# Bài: Con trỏ & Hằng trong C++

*Xem bài học trên website để ủng hộ Kteam: Con trỏ & Hằng trong C++*

*Mọi vấn đề về lỗi website làm ảnh hưởng đến bạn hoặc thắc mắc, mong muốn khóa học mới, nhằm hỗ trợ cải thiện Website. Các bạn vui lòng phản hồi đến Fanpage How Kteam nhé!*

## Dẫn nhập

Ở bài học trước, bạn đã nắm được cách CẤP PHÁT MẢNG ĐỘNG (Dynamically allocating arrays). Với kỹ thuật này, bạn có thể sử dụng mảng với số lượng phần tử lớn hơn và có thể thay đổi trong quá trình chạy chương trình.

Hôm nay, chúng ta sẽ cùng tìm hiểu về **Con trỏ và Hằng (Pointers and const) trong C++**.

---

## Nội dung

Để đọc hiểu bài này tốt nhất các bạn nên có kiến thức cơ bản về:

- CON TRỎ TRONG C++ (Pointer)
- CẤP PHÁT ĐỘNG TRONG C++ (Dynamic memory allocation)

Trong bài ta sẽ cùng tìm hiểu các vấn đề:

- Con trỏ hằng (Pointer to const value)
- Hằng con trỏ (Const pointers)
- Hằng con trỏ hằng (Const pointer to a const value)

---

## Con trỏ hằng (Pointer to const value)

Ở thời điểm này, có thể bạn đã quen thuộc với đoạn code bên dưới:

**C++:**

```
int value = 10;
int *ptr = &value;
*ptr = 20; // gán biến value = 20
```

Đoạn code trên hoạt động một cách bình thường. Tuy nhiên, nếu value là một hằng, chương trình sẽ gây ra lỗi:

**C++:**

```
const int value = 10;
int *ptr = &value; // lỗi biên dịch: cannot convert from 'const int *' to 'int *'
```

Trong ví dụ trên, **value** lúc này là một **hằng**, nghĩa là giá trị của **hằng value** sẽ **không thể thay đổi** trong suốt vòng đời của chương trình. Vì vậy, việc gán một **biến con trỏ** cho một hằng sẽ gây ra lỗi biên dịch.

> **Chú ý:** Thông thường, khi nói một **biến con trỏ** (hoặc con trỏ), điều đó nghĩa là một **con trỏ trỏ đến một biến**.

Để dùng **con trỏ** trỏ đến một **hằng**, C++ cung cấp cho chúng ta khái niệm **con trỏ hằng (Pointer to const value)**.

**Con trỏ hằng (Pointer to const value)** là con trỏ trỏ đến **vùng dữ liệu hằng**. Ta **không thể thay đổi giá trị** mà nó đang trỏ đến. Nhưng có thể cho nó **trỏ đến một địa chỉ vùng nhớ khác.**

**C++:**

```
const int value = 10;
const int *ptr = &value; // ptr là con trỏ hằng
*ptr = 20; // lỗi biên dịch: you cannot assign to a variable that is const
```

Mặt khác, **con trỏ hằng** có thể trỏ đến một **biến**:

**C++:**

```
int value = 10;
const int *ptr = &value; // ptr là con trỏ hằng
```

> **Chú ý:** **Con trỏ hằng** xử lý **giá trị tại địa chỉ mà nó trỏ tới là hằng** khi được truy cập thông qua con trỏ, bất kể biến ban đầu được định nghĩa là hằng hay không.

**C++:**

```
int value = 10;
const int *ptr = &value; // ptr là con trỏ hằng
value = 20; // ok
*ptr = 20; // lỗi, vì ptr là con trỏ hằng
```

Vì **con trỏ hằng** là con trỏ trỏ đến **vùng dữ liệu hằng** (không phải **hằng con trỏ**), nên nó có thể trỏ đến một địa chỉ vùng nhớ khác:

**C++:**

```
int value1 = 5;
const int *ptr = &value1; // ptr là con trỏ hằng

int value2 = 6;
ptr = &value2; // con trỏ hằng có thể trỏ đến địa chỉ khác
```

## Hằng con trỏ (Const pointers)

Ngược lại với con trỏ hằng, **hằng con trỏ (const pointers)** là con trỏ **không thể thay đổi được địa chỉ vùng nhớ** mà nó lưu trữ, nhưng có thể **thay đổi được giá trị mà nó trỏ đến**.

**C++:**

```
int value = 10;
int *const ptr = &value;
```

> **Chú ý:** Như một hằng thông thường, **con trỏ hằng phải được khởi tạo khi khai báo**, và địa chỉ được gán cho con trỏ hằng sẽ **không thể thay đổi về sau**.

**C++:**

```
int value1 = 10;
int value2 = 20;

int *const ptr = &value1; // ptr là hằng con trỏ
ptr = &value2; // lỗi biên dịch, địa chỉ hằng con trỏ trỏ đến không thể thay đổi
```

Tuy nhiên, vì **biến value** được ptr trỏ đến **không phải hằng**, nên **hằng con trỏ ptr** có thể thay đổi giá trị **biến value**:

**C++:**

```
int value = 10;
int *const ptr = &value; // ptr không thể thay đổi địa chỉ vùng nhớ nó lưu trữ
*ptr = 20; // nhưng ptr có thể thay đổi giá trị vùng nhớ nó trỏ đến
```

## Hằng con trỏ hằng (Const pointer to a const value)

Khi kết hợp giữa **hằng con trỏ** và **con trỏ hằng** với nhau, ta có thể có một con trỏ **vừa không thay đổi được địa chỉ vùng nhớ** mà nó lưu trữ, vừa **không thay đổi được giá trị của vùng nhớ** đó.

Để sử dụng **hằng con trỏ hằng**, ta sử dụng **2 từ khóa const** như sau:

**C++:**

```
int value = 10;
const int *const ptr = &value;
```

> **Chú ý:** Một hằng con trỏ hằng không thể trỏ đến một địa chỉ khác, cũng như giá trị mà nó trỏ đến sẽ không thể thay đổi thông qua con trỏ.

## Kết luận

Qua bài học này, bạn đã nắm được các khái niệm liên quan đến **Con trỏ và Hằng (Pointers and const) trong C++**. Ba từ khóa con trỏ hằng, hằng con trỏ và hằng con trỏ hằng tương đối dễ nhầm lẫn khi khai báo và sử dụng, mình tin bài học này đã giúp bạn hiểu rõ hơn về nó.

Con trỏ hằng thường được **sử dụng chủ yếu trong các tham số hàm** (hàm hoặc đối số có mục đích read-only) để giúp đảm bảo hàm **không vô tình thay đổi đối số** được truyền vào trong hàm.

Trong bài tiếp theo, mình sẽ giới thiệu cho các bạn khái niệm BIẾN THAM CHIẾU TRONG C++ (Reference variables).

Cảm ơn các bạn đã theo dõi bài viết. Hãy để lại bình luận hoặc góp ý của mình để phát triển bài viết tốt hơn. Đừng quên "**Luyện tập – Thử thách – Không ngại khó**".