**MỤC LỤC**

[Chương 1. Các thuật toán cơ bản và nâng cao 3](#_Toc51517692)

[1.1. Thuật toán cơ bản 3](#_Toc51517693)

[1.1.1. Thuật toán sắp xếp 3](#_Toc51517694)

[1.1.2. Thuật toán tìm kiếm 5](#_Toc51517695)

[1.2. Thuật toán nâng cao 7](#_Toc51517696)

[1.2.1. Thuật toán đệ quy – quay lui 7](#_Toc51517697)

[1.2.2. Kỹ thuật nhánh cận 8](#_Toc51517698)

[1.2.3. Thuật toán tham lam 8](#_Toc51517699)

[1.2.4. Thuật toán Quy hoạch động 8](#_Toc51517700)

[Chương 2. Lập trình hướng đối tượng 9](#_Toc51517701)

[2.1. Các khái niệm trong lập trình hướng đối tượng 9](#_Toc51517702)

[2.1.1. Đối tượng (object) 9](#_Toc51517703)

[2.1.2. Lớp (Class) 9](#_Toc51517704)

[2.2. Các tính chất của lập trình hướng đối tượng 9](#_Toc51517705)

[2.2.1. Tính đóng gói (Encapsulation) 9](#_Toc51517706)

[2.2.2. Tính kế thừa (Inheritance) 10](#_Toc51517707)

[2.2.3. Tính đa hình (Polymorphism) 11](#_Toc51517708)

[2.2.4. Tính trừu tượng (Abstraction) 11](#_Toc51517709)

[Chương 3. Mini Project 12](#_Toc51517710)

[3.1. Mục tiêu 12](#_Toc51517711)

[3.2. Xây dựng Game Pacman theo phương pháp Lập trình hướng đối tượng 12](#_Toc51517712)

[3.2.1. Các đối tượng 12](#_Toc51517713)

[3.2.2. Kỹ thuật xử lý đa luồng trong Game Pacman 13](#_Toc51517714)

**DANH MỤC HÌNH ẢNH**

[Hình 1‑1Ví dụ thuật toán SelectionSort 3](#_Toc51517678)

[Hình 1‑2 Ví dụ thuật toán QuickSort 4](#_Toc51517679)

[Hình 1‑3 Ví dụ thuật toán BinarySearch 5](#_Toc51517680)

[Hình 1‑4 Ví dụ thuật toán tìm kiếm tuần tự 6](#_Toc51517681)

[Hình 1‑5 Ví dụ thuật toán đệ quy 7](#_Toc51517682)

[Hình 2‑1 Ví dụ về Class 10](#_Toc51517683)

# Các thuật toán cơ bản và nâng cao

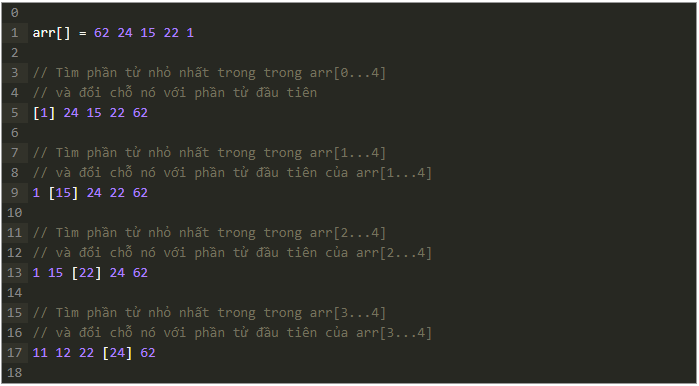
## Thuật toán cơ bản

### Thuật toán sắp xếp

#### Thuật toán sắp xếp chọn

**Ý tưởng thuật toán:** Chọn phần tử nhỏ nhất trong n phần tử ban đầu, đưa phần tử này về vị trí đúng là đầu tiên của dãy hiện hành. Sau đó không quan tâm đến nó nữa, xem dãy hiện hành chỉ còn n-1 phần tử của dãy ban đầu, bắt đầu từ vị trí thứ 2. Lặp lại quá trình trên cho dãy hiện hành đến khi dãy hiện hành chỉ còn một phần tử. Dãy ban đầu có n phần tử, vậy tóm tắt ý tưởng thuật toán là thực hiện n-1 lượt việc đưa phần tử nhỏ nhất trong dãy hiện hành về vị trí đúng ở đầu dãy.

**Ví dụ minh họa:**



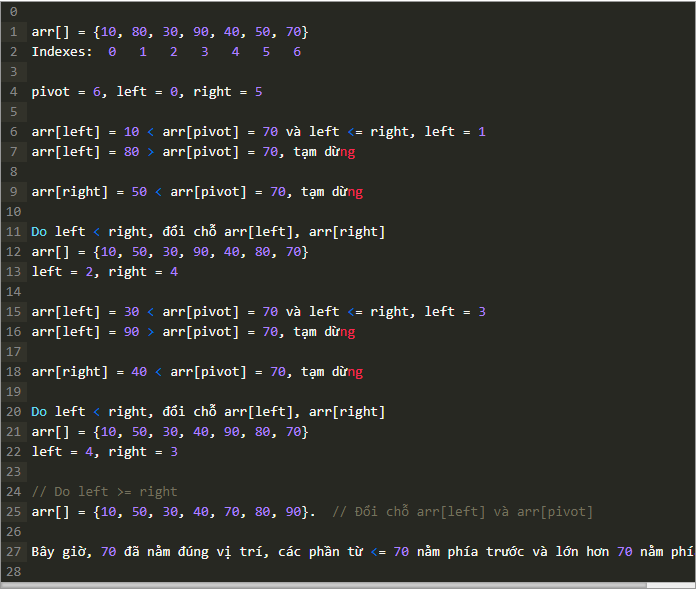
Hình 1‑1Ví dụ thuật toán SelectionSort

#### Thuật toán QuickSort

**Ý tưởng thuật toán:** Chọn một phần tử trong mảng làm điểm đánh dấu(pivot). Thuật toán sẽ thực hiện chia mảng thành các mảng con dựa vào pivot đã chọn. Việc lựa chọn pivot ảnh hưởng rất nhiều tới tốc độ sắp xếp. Nhưng máy tính lại không thể biết khi nào thì nên chọn theo cách nào. Dưới đây là một số cách để chọn pivot thường được sử dụng:

* Luôn chọn phần tử đầu tiên của mảng.
* Luôn chọn phần tử cuối cùng của mảng. (Được sử dụng trong bài viết này)
* Chọn một phần tử random.
* Chọn một phần tử có giá trị nằm giữa mảng(median element).

Ví dụ minh họa:



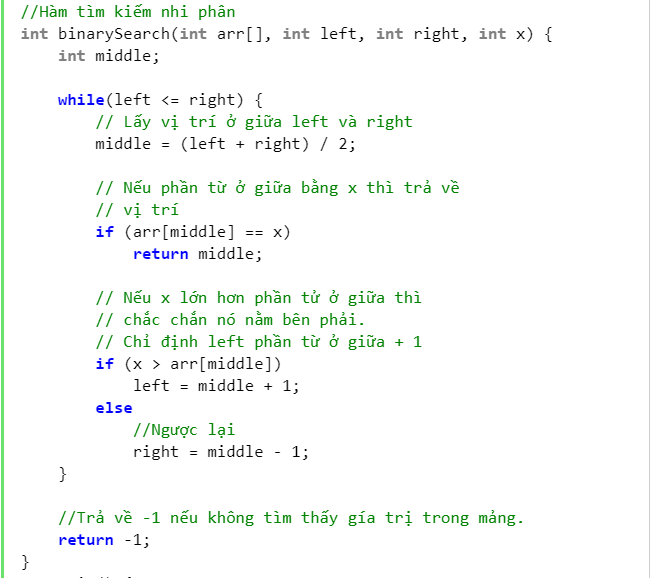
Hình 1‑2 Ví dụ thuật toán QuickSort

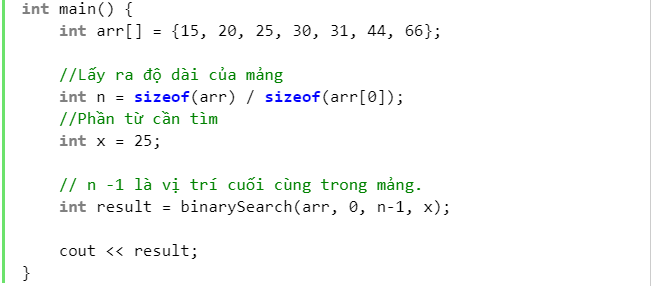
### Thuật toán tìm kiếm

#### Thuật toán BinarySearch

**Ý tưởng thuật toán:** Thuật toán tiến hành so sánh giá trị cần tìm với phần tử đứng giữa mảng. Nếu hai giá trị không bằng nhau, phần nửa mảng không chứa giá trị cần tìm sẽ bị bỏ qua và tiếp tục tìm kiếm trên nửa còn lại, một lần nữa lấy phần tử ở giữa và so sánh với giá trị cần tìm, cứ thế lặp lại cho đến khi tìm thấy giá trị đó. Nếu phép tìm kiếm kết thúc khi nửa còn lại trống thì giá trị cần tìm không có trong mảng.

**Ví dụ minh họa:**



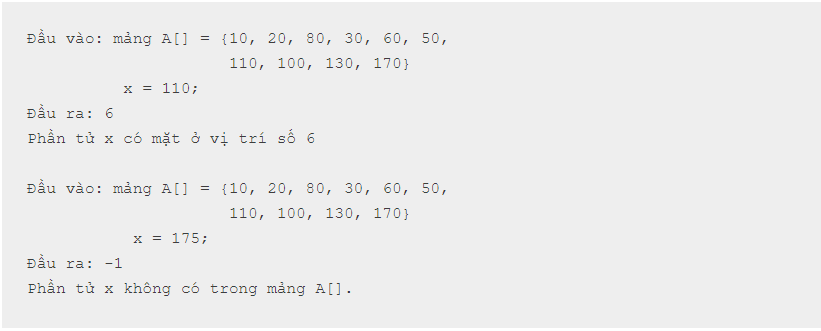


Hình 1‑3 Ví dụ thuật toán BinarySearch

#### Thuật toán tìm kiếm tuần tự

**Ý tưởng thuật toán:** là một phương pháp tìm kiếm một phần tử cho trước trong một danh sách bằng cách duyệt lần lượt từng phần tử của danh sách đó cho đến lúc tìm thấy giá trị mong muốn hay đã duyệt qua toàn bộ danh sách.

**Ví dụ minh họa:**



Hình 1‑4 Ví dụ thuật toán tìm kiếm tuần tự

## Thuật toán nâng cao

### Thuật toán đệ quy – quay lui

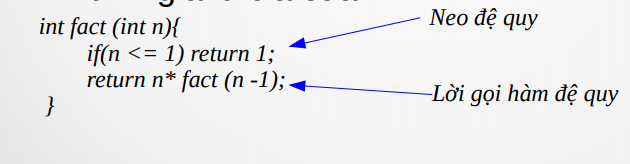
**Ý tưởng thuật toán:**

Một hàm đề quy gồm 2 phần:

* Phần Neo đệ quy: trả về giá trị hàm đệ quy dựa vào tham số đầu vào
* Phần Đệ quy: gọi đến chính hàm đó với một bộ tham số khác

Giải thuật quay lui: Liệt kê các cấu hình thỏa mãn yêu cầu đề bài bằng cách thử tất cả các cách sắp xếp, lựa chọn các phần tử cho mỗi cấu hình.

**Ví dụ minh họa:** Tính giai thừa của n



Hình 1‑5 Ví dụ thuật toán đệ quy

### Kỹ thuật nhánh cận

**Ý tưởng:**

Mỗi khi tìm ra cấu hình cuối cùng , tính một giá trị z\* để đánh giá chất lượng của cấu hình tìm được.

Tại mỗi bước xét vị trí thứ k của giải thuật quay lui, với mỗi ứng viên, tính giá trị để ước lượng chất lượng của cấu hình , nếu có nghĩa là việc chọn ứng viên này là vẫn “ổn”, nếu chọn tiếp các giá trị tiếp theo theo nhánh này vẫn có thể hi vọng đem tới kết quả tốt hơn.

**Ví dụ:** Bài toán người du lịch

### Thuật toán tham lam

**Ý tưởng thuật toán:** Tìm cách tối ưu cục bộ với hy vọng dẫn đến tối ưu toàn cục. Thường không đưa ra được lời giải tối ưu hoàn chỉnh nhưng trong nhiều trường hợp đưa ra được các lời giải có thể chấp nhận.

**Ví dụ minh họa:** Bài toán đổi tiền, Sắp xếp thời gian…

### Thuật toán Quy hoạch động

**Ý tưởng thuật toán:** Thuật toán Quy hoạch động bao gồm 3 giai đoạn:

* Phân rã: Chia bài toán ban đầu thành các bài toán con đồng dạng nhỏ hơn, đến khi bài toán con nhỏ nhất có thể giải trực tiếp -> Bài toán xuất phát là bài toán lớn nhất
* Ghi nhận lời giải: Lưu trữ lời giải các bài toán con vào bảng
* Tổng hợp lời giải: Tìm lời giải của bài toán lớn từ lời giải các bài toán con nhỏ hơn cho đến bài toán lớn nhất (bài toán ban đầu)

**Ví dụ minh họa:** Bài toán tìm mảng con lớn nhất, Bài toán cái túi, Dãy con tăng dài nhất, Xâu con chung dài nhất …

# Lập trình hướng đối tượng

## Các khái niệm trong lập trình hướng đối tượng

### Đối tượng (object)

Một đối tượng bao gồm 2 thông tin: **thuộc tính** và **phương thức**.

* **Thuộc tính**chính là những thông tin, đặc điểm của đối tượng. Ví dụ: con người có các đặc tính như mắt, mũi, tay, chân…
* **Phương thức**là những thao tác, hành động mà đối tượng đó có thể thực hiện. Ví dụ: một người sẽ có thể thực hiện hành động nói, đi, ăn, uống, . . .

### Lớp (Class)

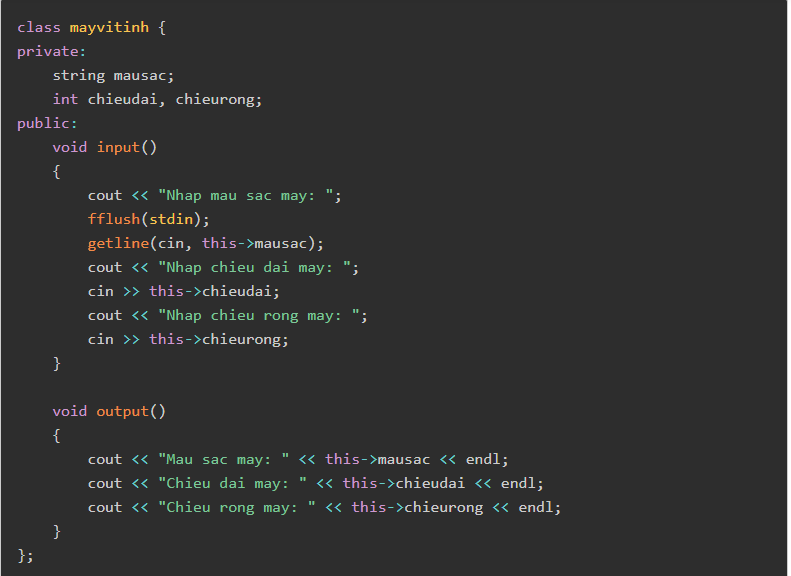
Một class là một tập hợp các đối tượng(objects) và mô tả của các đối tượng có chung các thuộc tính và phương thức.

## Các tính chất của lập trình hướng đối tượng

### Tính đóng gói (Encapsulation)

Là cách để che dấu những tính chất xử lý bên trong của đối tượng, những đối tượng khác không thể tác động trực tiếp làm thay đổi trạng thái  chỉ có thể tác động thông qua các method public của đối tượng đó.

Ví dụ:



Hình 2‑1 Ví dụ về Class

Trong đoạn code trên **tính đóng gói** được thể hiện qua các **thuộc tính** mausac, chieudai, chieurong và **phương thức** input(), output() vào trong class mayvitinh. Bạn không thể truy cập đến các private data hoặc gọi đến private methods của class từ bên ngoài class đó.

### Tính kế thừa (Inheritance)

Tính kế thừa là khả năng cho phép xây dựng một lớp mới dựa trên các định nghĩa của một lớp đã có. Lớp đã có gọi là lớp Cha, lớp mới phát sinh gọi là lớp Con và đương nhiên kế thừa tất cả các thành phần của lớp Cha, có thể chia sẻ hay mở rộng các đặc tính sẵn có mà không phải tiến hành định nghĩa lại. Tuy nhiên, chỉ những thành phần nào mà lớp Cha cho phép thì khi đó lớp con kế thừa mới có thể sử dụng

Cũng với ví dụ là class mayvitinh, class này sẽ có các **thuộc tính**: mausac, chieudai, chieurong. Một class mayAsus, mayAcer sẽ **kế thừa** class mayvitinh do mayAsus cũng có các **thuộc tính**: mausac, chieudai, chieurong.  Thay vì sao chép những thuộc tính này, ta nên đặt chúng vào một lớp chung gọi là lớp cha – trong trường hợp này là mayvitinh và có những lớp con mayAsus, mayAcer **kế thừa** từ nó, tạo ra một mối quan hệ cha/con.

### Tính đa hình (Polymorphism)

Khi một tác vụ được thực hiện theo nhiều cách khác nhau được gọi là tính đa hình.

Ví dụ như class mayvitinh ở trên, mỗi một dòng máy đều kế thừa từ lớp cha mayvitinh nhưng được chạy với một hệ điều hành khác nhau, mayAsus thì được chạy với hệ điều hành Windows, còn mayMacbook sẽ được chạy với hệ diều hành MacOs.

### Tính trừu tượng (Abstraction)

Tính trừu tượng là một tiến trình ẩn các chi tiết trình triển khai và chỉ hiển thị tính năng tới người dùng. Tính trừu tượng cho phép bạn loại bỏ tính chất phức tạp của đối tượng bằng cách chỉ đưa ra các thuộc tính và phương thức cần thiết của đối tượng trong lập trình.

Tính trừu tượng giúp bạn tập trung vào những cốt lõi cần thiết của đối tượng thay vì quan tâm đến cách nó thực hiện.

# Mini Project

## Mục tiêu

Xây dựng game Pacman thiết kế theo phương pháp lập trình hướng đối tượng

Trong game sẽ có các đối tượng, đối tượng người chơi chính sẽ đi ăn các đồ vật có trên bản đồ, mỗi lần ăn thì điểm của người chơi sẽ được tăng lên sao cho không bị va chạm vào các đối tượng bot. Nếu bị va phải một trong các bot thì người chơi sẽ thua.

Game sẽ có các level từ dễ đến khó, với mỗi level, tốc độ di chuyển của các bot sẽ thay đổi nhưng tốc độ di chuyển của người chơi chính thì vẫn sẽ giữ nguyên.

## Xây dựng Game Pacman theo phương pháp Lập trình hướng đối tượng

### Các đối tượng

Đối tượng **Pacman**( người chơi chính)

* Các thuộc tính:
* Possition
* CurrentPoint
* Direction
* Các phương thức:
* Move
* Eat

Đối tượng **Bot**:

* Các thuộc tính:
* Speed
* Possition
* Các phương thức
* FindPath (tìm đường đến mục tiêu)
* Move

### Kỹ thuật xử lý đa luồng trong Game Pacman

Kỹ thuật đa luồng được áp dụng để xử lý khi số lượng Bot ở trên mỗi Map tăng lên. Việc này giúp cho chương trình sẽ không có độ trễ, các Bot sẽ tìm tự tìm kiếm đến mục tiêu của mình là đối tượng Pacman một cách độc lập trên mỗi Thread.

**TÀI LIỆU THAM KHẢO**

[1] Bài giảng môn học Chuyên đề 2- Ths. Lê Đức Quang, Bộ môn KTHT & MMT