong các danh sách nhỏ.
  + **Tìm kiếm nhị phân (Binary Search)**: Nếu dữ liệu đã được sắp xếp (theo mã sinh viên hoặc số phòng), tìm kiếm nhị phân sẽ giúp tăng tốc độ tìm kiếm.
  + **Sắp xếp (Sorting)**: Thuật toán sắp xếp (như Bubble Sort, Quick Sort, hoặc Merge Sort) có thể được sử dụng để sắp xếp danh sách sinh viên theo tên, mã số, hoặc các tiêu chí khác.

### 2.2.6. Quản lý bộ nhớ động

* **Lý thuyết**: Bộ nhớ động trong C++ giúp chương trình phân bổ và giải phóng bộ nhớ khi cần thiết (sử dụng new và delete).
* **Ứng dụng trong dự án**: Khi quản lý một danh sách lớn các đối tượng (sinh viên hoặc phòng), sử dụng bộ nhớ động giúp tối ưu việc quản lý bộ nhớ và giảm thiểu tình trạng lãng phí tài nguyên.

### 2.2.7. Phân quyền và bảo mật

* **Lý thuyết**: Đảm bảo an toàn và tính toàn vẹn của dữ liệu là yếu tố quan trọng trong mọi hệ thống quản lý.
* **Ứng dụng trong dự án**:
  + **Phân quyền**: Các vai trò khác nhau (quản trị viên, sinh viên) có quyền truy cập và chỉnh sửa thông tin khác nhau.
  + **Mã hóa**: Dữ liệu nhạy cảm như mật khẩu hoặc thông tin tài chính cần được mã hóa khi lưu trữ và truyền tải.

### 2.2.8. Giao diện dòng lệnh (CLI)

* **Lý thuyết**: Giao diện dòng lệnh là một phương thức tương tác với chương trình thông qua các lệnh văn bản.
* **Ứng dụng trong dự án**:
  + Xây dựng một giao diện người dùng đơn giản, hiển thị các menu tùy chọn và hướng dẫn người dùng nhập lệnh để thực hiện các chức năng như thêm, tìm kiếm, hoặc xóa sinh viên/phòng.

### 2.2.9. Kiểm thử (Testing)

* **Lý thuyết**: Kiểm thử giúp đảm bảo chương trình hoạt động đúng với yêu cầu.
* **Ứng dụng trong dự án**:
  + Thực hiện kiểm thử đơn vị (unit testing) cho từng chức năng như thêm sinh viên, tính toán chi phí.
  + Kiểm thử hệ thống để đảm bảo các chức năng hoạt động tốt khi tích hợp.

# TỔ CHỨC CẤU TRÚC DỮ LIỆU VÀ THUẬT TOÁN

## 3.1. Phát biểu bài toán

Hệ thống quản lý ký túc xá là một bài toán phổ biến trong việc xây dựng hệ thống quản lý cho các cơ sở có số lượng người lưu trú lớn, như các trường học, khu công nghiệp hay khu căn hộ. Mục tiêu của bài toán là xây dựng một hệ thống quản lý các thông tin liên quan đến việc sắp xếp, phân phối và quản lý sinh viên (hoặc cư dân) trong ký túc xá.

**Mô hình quan hệ**

Mô hình quan hệ (ERD - Entity Relationship Diagram) giúp biểu diễn các thực thể và mối quan hệ giữa chúng trong hệ thống quản lý ký túc xá. Dưới đây là một mô hình quan hệ đơn giản cho bài toán quản lý ký túc xá.

* **Các thực thể chính:**

**Sinh viên (Student):**

* + **Mã sinh viên** (StudentID - khóa chính)
  + Họ tên (FullName)
  + Ngày sinh (DOB)
  + Giới tính (Gender)
  + Số điện thoại (PhoneNumber)
  + Mã phòng (RoomID - khóa ngoại)

**Phòng ký túc xá (Room):**

* + **Mã phòng** (RoomID - khóa chính )
  + Số lượng giường (TotalBeds)
  + Số giường còn trống (AvailableBeds)
  + Loại phòng (RoomType - đơn, đôi, ba,...)
  + Giá phòng (RoomPrice)

**Hóa đơn (Invoice):**

* + **Mã hóa đơn** (InvoiceID - khóa chính)
  + Mã sinh viên (StudentID - khóa ngoại)
  + Ngày lập hóa đơn (InvoiceDate)
  + Tổng tiền (TotalAmount)
  + Chi tiết (Description)

**Người quản lý (Manager):**

* + **Mã quản lý** (ManagerID - khóa chính)
  + Họ tên (FullName)
  + Số điện thoại (PhoneNumber)
  + Mã khu vực quản lý (BuildingID)
* **Mối quan hệ giữa các thực thể:**

**Sinh viên - Phòng:**

* + Một **sinh viên** sẽ được phân ở **một phòng** duy nhất tại một thời điểm (quan hệ 1-n).
  + Một **phòng** có thể chứa nhiều **sinh viên** (quan hệ 1-n).

**Phòng - Hóa đơn:**

* + Mỗi sinh viên có thể nhận nhiều hóa đơn, và mỗi hóa đơn gắn với một sinh viên. Do đó, quan hệ giữa **Sinh viên** và **Hóa đơn** là 1-n.

**Sinh viên - Lịch sử cư trú:**

* + Một **sinh viên** có thể có nhiều **lịch sử cư trú** (quan hệ 1-n), vì sinh viên có thể chuyển phòng nhiều lần trong thời gian lưu trú.

**Người quản lý - Phòng:**

* + Một **người quản lý** có thể quản lý nhiều **phòng** trong một khu vực, và mỗi **phòng** chỉ thuộc quyền quản lý của một người quản lý duy nhất (quan hệ 1-n).
* **Mô hình quan hệ (ERD) dưới dạng sơ đồ:**
* **Sinh viên** (Student) <--(1-n)--> **Phòng** (Room)
* **Sinh viên** (Student) <--(1-n)--> **Hóa đơn** (Invoice)
* **Sinh viên** (Student) <--(1-n)--> **Lịch sử cư trú** (StayHistory)
* **Phòng** (Room) <--(n-1)--> **Người quản lý** (Manager)

A diagram of a computer

Description automatically generated

Hình 1: Sơ đồ quan hệ

## 3.2. Cấu trúc tổ chức dữ liệu đầu vào

### 3.2.1 Dữ liệu vào ra

1. Đầu vào (Input)

Đầu vào của chương trình sẽ bao gồm hai nguồn chính: **nhập từ bàn phím** và **đọc từ file**.

**Nhập từ bàn phím**: Người dùng (quản lý ký túc xá) nhập dữ liệu thủ công vào chương trình thông qua giao diện dòng lệnh hoặc menu. Đây là cách nhập chính cho các thông tin không có sẵn trong hệ thống hoặc khi cần cập nhật dữ liệu mới.

**Đọc từ file**: Dữ liệu từ file sẽ được nạp khi chương trình khởi động hoặc để cập nhật hệ thống với dữ liệu đã được lưu trữ trước đó.

**Thông tin sinh viên**:

* Mã sinh viên (StudentID)
* Họ và tên (FullName)
* Ngày sinh (DOB)
* Giới tính (Gender)
* Số điện thoại (PhoneNumber)
* Khoa và khóa học (Department, Course)
* Mã phòng ký túc (RoomID)

**Thông tin phòng ký túc xá**:

* Mã phòng (RoomID)
* Số lượng giường (TotalBeds)
* Số giường còn trống (AvailableBeds)
* Loại phòng (RoomType)
* Khu vực (Building)
* Giá phòng (RoomPrice)

**Thông tin hóa đơn**:

* Mã hóa đơn (InvoiceID)
* Mã sinh viên (StudentID)
* Ngày lập hóa đơn (InvoiceDate)
* Tổng số tiền (TotalAmount)
* Chi tiết (Description)

**Thông tin người quản lý**:

* Mã quản lý (ManagerID)
* Họ tên (FullName)
* Số điện thoại (PhoneNumber)
* Khu vực quản lý (BuildingID)

1. Đầu ra (Output)

Đầu ra có thể hiển thị theo hai cách chính: **hiển thị ra màn hình** và **xuất ra file**.

**Hiển thị ra màn hình**: Kết quả sẽ được hiển thị trực tiếp lên màn hình qua giao diện dòng lệnh (console) dưới dạng văn bản: Thông tin về sinh viên, phòng ký túc xá, hóa đơn, lịch sử cư trú, hoặc báo cáo tìm kiếm.

**Xuất ra file**: Khi cần lưu trữ hoặc tạo báo cáo, dữ liệu sẽ được xuất ra file dưới dạng TEST.

**Kết quả xử lý:**

* Danh sách sinh viên theo phòng
* Tình trạng phòng
* Báo cáo chi phí hàng tháng
* Danh sách phòng trống
* Tìm kiếm sinh viên

### 3.2.2 Cấu trúc dữ liệu

1. Lựa chọn cấu trúc dữ liệu:

* Để quản lý thông tin sinh viên và phòng trong kí túc xá, cần chọn cấu trúc dữ liệu phù hợp để lưu trữ và thao tác dễ dàng. Các cấu trúc dữ liệu đề xuất:
  + **Struct**: Cấu trúc dữ liệu phù hợp để lưu thông tin cơ bản về sinh viên và phòng (ví dụ như mã sinh viên, tên, phòng ở).
  + **Danh sách liên kết (Linked List)**: Phù hợp cho việc quản lý các sinh viên trong từng phòng. Điều này cho phép dễ dàng thêm hoặc xóa sinh viên mà không phải thực hiện các thao tác phức tạp trên mảng cố định.
  + **Bảng băm (Hash Table)**: Phù hợp để tra cứu nhanh chóng thông tin sinh viên dựa trên mã sinh viên hoặc phòng.

1. Lý giải việc lựa chọn:

* **Struct** giúp dễ dàng quản lý và truy xuất thông tin chi tiết về một đối tượng (sinh viên hoặc phòng).
* **Danh sách liên kết** cho phép dễ dàng thêm, xóa sinh viên khi có thay đổi mà không cần phải dịch chuyển dữ liệu giống như trong mảng cố định.
* **Bảng băm** giúp tra cứu nhanh thông tin sinh viên hoặc phòng dựa trên mã, giảm thiểu thời gian tìm kiếm khi số lượng sinh viên lớn.

1. Mô tả chi tiết thiết kế cấu trúc dữ liệu:

**StructSinh**

struct SinhVien {

string maSV; // Mã sinh viên

string tenSV; // Tên sinh viên

string gioiTinh; // Giới tính

string ngaySinh; // Ngày sinh

string soDienThoai; // Số điện thoại

string maPhong; // Mã phòng ở

};

**StructPhong:**  
struct Phong {

string maPhong; // Mã phòng

int soLuongGiuong; // Số lượng giường

list<string> danhSachSinhVien; // Danh sách mã sinh viên đang ở trong phòng (dùng danh sách liên kết)

};

* **Struct SinhVien** và **Struct Phong** giúp dễ dàng quản lý và truy xuất thông tin sinh viên hoặc phòng ký túc xá.c
* Danh sách liên kết trong “danhSachSinhVien” của mỗi phòng giúp dễ dàng thêm hoặc xóa sinh viên mà không cần dịch chuyển dữ liệu.

# CHƯƠNG TRÌNH VÀ KẾT QUẢ

## 4.1 Các Class:

* class hopdong:
* thuộc tính

+ mahopdong

+ masinhvien

+ maphong

+ ngaybatdau

+ ngayketthuc

* phương thức

+ hopdong():

+ input()

+ display()

+ getmahopdong()

+ getmasinhvien()

+ getmaphong()

+ getngaybatdau()

+ getngayketthuc()

* Class Phòng:
* Thuộc tính:

+ maphong

+ loaiphong

+succhua

+ tienphong

+ danhsachsinhvien

+danhsachhopdong

- Phương thức:

+ hàm tạo:

+ getter:

+ input():

+ getdanhsachsinhvien()

+ addsinhvien(const std::string& masv):.

+ removesinhvien(const std::string& masv):

+ addhopdong(const std::string& mahd):

+ removehopdong(const std::string& mahd):

+ display():

* Class QuanLiKitucxa:
* Thuộc tính

+ danhsachsinhvien

+ danhsachcho

+ danhsachphong

+ danhsachhopdong

+ tendangnhap

+ matkhau

* Phương Thức

+ quanlykitucxa()

+ menu()

+ dangnhap()

+ themhopdong()

+ xemdanhsachhopdong()

+ xemdanhsachcho()

+ themsinhvien()

+ xemdanhsachsinhvien()

+ themphong()

+ xemdanhsachphong()

+ loadfromfile(const std::string& filename)

+ savetofile(const std::string& filename)

* Class SinhVien
* Thuộc tính

+ masinhvien

+ tensinhvien

+ sodienthoai

+ ngaysinh

+ gioitinh

+ maphong

+ mahopdong

* Phương Thức:

+ sinhvien()

+ sinhvien(const std::string& masv, const std::string& tensv, const std::string& sdt, const date& ns, bool gt, const std::string& map, const std::string& mahd)

+getmasinhvien()

+ gettensinhvien()

+ getsodienthoai()

+ getngaysinh()

+ isgioitinh()

+ getmaphong()

+ setmaphong(const std::string& ma)

+ getmahopdong()

+ setmahopdong(const std::string& ma)

+ input()

+ display()

### Giao diện chính của chương trình

Giao diện khi đăng nhập

A black background with white text

Description automatically generated

Hình 2: Đăng Nhập

Giao diện khi đăng nhập thành công:

A black rectangular object with white text

Description automatically generated

Hình 3: Giao diện khi đăng nhập thành công

Thêm Hợp Đồng

A black rectangle with white text

Description automatically generated

Hình 4: Thêm Hợp Đồng

A screenshot of a computer

Description automatically generatedXem danh sách sinh viên

Hình 5: Xem danh sách sinh viên

Xem danh sách phòng

A screenshot of a computer

Description automatically generated

Hình 6: Xem danh sách phòng

Thêm sinh viên

A black screen with white text

Description automatically generated

Hình 7: Thêm sinh viên

Thêm phòng

A black background with white text

Description automatically generated

Hình 8: Thêm phòng

Xem danh sách cho xếp phòng

A black screen with white text

Description automatically generated

Hình 9: Xem danh sách cho xếp phòng

Xem danh sách hợp đồng

A black background with white text

Description automatically generated

Hình 10: Xem danh sách hợp đồng

Thoát khỏi chương trình

A black rectangular object with white text

Description automatically generated

Hình 11: Thoát khỏi chương trình

### Nhận xét

Chương trình hoạt động hiệu quả, có tính ứng dụng cao trong thực tế. Thiết kế giao diện đẹp, dễ nhìn, thân thiện với người dùng, thuận tiện trong khi sử dụng, áp dụng được trong thực tế.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

Mã nguồn quản lý ký túc xá đã được phát triển với mục tiêu tự động hóa các quy trình quản lý, giảm thiểu công việc thủ công và nâng cao hiệu quả làm việc. Hệ thống đã tích hợp các tính năng chính như quản lý thông tin sinh viên, phòng ở, hợp đồng thuê, và thanh toán. Qua quá trình thử nghiệm, mã nguồn đã chứng minh được khả năng hoạt động ổn định và đáp ứng được các yêu cầu đặt ra. Tuy nhiên, hệ thống vẫn còn một số hạn chế như chưa có giao diện cho người dùng cũng như chưa có một số chức năng báo cáo vật dụng trong kí túc xá bị hư chỉ quản lí thay phần quản lí sinh viên ở trên sổ sách.Trong tương lai, cần tiếp tục cải tiến hệ thống để bổ sung thêm các tính năng mới và nâng cao hiệu suất.

## 5.2. Hướng phát triển

Hệ thống quản lý ký túc xá bằng code có thể được phát triển theo hướng hiện đại hóa và tự động hóa nhằm tối ưu hóa quy trình quản lý và nâng cao trải nghiệm người dùng. Một hướng đi tiềm năng là xây dựng hệ thống sử dụng cơ sở dữ liệu tập trung để quản lý thông tin sinh viên, tình trạng phòng, và các dịch vụ hỗ trợ. Đồng thời gia tăng an ninh bằng cách quản lí ra vào của sinh viên có trong kí túc xác. Hệ thống cũng có thể mở rộng với các tính năng như thông báo tự động về hóa đơn, lịch dọn phòng, hoặc các sự kiện ký túc xá. Ngoài ra, phát triển giao diện thân thiện trên nền tảng web và ứng dụng di động sẽ tạo điều kiện thuận lợi cho sinh viên và ban quản lý trong việc truy cập thông tin và xử lý các yêu cầu nhanh chóng.

TÀI LIỆU THAM KHẢO

[1] Tài liệu OOP, TS. Lê Thị Mỹ Hạnh

**[2]** Tài liệu OOP, ThS. Mai Văn Hà

**[3]**https://viblo.asia/p/tim-hieu-ve-lap-trinh-huong-doi-tuong-object-oriented-programming-bxjvZwlBGJZ